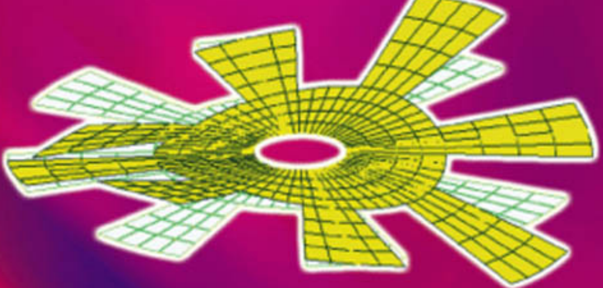
THE FINITE ELEMENT METHOD

A practical course



G. R. LIU and S. S. QUEK

Butterworth-Heinemann
An imprint of Elsevier Science
Linacre House, Jordan Hill, Oxford OX2 8DP
200 Wheeler Road, Burlington MA 01803

First published 2003

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

Library of Congress Cataloguing in Publication Data
A catalogue record for this book is available from the Library of Congress

ISBN 0 7506 5866 5

CONTENTS

BIOGRAPHICAL INFORMATION

DR. G. R. LIU

Dr. Liu received his PhD from Tohoku University, Japan in 1991. He was a Postdoctoral Fellow at Northwestern University, U.S.A. He is currently the Director of the Centre for Advanced Computations in Engineering Science (ACES), National University of Singapore. He is also an Associate Professor at the Department of Mechanical Engineering, National University of Singapore. He authored more than **200** technical publications including two books and 160 international journal papers. He is the recipient of the **Outstanding University Researchers Awards** (1998), and the **Defence Technology Prize** (National award, 1999). He won the **Silver Award** at **CrayQuest 2000** (Nationwide competition in 2000). His research interests include Computational Mechanics, Mesh-free Methods, Nano-scale Computation, Vibration and Wave Propagation in Composites, Mechanics of Composites and Smart Materials, Inverse Problems and Numerical Analysis.

MR. S. S. QUEK

Mr. Quek received his B. Eng. (Hon.) in mechanical engineering from the National University of Singapore in 1999. He did an industrial attachment in the then aeronautics laboratory of DSO National Laboratories, Singapore, gaining much experience in using the finite element method in areas of structural dynamics. He also did research in the areas of wave propagation and infinite domains using the finite element method. In the course of his research, Mr Quek had gained tremendous experience in the applications of the finite element method, especially in using commercially available software like Abaqus. Currently, he is doing research in the field of numerical simulation of quantum dot nanostructures, which will lead to a dissertation for his doctorate degree. To date, he had authored two international journal papers. His research interests include Computational Mechanics, Nano-scale Computation, Vibration and Wave Propagation in Structures and Numerical Analysis.

Preface

In the past few decades, the Finite Element Method (FEM) has been developed into a key indispensable technology in the modelling and simulation of various engineering systems. In the development of an advanced engineering system, engineers have to go through a very rigorous process of modelling, simulation, visualization, analysis, designing, prototyping, testing, and finally, fabrication/construction. As such, techniques related to modelling and simulation in a rapid and effective way play an increasingly important role in building advanced engineering systems, and therefore the application of the FEM has multiplied rapidly.

This book provides unified and detailed course material on the FEM for engineers and university students to solve primarily linear problems in mechanical and civil engineering, with the main focus on structural mechanics and heat transfer. The aim of the book is to provide the necessary concepts, theories and techniques of the FEM for readers to be able to use a commercial FEM package comfortably to solve practical problems and structural analysis and heat transfer. Important fundamental and classical theories are introduced in a straightforward and easy to understand fashion. Modern, state-of-the-art treatment of engineering problems in designing and analysing structural and thermal systems, including microstructural systems, are also discussed. Useful key techniques in FEMs are described in depth, and case studies are provided to demonstrate the theory, methodology, techniques and the practical applications of the FEM. Equipped with the concepts, theories and modelling techniques described in this book, readers should be able to use a commercial FEM software package effectively to solve engineering structural problems in a professional manner.

The general philosophy governing the book is to make all the topics insightful but simple, informative but concise, and theoretical but applicable.

The book unifies topics on mechanics for solids and structures, energy principles, weighted residual approach, the finite element method, and techniques of modelling and computation, as well as the use of commercial software packages. The FEM was originally invented for solving mechanics problems in solids and structures. It is thus appropriate to learn the FEM via problems involving the mechanics of solids. Mechanics for solid structures is a vast subject by itself, which needs volumes of books to describe thoroughly. This book will devote one chapter to try to briefly cover the mechanics of solids and structures by presenting the important basic principles. It focuses on the derivation of key governing equations for three-dimensional solids. Drawings are used to illustrate all the field variables in solids, and the relationships between them. Equations for various types of solids and structures, such as 2D solids, trusses, beams and plates, are then deduced from the general equations for 3D solids. It has been found from our teaching practices that this method of delivering the basics of the mechanics of solid structures is very effective. The introduction of the general 3D equations before examining the other structural components actually gives students a firm fundamental background, from which the other equations can be easily derived and naturally understood. Understanding is then enforced by studying the examples and case studies that are solved using the FEM in other chapters. Our practice of teaching in the past few years has shown that most students managed to understand the fundamental basics of mechanics without too much difficulty, and many of them do not even possess an engineering background.

We have also observed that, over the past few years of handling industrial projects, many engineers are asked to use commercial FEM software packages to simulate engineering systems. Many do not have proper knowledge of the FEM, and are willing to learn via self-study. They thus need a book that describes the FEM in their language, and not in overly obtuse symbols and terminology. Without such a book, many would end up using the software packages blindly like a black box. This book therefore aims to throw light into the black box so that users can see clearly what is going on inside by relating things that are done in the software with the theoretical concepts in the FEM. Detailed description and references are provided in case studies to show how the FEM's formulation and techniques are implemented in the software package.

Being informative need not necessarily mean being exhaustive. A large number of techniques has been developed during the last half century in the area of the FEM. However, very few of them are often used. This book does not want to be an encyclopaedia, but to be informative enough for the useful techniques that are alive. Useful techniques are often very interesting, and by describing the key features of these lively techniques, this book is written to instil an appreciation of them for solving practical problems. It is with this appreciation that we hope readers will be enticed even more to FEM by this book.

Theories can be well accepted and appreciated if their applications can be demonstrated explicitly. The case studies used in the book also serve the purpose of demonstrating the finite element theories. They include a number of recent applications of the FEM for the modelling and simulation of microstructures and microsystems. Most of the case studies are idealized practical problems to clearly bring forward the concepts of the FEM, and will be presented in a manner that make it easier for readers to follow. Following through these case studies, ideally in front of a workstation, helps the reader to understand the important concepts, procedures and theories easily.

A picture tells a thousand words. Numerous drawings and charts are used to describe important concepts and theories. This is very important and will definitely be welcomed by readers, especially those from non-engineering backgrounds.

The book provides practical techniques for using a commercial software package, ABAQUS. The case studies and examples calculated using ABAQUS could be easily repeated using any other commercial software, such as NASTRAN, ANSYS, MARC, etc. Commonly encountered problems in modelling and simulation using commercial software packages are discussed, and rules-of-thumb and guidelines are also provided to solve these problems effectively in professional ways.

Note that the focus of this book is on developing a good understanding of the fundamentals and principles of linear FE analysis. We have chosen ABAQUS as it can easily handle linear analyses, however, with further reading readers could also extend the use of ABAQUS for projects involving non-linear FE analyses too.

Preparing lectures for FEM courses is a very time consuming task, as many drawings and pictures are required to explain all these theories, concepts and techniques clearly. A set of colourful PowerPoint slides for the materials in the book has therefore been produced by the authors for lecturers to use. These slides can be found at the following website: www.bh.com/companions/0750658665. It is aimed at reducing the amount of time taken in preparing lectures using this textbook. All the slides are grouped according to the chapters. The lecturer has the full freedom to cut and add slides according to the level of the class and the hours available for teaching the subject, or to simply use them as provided.

A chapter-by-chapter description of the book is given below.

Chapter 1: Highlights the role and importance of the FEM in computational modelling and simulation required in the design process for engineering systems. The general aspects of computational modelling and simulation of physical problems in engineering are discussed. Procedures for the establishment of mathematical and computational models of physical problems are outlined. Issues related to geometrical simplification, domain discretization, numerical computation and visualization that are required in using the FEM are discussed.

Chapter 2: Describes the basics of mechanics for solids and structures. Important field variables of solid mechanics are introduced, and the key dynamic equations of these variables are derived. Mechanics for 2D and 3D solids, trusses, beams, frames and plates are covered in a concise and easy to understand manner. Readers with a mechanics background may skip this chapter.

Chapter 3: Introduces the general finite element procedure. Concepts of strong and weak forms of a system equations and the construction of shape functions for interpolation of field variables are described. The properties of the shape functions are also discussed with an emphasis on the sufficient requirement of shape functions for establishing FE equations. Hamilton's principle is introduced and applied to establish the general forms of the finite element equations. Methods to solve the finite element equation are discussed for static, eigenvalue analysis, as well as transient analyses.

Chapter 4: Details the procedure used to obtain finite element matrices for truss structures. The procedures to obtain shape functions, the strain matrix, local and global coordinate systems and the assembly of global finite element system equations are described. Very straightforward examples are used to demonstrate a complete and detailed finite element procedure to compute displacements and stresses in truss structures. The reproduction of features and the convergence of the FEM as a reliable numerical tool are revealed through these examples.

Chapter 5: Deals with finite element matrices for beam structures. The procedures followed to obtain shape functions and the strain matrix are described. Elements for thin beam elements are developed. Examples are presented to demonstrate application of the finite element procedure in a beam microstructure.

Chapter 6: Shows the procedure for formulating the finite element matrices for frame structures, by combining the matrices for truss and beam elements. Details on obtaining the transformation matrix and the transformation of matrices between the local and global coordinate systems are described. An example is given to demonstrate the use of frame elements to solve practical engineering problems.

Chapter 7: Formulates the finite element matrices for 2D solids. Matrices for linear triangular elements, bilinear rectangular and quadrilateral elements are derived in detail. Area and natural coordinates are also introduced in the process. Iso-parametric formulation and higher order elements are also described. An example of analysing a micro device is used to study the accuracy and convergence of triangular and quadrilateral elements.

Chapter 8: Deals with finite element matrices for plates and shells. Matrices for rectangular plate elements based on the more practical Reissner–Mindlin plate theory are derived in detail. Shell elements are formulated simply by combining the plate elements and 2D solid plane stress elements. Examples of analysing a micro device using ABAQUS are presented.

Chapter 9: Finite element matrices for 3D solids are developed. Tetrahedron elements and hexahedron elements are formulated in detail. Volume coordinates are introduced in the process. Formulation of higher order elements is also outlined. An example of using 3D elements for modelling a nano-scaled heterostructure system is presented.

Chapter 10: Special purpose elements are introduced and briefly discussed. Crack tip elements for use in many fracture mechanics problems are derived. Infinite elements formulated by mapping and a technique of using structure damping to simulate an infinite domain are both introduced. The finite strip method and the strip element method are also discussed.

Chapter 11: Modelling techniques for the stress analyses of solids and structures are discussed. Use of symmetry, multipoint constraints, mesh compatibility, the modelling of offsets, supports, joints and the imposition of multipoint constraints are all covered. Examples are included to demonstrate use of the modelling techniques.

Chapter 12: A FEM procedure for solving partial differential equations is presented, based on the weighted residual method. In particular, heat transfer problems in 1D and 2D are formulated. Issues in solving heat transfer problems are discussed. Examples are presented to demonstrate the use of ABAQUS for solving heat transfer problems.

Chapter 13: The basics of using ABAQUS are outlined so as to enable new users to get a head start on using the software. An example is presented to outline step-by-step the procedure of writing an ABAQUS input file. Important information required by most FEM software packages is highlighted.

Most of the materials in the book are selected from lecture notes prepared for classes conducted by the first author since 1995 for both under- and post-graduate students. Those lecture notes were written using materials in many excellent existing books on the FEM (listed in the References and many others), and evolved over years of lecturing at the National University of Singapore. The authors wish to express their sincere appreciation to those authors of all the existing FEM books. FEM has been well developed and documented in detail in various existing books. In view of this, the authors have tried their best to limit the information in this book to the necessary minimum required to make it useful for those applying FEM in practice. Readers seeking more advanced theoretical material are advised to refer to books such as those by Zienkiewicz and Taylor. The authors would like to also thank the students for their help in the past few years in developing these courses and studying the subject of the FEM.

G. R. Liu and S. S. Quek

1

COMPUTATIONAL MODELLING

1.1 INTRODUCTION

The Finite Element Method (FEM) has developed into a key, indispensable technology in the modelling and simulation of advanced engineering systems in various fields like housing, transportation, communications, and so on. In building such advanced engineering systems, engineers and designers go through a sophisticated process of modelling, simulation, visualization, analysis, designing, prototyping, testing, and lastly, fabrication. Note that much work is involved before the fabrication of the final product or system. This is to ensure the workability of the finished product, as well as for cost effectiveness. The process is illustrated as a flowchart in Figure 1.1. This process is often iterative in nature, meaning that some of the procedures are repeated based on the results obtained at a current stage, so as to achieve an optimal performance at the lowest cost for the system to be built. Therefore, techniques related to modelling and simulation in a rapid and effective way play an increasingly important role, resulting in the application of the FEM being multiplied numerous times because of this.

This book deals with topics related mainly to modelling and simulation, which are underlined in Figure 1.1. Under these topics, we shall address the computational aspects, which are also underlined in Figure 1.1. The focus will be on the techniques of physical, mathematical and computational modelling, and various aspects of computational simulation. A good understanding of these techniques plays an important role in building an advanced engineering system in a rapid and cost effective way.

So what is the FEM? The FEM was first used to solve problems of stress analysis, and has since been applied to many other problems like thermal analysis, fluid flow analysis, piezoelectric analysis, and many others. Basically, the analyst seeks to determine the distribution of some field variable like the displacement in stress analysis, the temperature or heat flux in thermal analysis, the electrical charge in electrical analysis, and so on. The FEM is a numerical method seeking an approximated solution of the distribution of field variables in the problem domain that is difficult to obtain analytically. It is done by dividing the problem domain into several elements, as shown in Figures 1.2 and 1.3. Known physical laws are then applied to each small element, each of which usually has a very simple geometry. Figure 1.4 shows the finite element approximation for a one-dimensional

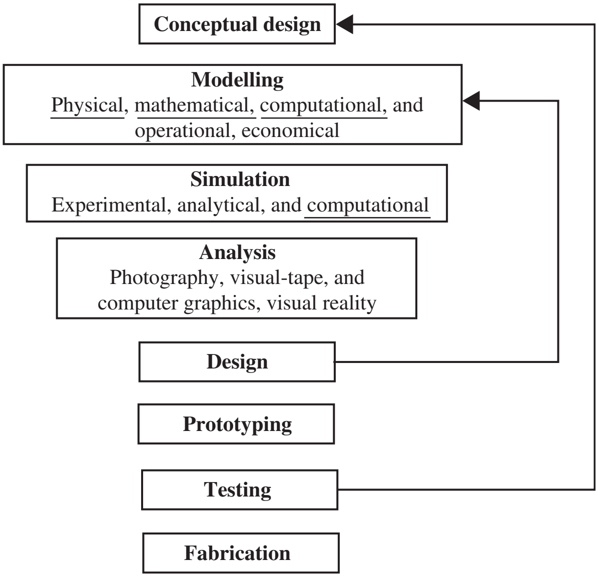

Conceptual design
Modelling
Physical, mathematical, computational, and operational, economical
Simulation
Experimental, analytical, and computational
Analysis
Photography, visual-tape, and computer graphics, visual reality
Design
Prototyping
Testing
Fabrication

Figure 1.1. Processes leading to fabrication of advanced engineering systems.

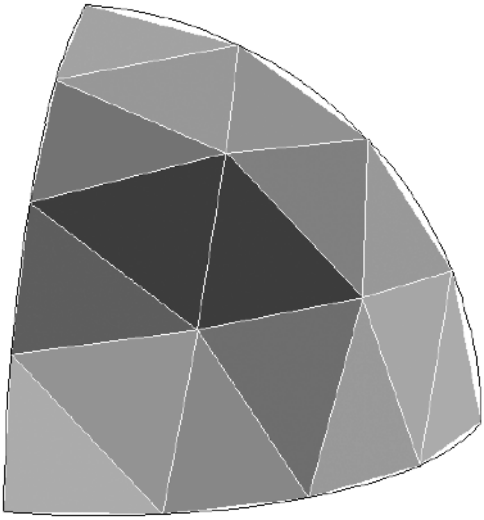


Figure 1.2. Hemispherical section discretized into several shell elements.

case schematically. A continuous function of an unknown field variable is approximated using piecewise linear functions in each sub-domain, called an element formed by nodes. The unknowns are then the discrete values of the field variable at the nodes. Next, proper principles are followed to establish equations for the elements, after which the elements are ‘tied’ to one another. This process leads to a set of linear algebraic simultaneous equations for the entire system that can be solved easily to yield the required field variable.

This book aims to bring across the various concepts, methods and principles used in the formulation of FE equations in a simple to understand manner. Worked examples and case studies using the well known commercial software package ABAQUS will be discussed, and effective techniques and procedures will be highlighted.

1.2 PHYSICAL PROBLEMS IN ENGINEERING

There are numerous physical engineering problems in a particular system. As mentioned earlier, although the FEM was initially used for stress analysis, many other physical problems can be solved using the FEM. Mathematical models of the FEM have been formulated for the many physical phenomena in engineering systems. Common physical problems solved using the standard FEM include:

- Mechanics for solids and structures.
- Heat transfer.

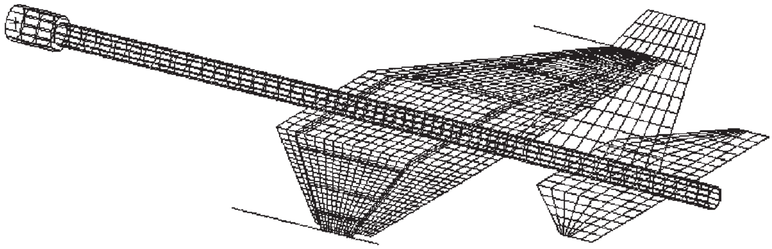


Figure 1.3. Mesh for the design of a scaled model of an aircraft for dynamic testing in the laboratory (Quek 1997–98).

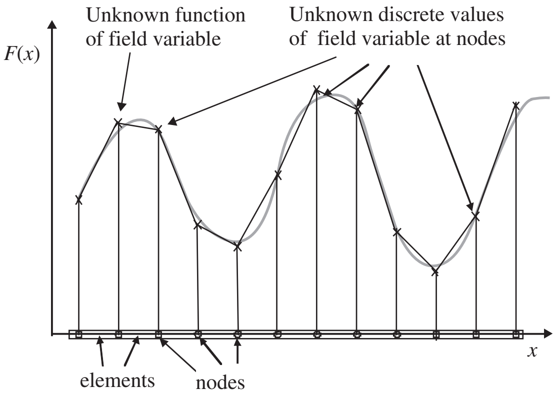


Figure 1.4. Finite element approximation for a one-dimensional case. A continuous function is approximated using piecewise linear functions in each sub-domain/element.

- Acoustics.
- Fluid mechanics.
- Others.

This book first focuses on the formulation of finite element equations for the mechanics of solids and structures, since that is what the FEM was initially designed for. FEM formulations for heat transfer problems are then described. The conceptual understanding of the methodology of the FEM is the most important, as the application of the FEM to all other physical problems utilizes similar concepts.

Computer modelling using the FEM consists of the major steps discussed in the next section.

1.3 COMPUTATIONAL MODELLING USING THE FEM

The behaviour of a phenomenon in a system depends upon the *geometry* or *domain* of the system, the *property* of the *material* or *medium*, and the *boundary*, *initial* and *loading conditions*. For an engineering system, the geometry or domain can be very complex. Further, the boundary and initial conditions can also be complicated. It is therefore, in general, very difficult to solve the governing differential equation via *analytical* means. In practice, most of the problems are solved using numerical methods. Among these, the methods of *domain discretization* championed by the FEM are the most popular, due to its practicality and versatility.

The procedure of computational modelling using the FEM broadly consists of four steps:

- Modelling of the geometry.
- Meshing (discretization).
- Specification of material property.
- Specification of boundary, initial and loading conditions.

1.3.1 Modelling of the Geometry

Real structures, components or domains are in general very complex, and have to be reduced to a manageable geometry. Curved parts of the geometry and its boundary can be modelled using curves and curved surfaces. However, it should be noted that the geometry is eventually represented by a collection of elements, and the curves and curved surfaces are approximated by piecewise straight lines or flat surfaces, if linear elements are used. Figure 1.2 shows an example of a curved boundary represented by the straight lines of the edges of triangular elements. The accuracy of representation of the curved parts is controlled by the number of elements used. It is obvious that with more elements, the representation of the curved parts by straight edges would be smoother and more accurate. Unfortunately, the more elements, the longer the computational time that is required. Hence, due to the constraints on computational hardware and software, it is always necessary to limit the number of elements. As such, compromises are usually made in order to decide on an optimum number of elements used. As a result, fine details of the geometry need to be modelled only if very accurate results are required for those regions. The analysts have to interpret the results of the simulation with these geometric approximations in mind.

Depending on the software used, there are many ways to create a proper geometry in the computer for the FE mesh. Points can be created simply by keying in the coordinates. Lines and curves can be created by connecting the points or nodes. Surfaces can be created by connecting, rotating or translating the existing lines or curves; and solids can be created by connecting, rotating or translating the existing surfaces. Points, lines and curves, surfaces and solids can be translated, rotated or reflected to form new ones.

Graphic interfaces are often used to help in the creation and manipulation of the geometrical objects. There are numerous Computer Aided Design (CAD) software packages used for engineering design which can produce files containing the geometry of the designed engineering system. These files can usually be read in by modelling software packages, which can significantly save time when creating the geometry of the models. However, in many cases, complex objects read directly from a CAD file may need to be modified and simplified before performing meshing or discretization. It may be worth mentioning that there are CAD packages which incorporate modelling and simulation packages, and these are useful for the rapid prototyping of new products.

Knowledge, experience and engineering judgment are very important in modelling the geometry of a system. In many cases, finely detailed geometrical features play only an aesthetic role, and have negligible effects on the performance of the engineering system. These features can be deleted, ignored or simplified, though this may not be true in some cases, where a fine geometrical change can give rise to a significant difference in the simulation results.

An example of having sufficient knowledge and engineering judgment is in the simplification required by the mathematical modelling. For example, a plate has three dimensions geometrically. The plate in the plate theory of mechanics is represented mathematically only in two dimensions (the reason for this will be elaborated in Chapter 2). Therefore, the geometry of a 'mechanics' plate is a two-dimensional flat surface. *Plate elements* will be used in meshing these surfaces. A similar situation can be found in shells. A physical beam has also three dimensions. The beam in the beam theory of mechanics is represented mathematically only in one dimension, therefore the geometry of a 'mechanics' beam is a one-dimensional straight line. *Beam elements* have to be used to mesh the lines in models. This is also true for truss structures.

1.3.2 Meshing

Meshing is performed to discretize the geometry created into small pieces called *elements* or *cells*. Why do we discretize? The rational behind this can be explained in a very straightforward and logical manner. We can expect the solution for an engineering problem to be very complex, and varies in a way that is very unpredictable using functions across the whole domain of the problem. If the problem domain can be divided (*meshed*) into small elements or cells using a set of *grids* or *nodes*, the solution within an element can be approximated very easily using simple functions such as polynomials. The solutions for all of the elements thus form the solution for the whole problem domain.

How does it work? Proper theories are needed for *discretizing* the governing differential equations based on the discretized domains. The theories used are different from problem to problem, and will be covered in detail later in this book for various types of problems. But before that, we need to generate a mesh for the problem domain.

Mesh generation is a very important task of the *pre-process*. It can be a very time consuming task to the analyst, and usually an experienced analyst will produce a more credible mesh for a complex problem. The domain has to be meshed properly into elements of specific shapes such as triangles and quadrilaterals. Information, such as *element connectivity*, must be created during the meshing for use later in the formation of the FEM equations. It is ideal to have an entirely automated mesh generator, but unfortunately this is currently not available in the market. A semi-automatic pre-processor is available for most commercial application software packages. There are also packages designed mainly for meshing. Such packages can generate files of a mesh, which can be read by other modelling and simulation packages.

Triangulation is the most flexible and well-established way in which to create meshes with triangular elements. It can be made almost fully automated for two-dimensional (2D) planes, and even three-dimensional (3D) spaces. Therefore, it is commonly available in most of the pre-processors. The additional advantage of using triangles is the flexibility of modelling complex geometry and its boundaries. The disadvantage is that the accuracy of the simulation results based on triangular elements is often lower than that obtained using quadrilateral elements. Quadrilateral element meshes, however, are more difficulty to generate in an automated manner. Some examples of meshes are given in Figures 1.3–1.7.

1.3.3 Property of Material or Medium

Many engineering systems consist of more than one material. Property of materials can be defined either for a group of elements or each individual element, if needed. For different phenomena to be simulated, different sets of material properties are required. For example, Young's modulus and shear modulus are required for the stress analysis of solids and structures, whereas the thermal conductivity coefficient will be required for a thermal analysis. Inputting of a material's properties into a pre-processor is usually straightforward; all the analyst needs to do is key in the data on material properties and specify either to which region of the geometry or which elements the data applies. However, obtaining these properties is not always easy. There are commercially available material databases to choose from, but experiments are usually required to accurately determine the property of materials to be used in the system. This, however, is outside the scope of this book, and here we assume that the material property is known.

1.3.4 Boundary, Initial and Loading Conditions

Boundary, initial and loading conditions play a decisive role in solving the simulation. Inputting these conditions is usually done easily using commercial pre-processors, and it is often interfaced with graphics. Users can specify these conditions either to the geometrical

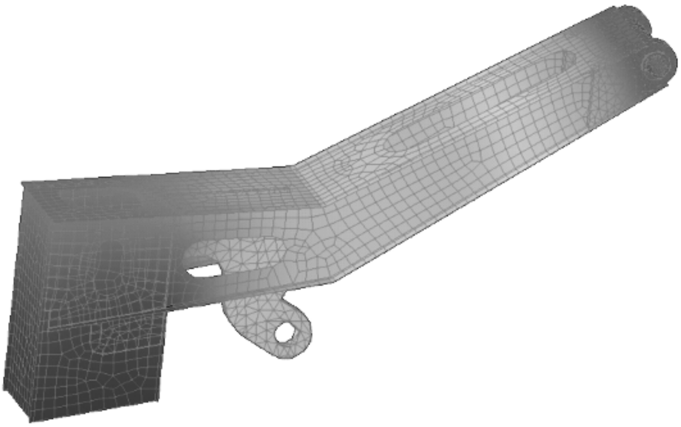


Figure 1.5. Mesh for a boom showing the stress distribution. (Picture used by courtesy of EDS PLM Solutions.)

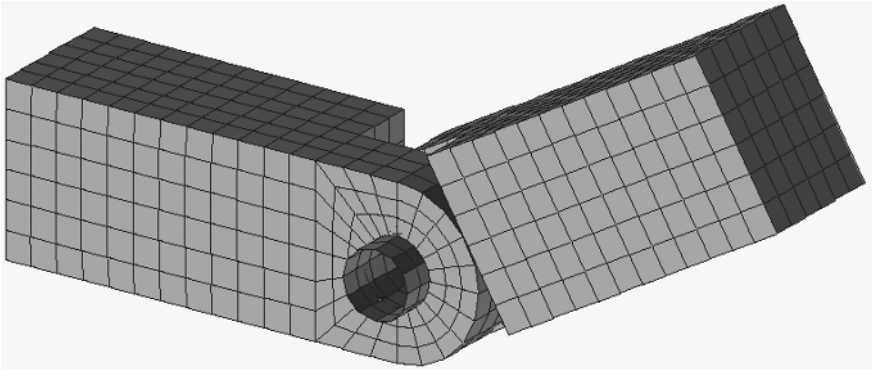


Figure 1.6. Mesh of a hinge joint.

identities (points, lines or curves, surfaces, and solids) or to the elements or grids. Again, to accurately simulate these conditions for actual engineering systems requires experience, knowledge and proper engineering judgments. The boundary, initial and loading conditions are different from problem to problem, and will be covered in detail in subsequent chapters.

1.4 SIMULATION

1.4.1 Discrete System Equations

Based on the mesh generated, a set of discrete simultaneous system equations can be formulated using existing approaches. There are a few types of approach for establishing the

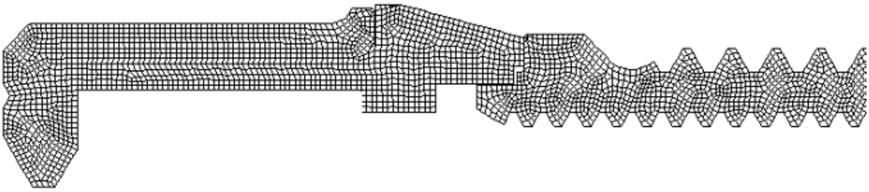


Figure 1.7. Axisymmetric mesh of part of a dental implant (The CeraOne® abutment system, Nobel Biocare).

simultaneous equations. The first is based on energy principles, such as Hamilton's principle (Chapter 3), the minimum potential energy principle, and so on. The traditional Finite Element Method (FEM) is established on these principles. The second approach is the weighted residual method, which is also often used for establishing FEM equations for many physical problems and will be demonstrated for heat transfer problems in Chapter 12. The third approach is based on the Taylor series, which led to the formation of the traditional Finite Difference Method (FDM). The fourth approach is based on the control of conservation laws on each finite volume (elements) in the domain. The Finite Volume Method (FVM) is established using this approach. Another approach is by integral representation, used in some mesh free methods [Liu, 2002]. Engineering practice has so far shown that the first two approaches are most *often* used for solids and structures, and the other two approaches are *often* used for fluid flow simulation. However, the FEM has also been used to develop commercial packages for fluid flow and heat transfer problems, and FDM can be used for solids and structures. It may be mentioned without going into detail that the mathematical foundation of all these three approaches is the *residual method*. An appropriate choice of the test and trial functions in the residual method can lead to the FEM, FDM or FVM formulation.

This book first focuses on the formulation of finite element equations for the mechanics of solids and structures based on energy principles. FEM formulations for heat transfer problems are then described, so as to demonstrate how the weighted residual method can be used for deriving FEM equations. This will provide the basic knowledge and key approaches into the FEM for dealing with other physical problems.

1.4.2 Equation Solvers

After the computational model has been created, it is then fed to a *solver* to solve the discretized system, simultaneous equations for the field variables at the nodes of the mesh. This is the most computer hardware demanding process. Different software packages use different algorithms depending upon the physical phenomenon to be simulated. There are two very important considerations when choosing algorithms for solving a system of equations: one is the storage required, and another is the CPU (Central Processing Unit) time needed.

There are two main types of method for solving simultaneous equations: direct methods and iterative methods. Commonly used direct methods include the Gauss elimination method and the LU decomposition method. Those methods work well for relatively small equation systems. Direct methods operate on fully assembled system equations, and therefore demand larger storage space. It can also be coded in such a way that the assembling of the equations is done only for those elements involved in the current stage of equation solving. This can reduce the requirements on storage significantly.

Iterative methods include the Gauss–Jacobi method, the Gauss–Deidel method, the SOR method, generalized conjugate residual methods, the line relaxation method, and so on. These methods work well for relatively larger systems. Iterative methods are often coded in such a way as to avoid full assembly of the system matrices in order to save significantly on the storage. The performance in terms of the rate of convergence of these methods is usually very problem-dependent. In using iterative methods, pre-conditioning plays a very important role in accelerating the convergence process.

For nonlinear problems, another iterative loop is needed. The nonlinear equation has to be properly formulated into a linear equation in the iteration. For time-dependent problems, time stepping is also required, i.e. first solving for the solution at an initial time (or it could be prescribed by the analyst), then using this solution to march forward for the solution at the next time step, and so on until the solution at the desired time is obtained. There are two main approaches to time stepping: the implicit and explicit approaches. Implicit approaches are usually more stable numerically but less efficient computationally than explicit approaches. Moreover, contact algorithms can be developed more easily using explicit methods. Details on these issues will be given in Chapter 3.

1.5 VISUALIZATION

The result generated after solving the system equation is usually a vast volume of digital data. The results have to be visualized in such a way that it is easy to interpolate, analyse and present. The visualization is performed through a so-called post-processor, usually packaged together with the software. Most of these processors allow the user to display 3D objects in many convenient and colourful ways on-screen. The object can be displayed in the form of wire-frames, group of elements, and groups of nodes. The user can rotate, translate and zoom into and out from the objects. Field variables can be plotted on the object in the form of contours, fringes, wire-frames and deformations. Usually, there are also tools available for the user to produce iso-surfaces, or vector fields of variable(s). Tools to enhance the visual effects are also available, such as shading, lighting and shrinking. Animations and movies can also be produced to simulate the dynamic aspects of a problem. Outputs in the form of tables, text files and x–y plots are also routinely available. Throughout this book, worked examples with various post-processed results are given.

Advanced visualization tools, such as virtual reality, are available nowadays. These advanced tools allow users to display objects and results in a much realistic three-dimensional form. The platform can be a goggle, inversion desk or even in a room. When the object is immersed in a room, analysts can walk through the object, go to the exact location and view the results. Figures 1.8 and 1.9 show an airflow field in virtually designed buildings.

Figure 1.8. Air flow field in a virtually designed building (courtesy of the Institute of High Performance Computing).

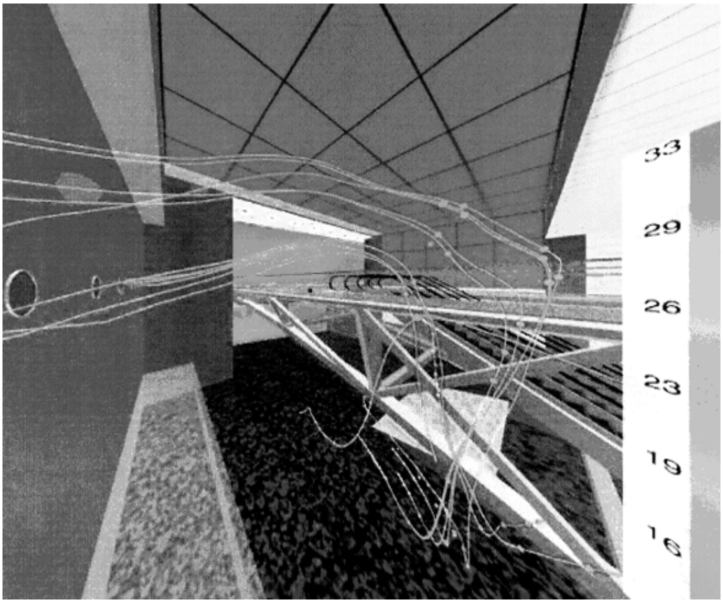


Figure 1.9. Air flow field in a virtually designed building system (courtesy of the Institute of High Performance Computing).

In a nutshell, this chapter has briefly given an introduction to the steps involved in computer modelling and simulation. With rapidly advancing computer technology, use of the computer as a tool in the FEM is becoming indispensable. Nevertheless, subsequent chapters discuss what is actually going on in the computer when performing a FEM analysis.

2

INTRODUCTION TO MECHANICS FOR SOLIDS AND STRUCTURES

2.1 INTRODUCTION

The concepts and classical theories of the mechanics of solids and structures are readily available in textbooks (see e.g. Timoshenko, 1940; Fung, 1965; Timoshenko and Goodier, 1970; etc.). This chapter tries to introduce these basic concepts and classical theories in a brief and easy to understand manner. Solids and structures are stressed when they are subjected to *loads* or *forces*. The *stresses* are, in general, not uniform, and lead to *strains*, which can be observed as either *deformation* or *displacement*. *Solid mechanics* and *structural mechanics* deal with the relationships between stresses and strains, displacements and forces, stresses (strains) and forces for given boundary conditions of solids and structures. These relationships are vitally important in modelling, simulating and designing engineered structural systems.

Forces can be static and/or dynamic. *Statics* deals with the mechanics of solids and structures subjected to static loads such as the deadweight on the floor of buildings. Solids and structures will experience vibration under the action of dynamic forces varying with time, such as excitation forces generated by a running machine on the floor. In this case, the stress, strain and displacement will be functions of time, and the principles and theories of *dynamics* must apply. As statics can be treated as a special case of dynamics, the static equations can be derived by simply dropping out the dynamic terms in the general, dynamic equations. This book will adopt this approach of deriving the dynamic equation first, and obtaining the static equations directly from the dynamic equations derived.

Depending on the property of the material, solids can be *elastic*, meaning that the deformation in the solids disappears fully if it is unloaded. There are also solids that are considered *plastic*, meaning that the deformation in the solids cannot be fully recovered when it is unloaded. *Elasticity* deals with solids and structures of elastic materials, and *plasticity* deals with those of plastic materials. The scope of this book deals mainly with solids and structures of elastic materials. In addition, this book deals only with problems of very small deformation, where the deformation and load has a linear relationship. Therefore, our problems will mostly be *linear elastic*.

Materials can be *anisotropic*, meaning that the material property varies with direction. Deformation in anisotropic material caused by a force applied in a particular direction may be different from that caused by the same force applied in another direction. Composite materials are often anisotropic. Many material constants have to be used to define the material property of anisotropic materials. Many engineering materials are, however, *isotropic*, where the material property is not direction-dependent. Isotropic materials are a special case of anisotropic material. There are only two independent material constants for isotropic material. Usually, the two most commonly used material constants are the Young's modulus and the Poisson's ratio. This book deals mostly with isotropic materials. Nevertheless, most of the formulations are also applicable to anisotropic materials.

Boundary conditions are another important consideration in mechanics. There are displacement and force boundary conditions for solids and structures. For heat transfer problems there are temperature and convection boundary conditions. Treatment of the boundary conditions is a very important topic, and will be covered in detail in this chapter and also throughout the rest of the book.

Structures are made of structural components that are in turn made of solids. There are generally four most commonly used structural components: truss, beam, plate, and shell, as shown in Figure 2.1. In physical structures, the main purpose of using these structural components is to effectively utilize the material and reduce the weight and cost of the structure. A practical structure can consist of different types of structural components, including solid blocks. Theoretically, the principles and methodology in solid mechanics can be applied to solve a mechanics problem for all structural components, but this is usually not a very efficient method. Theories and formulations for taking geometrical advantages of the structural components have therefore been developed. Formulations for a truss, a beam, 2D solids and plate structures will be discussed in this chapter. In engineering practice, plate elements are often used together with two-dimensional solids for modelling shells. Therefore in this book, shell structures will be modelled by combining plate elements and 2D solid elements.

2.2 EQUATIONS FOR THREE-DIMENSIONAL SOLIDS

2.2.1 Stress and Strain

Let us consider a continuous three-dimensional (3D) elastic solid with a volume V and a surface S, as shown in Figure 2.2. The surface of the solid is further divided into two types of surfaces: a surface on which the external forces are prescribed is denoted S_F; and surface on which the displacements are prescribed is denoted S_d. The solid can also be loaded by body force f_b and surface force f_s in any distributed fashion in the volume of the solid.

At any point in the solid, the components of stress are indicated on the surface of an 'infinitely' small cubic volume, as shown in Figure 2.3. On each surface, there will be the normal stress component, and two components of shearing stress. The sign convention for the subscript is that the first letter represents the surface on which the stress is acting, and the second letter represents the direction of the stress. The directions of the stresses shown in the figure are taken to be the positive directions. By taking moments of forces about the central axes of the cube at the state of equilibrium, it is easy to confirm that

$$\sigma_{xy} = \sigma_{yx}; \quad \sigma_{xz} = \sigma_{zx}; \quad \sigma_{zy} = \sigma_{yz} \tag{2.1}$$

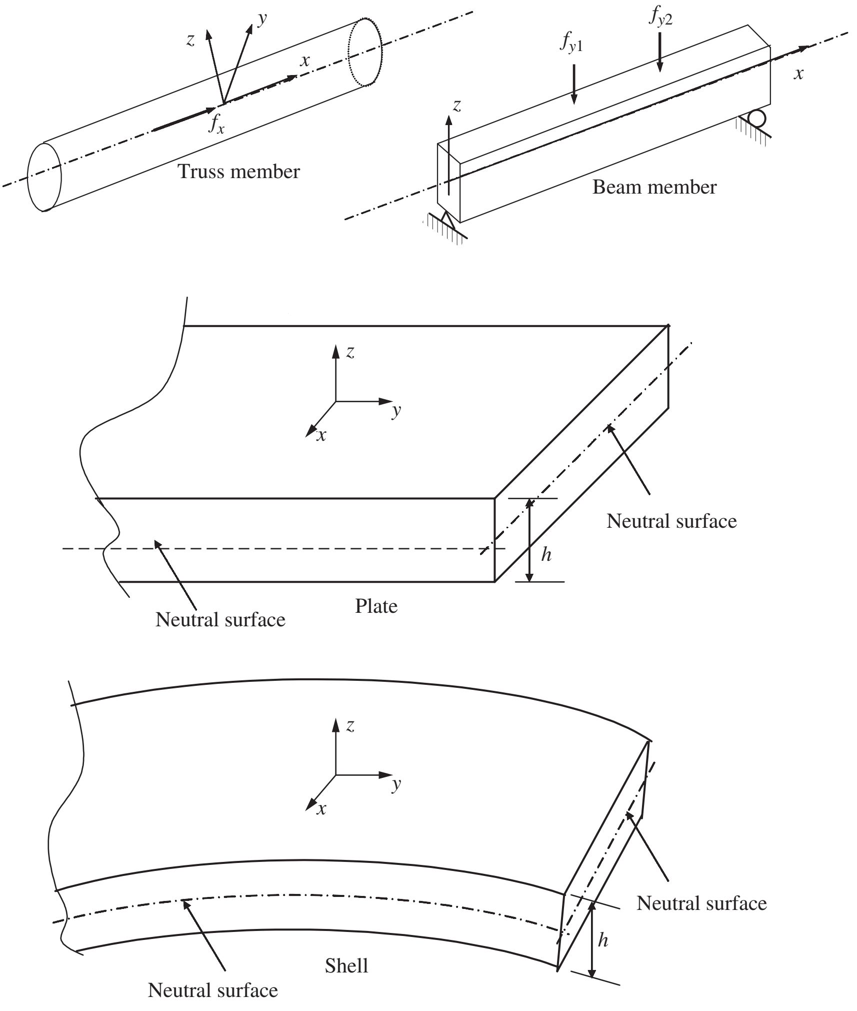


Figure 2.1. Four common types of structural components. Their geometrical features are made use of to derive dimension reduced system equations.

Therefore, there are six stress components in total at a point in solids. These stresses are often called a *stress tensor*. They are often written in a vector form of

$$\boldsymbol{\sigma}^T = \{\sigma_{xx} \quad \sigma_{yy} \quad \sigma_{zz} \quad \sigma_{yz} \quad \sigma_{xz} \quad \sigma_{xy}\} \tag{2.2}$$

Corresponding to the six stress tensors, there are six strain components at any point in a solid, which can also be written in a similar vector form of

$$\boldsymbol{\varepsilon}^T = \{\varepsilon_{xx} \quad \varepsilon_{yy} \quad \varepsilon_{zz} \quad \varepsilon_{yz} \quad \varepsilon_{xz} \quad \varepsilon_{xy}\} \tag{2.3}$$

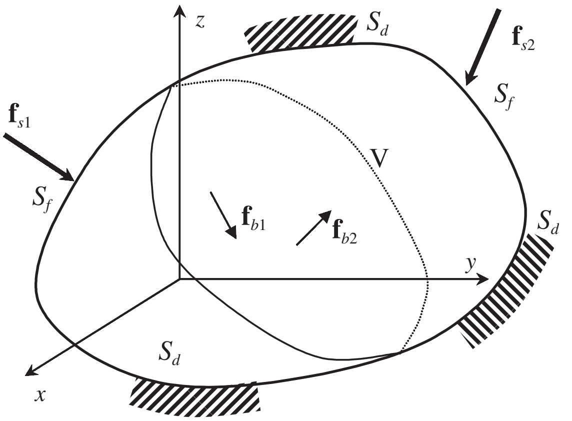


Figure 2.2. Solid subjected to forces applied within the solid (body force) and on the surface of the solid (surface force).

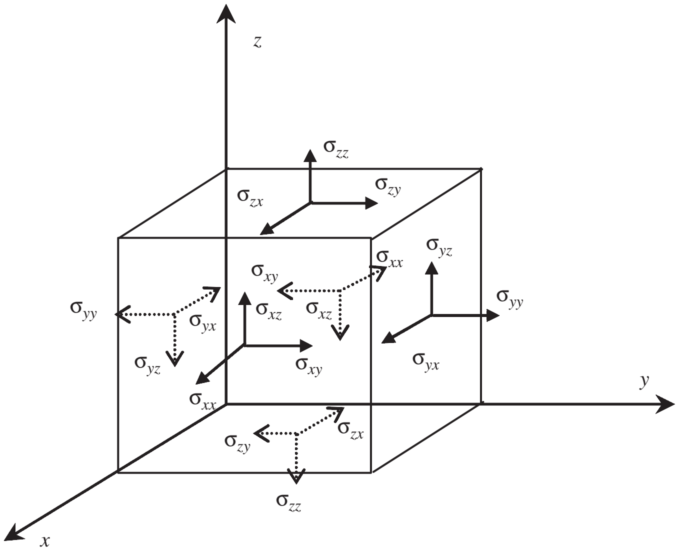


Figure 2.3. Six independent stress components at a point in a solid viewed on the surfaces of an infinitely small cubic block.

Strain is the change of displacement per unit length, and therefore the components of strain can be obtained from the derivatives of the displacements as follows:

$$\begin{aligned}
&\varepsilon_{xx} = \frac{\partial u}{\partial x}; \quad \varepsilon_{yy} = \frac{\partial v}{\partial y}; \quad \varepsilon_{zz} = \frac{\partial w}{\partial z}; \\
&\varepsilon_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}; \quad \varepsilon_{xz} = \frac{\partial u}{\partial z} + \frac{\partial w}{\partial x}; \quad \varepsilon_{yz} = \frac{\partial v}{\partial z} + \frac{\partial w}{\partial y}
\end{aligned} \tag{2.4}$$

where u, v and w are the displacement components in the x, y and z directions, respectively. The six strain–displacement relationships in Eq. (2.4) can be rewritten in the following matrix form:

$$\boldsymbol{\varepsilon} = \mathbf{LU} \tag{2.5}$$

where $\mathbf{U}$ is the displacement vector, and has the form of

$$\mathbf{U} = \begin{Bmatrix} u \\ v \\ w \end{Bmatrix} \tag{2.6}$$

and $\mathbf{L}$ is a matrix of partial differential operators obtained simply by inspection on Eq. (2.4):

$$\mathbf{L} = \begin{bmatrix} \partial/\partial x & 0 & 0 \\ 0 & \partial/\partial y & 0 \\ 0 & 0 & \partial/\partial z \\ 0 & \partial/\partial z & \partial/\partial y \\ \partial/\partial z & 0 & \partial/\partial x \\ \partial/\partial y & \partial/\partial x & 0 \end{bmatrix} \tag{2.7}$$

2.2.2 Constitutive Equations

The constitutive equation gives the relationship between the stress and strain in the material of a solid. It is often termed Hooke's law. The generalised Hooke's law for general anisotropic materials can be given in the following matrix form:

$$\boldsymbol{\sigma} = \mathbf{c}\boldsymbol{\varepsilon} \tag{2.8}$$

where $\mathbf{c}$ is a matrix of material constants, which are normally obtained through experiments. The constitutive equation can be written explicitly as

$$\begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{zz} \\ \sigma_{yz} \\ \sigma_{xz} \\ \sigma_{xy} \end{Bmatrix} = \begin{bmatrix} c_{11} & c_{12} & c_{13} & c_{14} & c_{15} & c_{16} \\ & c_{22} & c_{23} & c_{24} & c_{25} & c_{26} \\ & & c_{33} & c_{34} & c_{35} & c_{36} \\ & & & c_{44} & c_{45} & c_{46} \\ & sy. & & & c_{55} & c_{56} \\ & & & & & c_{66} \end{bmatrix} \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{zz} \\ \varepsilon_{yz} \\ \varepsilon_{xz} \\ \varepsilon_{xy} \end{Bmatrix} \tag{2.9}$$

Note that, since $c_{ij} = c_{ji}$, there are altogether 21 independent material constants c_{ij}, which is the case for a fully anisotropic material. For isotropic materials, however, $\mathbf{c}$ can be reduced to

$$\mathbf{c} = \begin{bmatrix} c_{11} & c_{12} & c_{12} & 0 & 0 & 0 \\ & c_{11} & c_{12} & 0 & 0 & 0 \\ & & c_{11} & 0 & 0 & 0 \\ & & & (c_{11} - c_{12})/2 & 0 & 0 \\ & sy. & & & (c_{11} - c_{12})/2 & 0 \\ & & & & & (c_{11} - c_{12})/2 \end{bmatrix} \tag{2.10}$$

where

$$c_{11} = \frac{E(1-\nu)}{(1-2\nu)(1+\nu)}; \quad c_{12} = \frac{E\nu}{(1-2\nu)(1+\nu)}; \quad \frac{c_{11}-c_{12}}{2} = G \tag{2.11}$$

in which E, ν and G are Young's modulus, Poisson's ratio, and the shear modulus of the material, respectively. There are only two independent constants among these three constants. The relationship between these three constants is

$$G = \frac{E}{2(1+\nu)} \tag{2.12}$$

That is to say, for any isotropic material, given any two of the three constants, the other one can be calculated using the above equation.

2.2.3 Dynamic Equilibrium Equation

To formulate the dynamic equilibrium equations, let us consider an infinitely small block of solid, as shown in Figure 2.4. As in forming all equilibrium equations, equilibrium of forces is required in all directions. Note that, since this is a general, dynamic system, we have to consider the inertial forces of the block. The equilibrium of forces in the x direction gives

$$\begin{aligned}(\sigma_{xx} + \mathrm{d}\sigma_{xx})\,\mathrm{d}y\,\mathrm{d}z - \sigma_{xx}\,\mathrm{d}y\,\mathrm{d}z + (\sigma_{yx} + \mathrm{d}\sigma_{yx})\,\mathrm{d}x\,\mathrm{d}z - \sigma_{yx}\,\mathrm{d}x\,\mathrm{d}z \\ + (\sigma_{zx} + \mathrm{d}\sigma_{zx})\,\mathrm{d}x\,\mathrm{d}y - \sigma_{zx}\,\mathrm{d}x\,\mathrm{d}y + \underbrace{f_x}_{\text{external force}} = \underbrace{\rho\ddot{u}\,\mathrm{d}x\,\mathrm{d}y\,\mathrm{d}z}_{\text{inertial force}}\end{aligned} \tag{2.13}$$

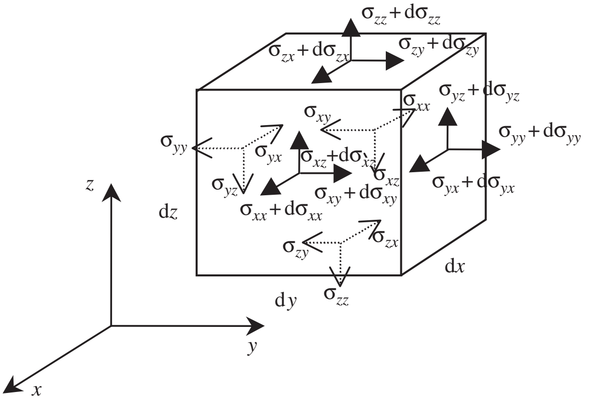


Figure 2.4. Stresses on an infinitely small block. Equilibrium equations are derived based on this state of stresses.

where the term on the right-hand side of the equation is the inertial force term, and f_x is the external body force applied at the centre of the small block. Note that

$$\mathrm{d}\sigma_{xx} = \frac{\partial \sigma_{xx}}{\partial x}\,\mathrm{d}x, \quad \mathrm{d}\sigma_{yx} = \frac{\partial \sigma_{yx}}{\partial y}\,\mathrm{d}y, \quad \mathrm{d}\sigma_{zx} = \frac{\partial \sigma_{zx}}{\partial z}\,\mathrm{d}z \tag{2.14}$$

Hence, Eq. (2.13) becomes one of the equilibrium equations, written as

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{yx}}{\partial y} + \frac{\partial \sigma_{zx}}{\partial z} + f_x = \rho \ddot{u} \tag{2.15}$$

Similarly, the equilibrium of forces in the y and z directions results in two other equilibrium equations:

$$\frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + \frac{\partial \sigma_{zy}}{\partial z} + f_y = \rho \ddot{v} \tag{2.16}$$

$$\frac{\partial \sigma_{xz}}{\partial x} + \frac{\partial \sigma_{yz}}{\partial y} + \frac{\partial \sigma_{zz}}{\partial z} + f_z = \rho \ddot{w} \tag{2.17}$$

The equilibrium equations, Eqs. (2.15) to (2.17), can be written in a concise matrix form

$$\mathbf{L}^T \sigma + \mathbf{f}_b = \rho \ddot{\mathbf{U}} \tag{2.18}$$

where $\mathbf{f}_b$ is the vector of external body forces in the x, y and z directions:

$$\mathbf{f}_b = \begin{Bmatrix} f_x \\ f_y \\ f_z \end{Bmatrix} \tag{2.19}$$

Using Eqs. (2.5) and (2.8), the equilibrium equation Eq. (2.18) can be further written in terms of displacements:

$$\mathbf{L}^T \mathbf{c} \mathbf{L} \mathbf{U} + \mathbf{f}_b = \rho \ddot{\mathbf{U}} \tag{2.20}$$

The above is the general form of the dynamic equilibrium equation expressed as a matrix equation. If the loads applied on the solid are static, the only concern is then the static status of the solid. Hence, the static equilibrium equation can be obtained simply by dropping the dynamic term in Eq. (2.20), which is the inertial force term:

$$\mathbf{L}^T \mathbf{c} \mathbf{L} \mathbf{U} + \mathbf{f}_b = 0 \tag{2.21}$$

2.2.4 Boundary Conditions

There are two types of boundary conditions: displacement (*essential*) and force (*natural*) boundary conditions. The displacement boundary condition can be simply written as

$$u = \bar{u} \quad \text{and/or} \quad v = \bar{v} \quad \text{and/or} \quad w = \bar{w} \tag{2.22}$$

on displacement boundaries. The bar stands for the prescribed value for the displacement component. For most of the actual simulations, the displacement is used to describe the support or constraints on the solid, and hence the prescribed displacement values are often zero. In such cases, the boundary condition is termed as a *homogenous boundary condition*. Otherwise, they are *inhomogeneous boundary conditions*.

The force boundary condition are often written as

$$\mathbf{n}\boldsymbol{\sigma} = \bar{\mathbf{t}} \tag{2.23}$$

on force boundaries, where **n** is given by

$$\mathbf{n} = \begin{bmatrix} n_x & 0 & 0 & 0 & n_z & n_y \\ 0 & n_y & 0 & n_z & 0 & n_x \\ 0 & 0 & n_z & n_y & n_x & 0 \end{bmatrix}$$

in which n_i $(i = x, y, z)$ are cosines of the outwards normal on the boundary. The bar stands for the prescribed value for the force component. A force boundary condition can also be both homogenous and inhomogeneous. If the condition is homogeneous, it implies that the boundary is a free surface.

The reader may naturally ask why the displacement boundary condition is called an essential boundary condition and the force boundary condition is called a natural boundary conditions. The terms 'essential' and 'natural' come from the use of the so-called *weak form* formulation (such as the weighted residual method) for deriving system equations. In such a formulation process, the displacement condition has to be satisfied first before derivation starts, or the process will fail. Therefore, the displacement condition is *essential*. As long as the essential (displacement) condition is satisfied, the process will lead to the equilibrium equations as well as the force boundary conditions. This means that the force boundary condition is *naturally* derived from the process, and it is therefore called the natural boundary condition. Since the terms essential and natural boundary do not describe the physical meaning of the problem, it is actually a mathematical term, and they are also used for problems other than in mechanics.

Equations obtained in this section are applicable to 3D solids. The objective of most analysts is to solve the equilibrium equations and obtain the solution of the field variable, which in this case is the displacement. Theoretically, these equations can be applied to all other types of structures such as trusses, beams, plates and shells, because physically they are all 3D in nature. However, treating all the structural components as 3D solids makes computation very expensive, and sometimes practically impossible. Therefore, theories for taking geometrical advantage of different types of solids and structural components have been developed. Application of these theories in a proper manner can reduce the analytical and computational effort drastically. A brief description of these theories is given in the following sections.

2.3 EQUATIONS FOR TWO-DIMENSIONAL SOLIDS

2.3.1 Stress and Strain

Three-dimensional problems can be drastically simplified if they can be treated as a two-dimensional (2D) solid. For representation as a 2D solid, we basically try to remove one coordinate (usually the z-axis), and hence assume that all the dependent variables are independent of the z-axis, and all the external loads are independent of the z coordinate, and applied only in the x–y plane. Therefore, we are left with a system with only two coordinates, the x and the y coordinates. There are primarily two types of 2D solids. One is a *plane stress* solid, and another is a *plane strain* solid. Plane stress solids are solids whose thickness in the z direction is very small compared with dimensions in the x and y directions. External forces are applied only in the x–y plane, and stresses in the z direction (σ_{zz}, σ_{xz}, σ_{yz}) are all zero, as shown in Figure 2.5. Plane strain solids are those solids whose thickness in the z direction is very large compared with the dimensions in the x and y directions. External forces are applied evenly along the z axis, and the movement in the z direction at any point is constrained. The strain components in the z direction (ε_{zz}, ε_{xz}, ε_{yz}) are, therefore, all zero, as shown in Figure 2.6.

Note that for the plane stress problems, the strains ε_{xz} and ε_{yz} are zero, but ε_{zz} will not be zero. It can be recovered easily using Eq.(2.9) after the in-plan stresses are obtained.

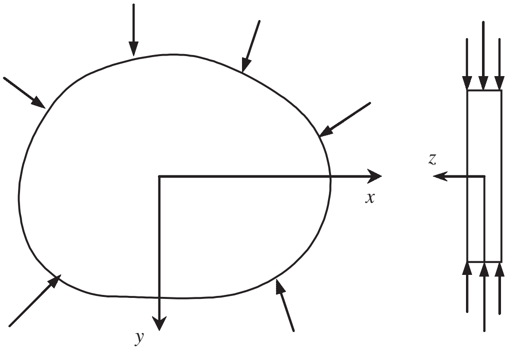


Figure 2.5. Plane stress problem. The dimension of the solid in the thickness (z) direction is much smaller than that in the *x* and *y* directions. All the forces are applied within the *x*–*y* plane, and hence the displacements are functions of *x* and *y* only.

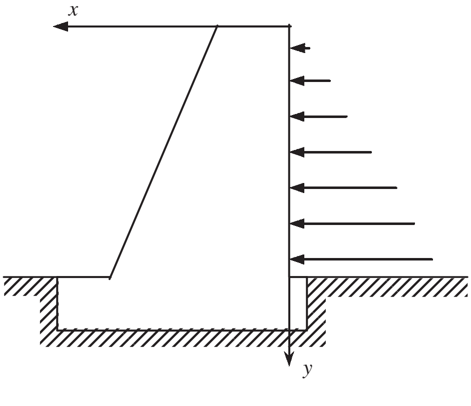


Figure 2.6. Plane strain problem. The dimension of the solid in the thickness (z) direction is much larger than that in the *x* and *y* directions, and the cross-section and the external forces do not vary in the z direction. A cross-section can then be taken as a representative cell, and hence the displacements are functions of *x* and *y* only.

Similarly, for the plane strain problems, the stresses σ_{xz} and σ_{yz} are zero, but σ_{zz} will not be zero. It can be recovered easily using Eq.(2.9) after the in-plan strains are obtained.

The system equations for 2D solids can be obtained immediately by omitting terms related to the z direction in the system equations for 3D solids. The stress components are

$$\boldsymbol{\sigma} = \begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \end{Bmatrix} \tag{2.24}$$

There are three corresponding strain components at any point in 2D solids, which can also be written in a similar vector form

$$\boldsymbol{\varepsilon} = \begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \varepsilon_{xy} \end{Bmatrix} \tag{2.25}$$

The strain-displacement relationships are

$$\varepsilon_{xx} = \frac{\partial u}{\partial x}; \quad \varepsilon_{yy} = \frac{\partial v}{\partial y}; \quad \varepsilon_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \tag{2.26}$$

where u, v are the displacement components in the x, y directions, respectively. The strain–displacement relation can also be written in the following matrix form:

$$\boldsymbol{\varepsilon} = \mathbf{LU} \tag{2.27}$$

where the displacement vector has the form of

$$\mathbf{U} = \begin{Bmatrix} u \\ v \end{Bmatrix} \tag{2.28}$$

and the differential operator matrix is obtained simply by inspection of Eq. (2.26) as

$$\mathbf{L} = \begin{bmatrix} \partial/\partial x & 0 \\ 0 & \partial/\partial y \\ \partial/\partial y & \partial/\partial x \end{bmatrix} \tag{2.29}$$

2.3.2 Constitutive Equations

Hooke's law for 2D solids has the following matrix form with $\boldsymbol{\sigma}$ and $\boldsymbol{\varepsilon}$ from Eqs. (2.24) and (2.25):

$$\boldsymbol{\sigma} = \mathbf{c}\boldsymbol{\varepsilon} \tag{2.30}$$

where $\mathbf{c}$ is a matrix of material constants, which have to be obtained through experiments. For plane stress, isotropic materials, we have

$$\mathbf{c} = \frac{E}{1-\nu^2} \begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & (1-\nu)/2 \end{bmatrix} \text{(Plane stress)} \tag{2.31}$$

To obtain the plane stress $\mathbf{c}$ matrix above, the conditions of $\sigma_{zz} = \sigma_{xz} = \sigma_{yz} = 0$ are imposed on the generalized Hooke's law for isotropic materials. For plane strain problems, $\varepsilon_{zz} = \varepsilon_{xz} = \varepsilon_{yz} = 0$ are imposed, or alternatively, replace E and ν in Eq. (2.31), respectively, with $E/(1-\nu^2)$ and $\nu/(1-\nu)$, which leads to

$$\mathbf{c} = \frac{E(1-\nu)}{(1+\nu)(1-2\nu)} \begin{bmatrix} 1 & \nu/(1-\nu) & 0 \\ \nu/(1-\nu) & 1 & 0 \\ 0 & 0 & (1-2\nu)/(2(1-\nu)) \end{bmatrix}$$

$$\text{(Plane strain)} \tag{2.32}$$

2.3.3 Dynamic Equilibrium Equations

The dynamic equilibrium equations for 2D solids can be easily obtained by removing the terms related to the z coordinate from the 3D counterparts of Eqs. (2.15)–(2.17):

$$\frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{yx}}{\partial y} + f_x = \rho \ddot{u} \tag{2.33}$$

$$\frac{\partial \sigma_{xy}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y} + f_y = \rho \ddot{v} \tag{2.34}$$

These equilibrium equations can be written in a concise matrix form of

$$\mathbf{L}^T \boldsymbol{\sigma} + \mathbf{f}_b = \rho \ddot{\mathbf{U}} \tag{2.35}$$

where $\mathbf{f}_b$ is the external force vector given by

$$\mathbf{f}_b = \begin{Bmatrix} f_x \\ f_y \end{Bmatrix} \tag{2.36}$$

For static problems, the dynamic inertia term is removed, and the equilibrium equations can be written as

$$\mathbf{L}^T \boldsymbol{\sigma} + \mathbf{f}_b = 0 \tag{2.37}$$

Equations (2.35) or (2.37) will be much easier to solve and computationally less expensive as compared with equations for the 3D solids.

2.4 EQUATIONS FOR TRUSS MEMBERS

A typical truss structure is shown in Figure 2.7. Each truss member in a truss structure is a solid whose dimension in one direction is much larger than in the other two directions as shown in Figure 2.8. The force is applied only in the x direction. Therefore a truss member is actually a one-dimensional (1D) solid. The equations for 1D solids can be obtained by further omitting the stress related to the y direction, σ_{yy}, σ_{xy}, from the 2D case.

Figure 2.7. A typical structure made up of truss members. The entrance of the faculty of Engineering, National University of Singapore.

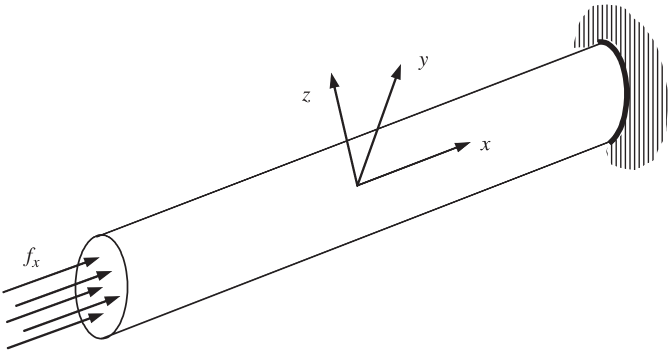


Figure 2.8. Truss member. The cross-sectional dimension of the solid is much smaller than that in the axial (*x*) directions, and the external forces are applied in the *x* direction, and hence the axial displacement is a function of *x* only.

2.4.1 Stress and Strain

Omitting the stress terms in the y direction, the stress in a truss member is only σ_{xx}, which is often simplified as σ_x. The corresponding strain in a truss member is ε_{xx}, which is simplified as ε_x. The strain–displacement relationship is simply given by

$$\varepsilon_x = \frac{\partial u}{\partial x} \tag{2.38}$$

2.4.2 Constitutive Equations

Hooke's law for 1D solids has the following simple form, with the exclusion of the y dimension and hence the Poisson effect:

$$\sigma = E\varepsilon \tag{2.39}$$

This is actually the original Hooke's law in one dimension. The Young's module E can be obtained using a simple tensile test.

2.4.3 Dynamic Equilibrium Equations

By eliminating the y dimension term from Eq. (2.33), for example, the dynamic equilibrium equation for 1D solids is

$$\frac{\partial \sigma_x}{\partial x} + f_x = \rho \ddot{u} \tag{2.40}$$

Substituting Eqs. (2.38) and (2.39) into Eq. (2.40), we obtain the governing equation for elastic and homogenous (E is independent of x) trusses as follows:

$$E\frac{\partial^2 u}{\partial x^2} + f_x = \rho \ddot{u} \tag{2.41}$$

The static equilibrium equation for trusses is obtained by eliminating the inertia term in Eq. (2.40):

$$\frac{\partial \sigma_x}{\partial x} + f_x = 0 \tag{2.42}$$

The static equilibrium equation in terms of displacement for elastic and homogenous trusses is obtained by eliminating the inertia term in Eq. (2.41):

$$E\frac{\partial^2 u}{\partial x^2} + f_x = 0 \tag{2.43}$$

For bars of constant cross-sectional area A, the above equation can be written as

$$EA\frac{\partial^2 u}{\partial x^2} + F_x = 0$$

where $F_x = f_x A$ is the external force applied in the axial direction of the bar.

2.5 EQUATIONS FOR BEAMS

A beam possesses geometrically similar dimensional characteristics as a truss member, as shown in Figure 2.9. The difference is that the forces applied on beams are transversal, meaning the direction of the force is perpendicular to the axis of the beam. Therefore, a beam experiences bending, which is the deflection in the y direction as a function of x.

2.5.1 Stress and Strain

The stresses on the cross-section of a beam are the normal stress, σ_{xz}, and shear stress, σ_{xz}. There are several theories for analysing beam deflections. These theories can be basically

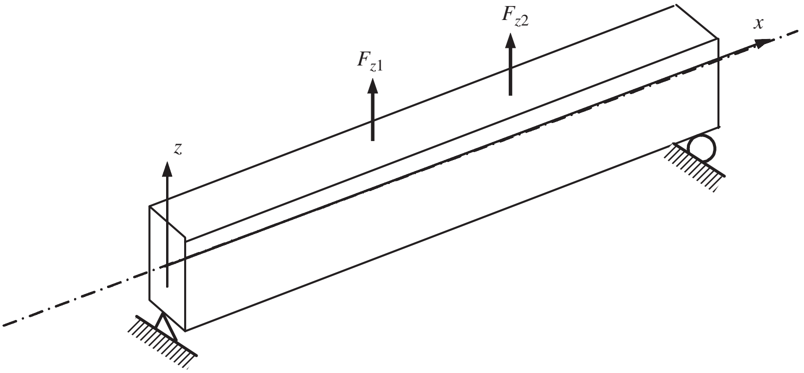


Figure 2.9. Simply supported beam. The cross-sectional dimensions of the solid are much smaller than in the axial (x) directions, and the external forces are applied in the transverse (z) direction, hence the deflection of the beam is a function of x only.

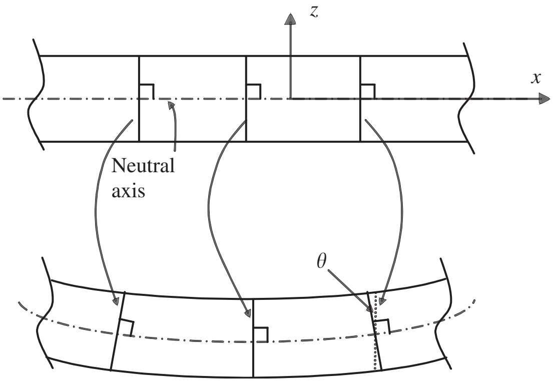


Figure 2.10. *Euler–Bernoulli* assumption for thin beams. The plane cross-sections that are normal to the undeformed, centroidal axis, remain plane and normal to the deformed axis after bending deformation. We hence have $u = -z\theta$.

divided into two major categories: a theory for *thin* beams and a theory for *thick* beams. This book focuses on the thin beam theory, which is often referred to as the *Euler–Bernoulli* beam theory. The Euler–Bernoulli beam theory assumes that the plane cross-sections, which are normal to the undeformed, centroidal axis, remain plane after bending and remain normal to the deformed axis, as shown in Figure 2.10. With this assumption, one can first have

$$\varepsilon_{xz} = 0 \tag{2.44}$$

which simply means that the shear stress is assumed to be negligible. Secondly, the axial displacement, u, at a distance z from the centroidal axis can be expressed by

$$u = -z\theta \tag{2.45}$$

where θ is the rotation in the x–z plane. The rotation can be obtained from the deflection of the centroidal axis of the beam, w, in the z direction:

$$\theta = \frac{\partial w}{\partial x} \tag{2.46}$$

The relationship between the normal strain and the deflection can be given by

$$\varepsilon_{xx} = \frac{\partial u}{\partial x} = -z\frac{\partial^2 w}{\partial x^2} = -zLw \tag{2.47}$$

where L is the differential operator given by

$$L = \frac{\partial^2}{\partial x^2} \tag{2.48}$$

2.5.2 Constitutive Equations

Similar to the equation for truss members, the original Hooke's law is applicable for beams:

$$\sigma_{xx} = E\varepsilon_{xx} \tag{2.49}$$

2.5.3 Moments and Shear Forces

Because the loading on the beam is in the transverse direction, there will be moments and corresponding shear forces imposed on the cross-sectional plane of the beam. On the other hand, bending of the beam can also be achieved if pure moments are applied instead of transverse loading. Figure 2.11 shows a small representative cell of length dx of the beam. The beam cell is subjected to external force, f_z, moment, M, shear force, Q, and inertial force, $\rho A\ddot{w}$, where ρ is the density of the material and A is the area of the cross-section.

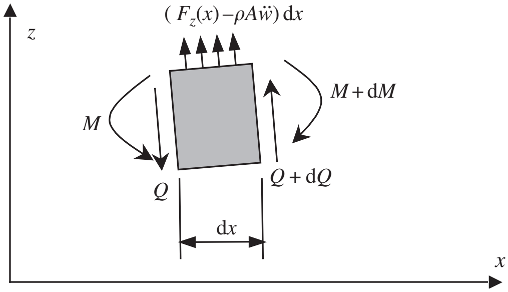


Figure 2.11. Isolated beam cell of length dx. Moments and shear forces are obtained by integration of stresses over the cross-section of the beam.

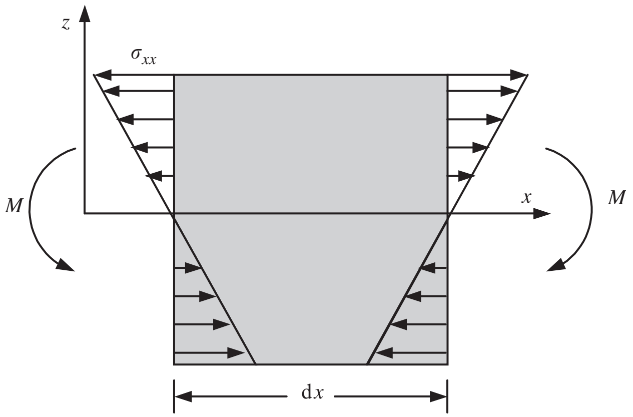


Figure 2.12. Normal stress that results in moment.

The moment on the cross-section at x results from the distributed normal stress σ_{xx}, as shown in Figure 2.12. The normal stress can be calculated by substituting Eq. (2.47) into Eq. (2.49):

$$\sigma_{xx} = -zELw \tag{2.50}$$

It can be seen from the above equation that the normal stress σ_{xx} varies linearly in the vertical direction on the cross-section of the beam. The moments resulting from the normal stress on the cross-section can be calculated by the following integration over the area of the cross-section:

$$M = \int_A \sigma_{xx} z \, \mathrm{d}A = -E\left(\int_A z^2 \, \mathrm{d}A\right) Lw = -EILw = -EI\frac{\partial^2 w}{\partial x^2} \tag{2.51}$$

where I is the second moment of area (or moment of inertia) of the cross-section with respect to the y-axis, which can be calculated for a given shape of the cross-section using the following equation:

$$I = \int_A z^2 \, \mathrm{d}A \tag{2.52}$$

We now consider the force equilibrium of the small beam cell in the z direction

$$\mathrm{d}Q + (F_z(x) - \rho A\ddot{w}) \, \mathrm{d}x = 0 \tag{2.53}$$

or

$$\frac{\mathrm{d}Q}{\mathrm{d}x} = -F_z(x) + \rho A\ddot{w} \tag{2.54}$$

We would also need to consider the moment equilibrium of the small beam cell with respect to any point at the right surface of the cell,

$$\mathrm{d}M - Q \, \mathrm{d}x + \tfrac{1}{2}(F_z - \rho \mathrm{A}\ddot{w}) \, (\mathrm{d}x)^2 = 0 \tag{2.55}$$

Neglecting the second order small term containing $(\mathrm{d}x)^2$ leads to

$$\frac{\mathrm{d}M}{\mathrm{d}x} = Q \tag{2.56}$$

And finally, substituting Eq. (2.51) into Eq. (2.56) gives

$$Q = -EI\frac{\partial^3 w}{\partial x^3} \tag{2.57}$$

Equations (2.56) and (2.57) give the relationship between the moments and shear forces in a beam with the deflection of the Euler–Bernoulli beam.

2.5.4 Dynamic Equilibrium Equations

The dynamic equilibrium equation for beams can be obtained simply by substituting Eq. (2.57) into Eq. (2.54):

$$EI\,\frac{\partial^4 w}{\partial x^4} + \rho A\ddot{w} = F_z \tag{2.58}$$

The static equilibrium equation for beams can be obtained similarly by dropping the dynamic term in Eq. (2.58):

$$EI\,\frac{\partial^4 w}{\partial x^4} = F_z \tag{2.59}$$

2.6 EQUATIONS FOR PLATES

The wings of an aircraft can sometimes be considered as a plate structure carrying transverse loads in the form of engines or other components, as shown in Figure 2.13. A plate possesses a geometrically similar dimensional characteristic as a 2D solid, as shown in Figure 2.14. The difference is that the forces applied on a plate are in the direction perpendicular to the plane of the plate. A plate can also be viewed as a 2D analogy of a beam. Therefore, a plate experiences bending resulting in deflection w in the z direction, which is a function of x and y.

2.6.1 Stress and Strain

The stress σ_{zz} in a plate is assumed to be zero. Similar to beams, there are several theories for analysing deflection in plates. These theories can also be basically divided into two major categories: a theory for *thin* plates and a theory for *thick* plates. This chapter addresses thin plate theory, often called *Classical* Plate Theory (CPT), or the *Kirchhoff* plate theory, as well as the *first order shear deformation theory* for thick plates known as the *Reissner–Mindlin* plate theory (Reissner, 1945; Mindin, 1951). However, pure plate elements are usually not available in most commercial finite element packages, since most people would use the more

Figure 2.13. An aircraft wing can be idealized as a plate structure.

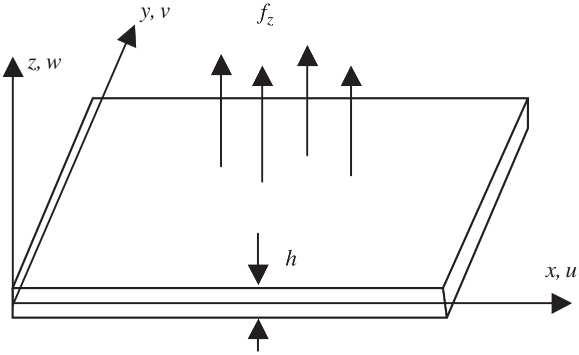


Figure 2.14. A plate subjected to transverse load that results in bending deformation.

general shell elements, which will be discussed in later chapters. Furthermore, it is found that the CPT is not very practical for many situations, since shear deformation should not be neglected for most structures. Hence, in this book, only equations of plate elements based on the Reissner–Mindlin plate theory will be formulated (Chapter 8). Nevertheless, the CPT will also be briefed in this chapter for completeness of this introduction to mechanics for solids and structures.

The CPT assumes that normals to the middle (neutral) plane of the undeformed plate remain straight and normal to the middle plane during deformation or bending. This assumption results in

$$\varepsilon_{xz} = 0, \quad \varepsilon_{yz} = 0 \tag{2.60}$$

Secondly, the displacements parallel to the undeformed middle plane, u and v, at a distance z from the centroidal axis can be expressed by

$$u = -z\frac{\partial w}{\partial x} \tag{2.61}$$

$$v = -z\frac{\partial w}{\partial y} \tag{2.62}$$

where w is the deflection of the middle plane of the plate in the z direction. The relationship between the components of strain and the deflection can be given by

$$\varepsilon_{xx} = \frac{\partial u}{\partial x} = -z\frac{\partial^2 w}{\partial x^2} \tag{2.63}$$

$$\varepsilon_{yy} = \frac{\partial v}{\partial y} = -z\frac{\partial^2 w}{\partial y^2} \tag{2.64}$$

$$\varepsilon_{xy} = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} = -2z\frac{\partial^2 w}{\partial x \partial y} \tag{2.65}$$

or in the matrix form

$$\boldsymbol{\varepsilon} = -z\mathbf{L}w \tag{2.66}$$

where $\boldsymbol{\varepsilon}$ is the vector of in-plane strains defined by Eq. (2.25), and $\mathbf{L}$ is the differential operator matrix given by

$$\mathbf{L} = \begin{bmatrix} \partial^2/\partial x^2 \\ \partial^2/\partial y^2 \\ 2\partial^2/\partial x\partial y \end{bmatrix} \tag{2.67}$$

2.6.2 Constitutive Equations

The original Hooke's law is applicable for plates:

$$\boldsymbol{\sigma} = \mathbf{c}\boldsymbol{\varepsilon} \tag{2.68}$$

where $\mathbf{c}$ has the same form for 2D solids defined by Eq. (2.31), the plane stress case, since σ_{zz} is assumed to be zero.

2.6.3 Moments and Shear Forces

Figure 2.15 shows a small representative cell of dx × dy from a plate of thickness h. The plate cell is subjected to external force f_z, and inertial force $\rho h\ddot{w}$, where ρ is the density of the material. Figure 2.16 shows the moments M_x, M_y, M_z and M_{xy}, and shear forces Q_x and Q_y present. The moments and shear forces result from the distributed normal and shear stresses σ_{xx}, σ_{yy} and σ_{xy}, shown in Figure 2.15. The stresses can be obtained by substituting Eq. (2.66) into Eq. (2.68)

$$\boldsymbol{\sigma} = -z\mathbf{c}\mathbf{L}w \tag{2.69}$$

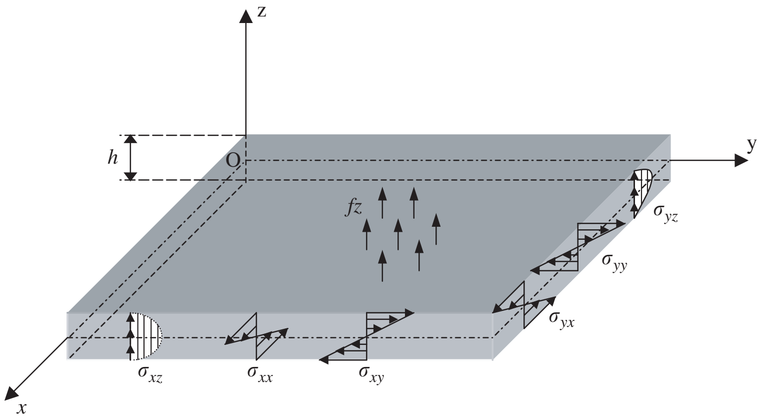


Figure 2.15. Stresses on an isolated plate cell. Integration of these stresses results in corresponding moments and shear forces.

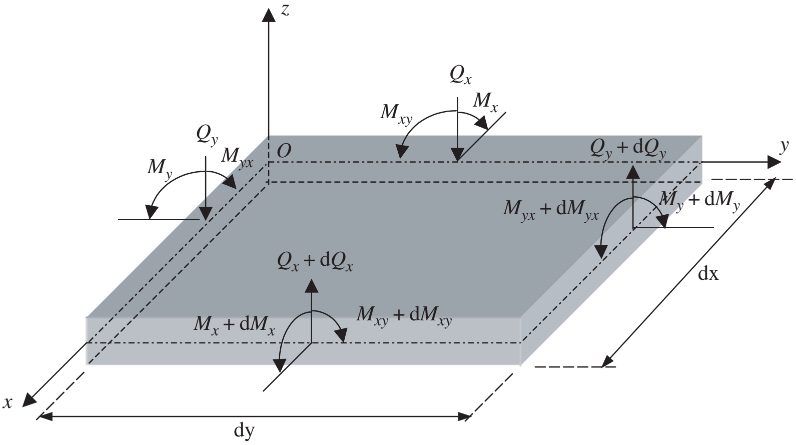


Figure 2.16. Shear forces and moments on an isolated plate cell of dx × dy. The equilibrium system equations are established based on this state of forces and moments.

It can be seen from the above equation that the normal stresses vary linearly in the vertical direction on the cross-sections of the plate. The moments on the cross-section can be calculated in a similar way as for beams by the following integration:

$$\mathbf{M}_p = \begin{Bmatrix} M_x \\ M_y \\ M_{xy} \end{Bmatrix} = \int_A \boldsymbol{\sigma} z \, \mathrm{d}A = -\mathbf{c}\left(\int_A z^2 \, \mathrm{d}A\right)\mathbf{L}w = -\frac{h^3}{12}\mathbf{c}\mathbf{L}w \tag{2.70}$$

Consider first the equilibrium of the small plate cell in the z direction, and note that $\mathrm{d}Q_x = (\partial Q_x/\partial x)\,\mathrm{d}x$ and $\mathrm{d}Q_y = (\partial Q_y/\partial y)\,\mathrm{d}y$ we have

$$\left(\frac{\partial Q_x}{\partial x}\,\mathrm{d}x\right)\mathrm{d}y + \left(\frac{\partial Q_y}{\partial y}\,\mathrm{d}y\right)\mathrm{d}x + (f_z - \rho h\ddot{w})\,\mathrm{d}x\,\mathrm{d}y = 0 \tag{2.71}$$

or

$$\frac{\partial Q_x}{\partial x} + \frac{\partial Q_y}{\partial y} + f_z = \rho h\ddot{w} \tag{2.72}$$

Consider then the moment equilibrium of the plate cell with respect to the x-axis, and neglecting the second order small term, leads to a formula for shear force Q_x:

$$Q_x = \frac{\partial M_x}{\partial x} + \frac{\partial M_{xy}}{\partial y} \tag{2.73}$$

Finally, consider the moment equilibrium of the plate cell with respect to the y-axis, and neglecting the second order small term, gives

$$Q_y = \frac{\partial M_{xy}}{\partial x} + \frac{\partial M_y}{\partial y} \tag{2.74}$$

in which we implied that $M_{yx} = M_{xy}$.

2.6.4 Dynamic Equilibrium Equations

To obtain the dynamic equilibrium equation for plates, we first substitute Eq. (2.70) into Eqs. (2.73) and (2.74), after which Q_x and Q_y are substituted into Eq. (2.72):

$$D\left(\frac{\partial^4 w}{\partial x^4} + 2\frac{\partial^4 w}{\partial x^2 \partial y^2} + \frac{\partial^4 w}{\partial y^4}\right) + \rho h\ddot{w} = f_z \tag{2.75}$$

where $D = Eh^3/(12(1-v^2))$ is the bending stiffness of the plate. The static equilibrium equation for plates can again be obtained by dropping the dynamic term in Eq. (2.75):

$$D\left(\frac{\partial^4 w}{\partial x^4} + 2\frac{\partial^4 w}{\partial x^2 \partial y^2} + \frac{\partial^4 w}{\partial y^4}\right) = f_z \tag{2.76}$$

2.6.5 Reissner–Mindlin Plate (Reissner, 1945; Mindlin, 1951)

The Reissner–Mindlin plate theory is applied for thick plates, where the shear deformation and rotary inertia effects are included. The Reissner–Mindlin theory does not require the cross-section to be perpendicular to the axial axes after deformation, as shown in Figure 2.17. Therefore, $\varepsilon_{xz} \neq 0$ and $\varepsilon_{yz} \neq 0$. The displacements parallel to the undeformed middle surface, u and v, at a distance z from the centroidal axis can be expressed by

$$u = z\theta_y \tag{2.77}$$

$$v = -z\theta_x \tag{2.78}$$

where θ_x and θ_y are, respectively, the rotations about the x and y axes of lines normal to the middle plane before deformation.

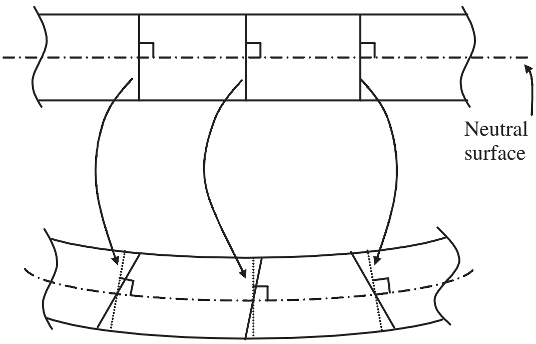


Figure 2.17. Shear deformation in a Mindlin plate. The rotations of the cross-sections are treated as independent variables.

The in-plane strains defined by Eq. (2.25) are given by

$$\varepsilon = -z\mathbf{L}\theta \tag{2.79}$$

where

$$\mathbf{L} = \begin{bmatrix} -\partial/\partial x & 0 \\ 0 & \partial/\partial y \\ -\partial/\partial y & \partial/\partial x \end{bmatrix} \tag{2.80}$$

$$\theta = \begin{Bmatrix} \theta_y \\ \theta_x \end{Bmatrix} \tag{2.81}$$

Using Eq. (2.4), the transverse shear strains ε_{xz} and ε_{yz} can be obtained as

$$\gamma = \begin{Bmatrix} \varepsilon_{xz} \\ \varepsilon_{yz} \end{Bmatrix} = \begin{Bmatrix} \theta_y + \partial w/\partial x \\ -\theta_x + \partial w/\partial y \end{Bmatrix} \tag{2.82}$$

Note that if the transverse shear strains are negligible, the above equation will lead to

$$\theta_x = \frac{\partial w}{\partial y} \tag{2.83}$$

$$\theta_y = -\frac{\partial w}{\partial x} \tag{2.84}$$

and Eq. (2.79) becomes Eq. (2.66) of the CPT. The transverse average shear stress τ relates to the transverse shear strain in the form

$$\tau = \begin{Bmatrix} \tau_{xz} \\ \tau_{yz} \end{Bmatrix} = \kappa \begin{bmatrix} G & 0 \\ 0 & G \end{bmatrix} \gamma = \kappa[\mathbf{D}_s]\gamma \tag{2.85}$$

where G is the shear modulus, and κ is a constant that is usually taken to be $\pi^2/12$ or 5/6.

The equilibrium equations for a Reissner–Mindlin plate can also be similarly obtained as that of a thin plate. Equilibrium of forces and moments can be carried out but this time taking into account the transverse shear stress and rotary inertia. For the purpose of this book, the above concepts for the Reissner-Mindlin plate will be sufficient and the equilibrium equations will not be shown here. Chapter 8 will detail the derivation of the discrete finite element equations by using energy principles.

2.7 REMARKS

Having shown how the equilibrium equations for various types of geometrical structures are obtained, it can be noted that all the equilibrium equations are just special cases of the general equilibrium equation for 3D solids. The use of proper assumptions and theories can lead to a dimension reduction, and hence simplify the problem. These kinds of simplification can significantly reduce the size of finite element models.

3

FUNDAMENTALS FOR FINITE ELEMENT METHOD

3.1 INTRODUCTION

As mentioned in Chapter 1, when using the Finite Element Method (FEM) to solve mechanics problems governed by a set of partial differential equations, the problem domain is first discretized into small elements. In each of these elements, the profile of the displacements is assumed in simple forms to obtain element equations. The equations obtained for each element are then assembled together with adjoining elements to form the global finite element equation for the whole problem domain. Equations thus created for the global problem domain can be solved easily for the entire displacement field.

The above-mentioned FEM process does not seem to be a difficult task. However, in close examination of the above-mentioned process, one would naturally ask a series of questions. How can one simply assume the profile of the solution of the displacement in any simple form? How can one ensure that the governing partial differential equations will be satisfied by the assumed displacement? What should one do when using the assumed profile of the displacement to determine the final displacement field? Yes, one can just simply assume the profile of the solution of the displacements, but a principle has to be followed in order to obtain discretized system equations that can be solved routinely for the final displacement field. The use of such a principle guarantees the *best* (not exact) satisfaction of the governing system equation under certain conditions. The following details one of the most important principles, which will be employed to establish the FEM equations for mechanics problems of solids and structures.

It may be common for a beginner to be daunted by the equations involved when formulating the finite element equations. Perhaps for a beginner, full understanding of the details of the equations in this chapter may prove to be a challenging task. It is thus advised that the novice reader just understands the basic concepts involved without digging too much into the equations. It is then recommended to review this chapter again after going through subsequent chapters with examples to fully understand the equations. Advanced readers are referred to the FEM handbook (Kardestuncer, 1987) for more complete topics related to FEM.

3.2 STRONG AND WEAK FORMS

The partial differential system equations developed in Chapter 2, such as Eqs. (2.20) and (2.21), are *strong forms* of the governing system of equations for solids. The strong form, in contrast to a *weak form*, requires strong continuity on the dependent field variables (the displacements u, v and w in this case). Whatever functions that define these field variables have to be differentiable up to the order of the partial differential equations that exist in the strong form of the system equations. Obtaining the exact solution for a strong form of the system equation is usually very difficult for practical engineering problems. The finite difference method can be used to solve system equations of the strong form to obtain an approximated solution. However, the method usually works well for problems with simple and regular geometry and boundary conditions.

A weak form of the system equations is usually created using one of the following widely used methods:

- Energy principles (see, e.g. Washizu, 1974; Reddy, 1984)
- Weighted residual methods (see, e.g. Crandall, 1956; Finlayson and Scriven, 1966; Finlayson, 1972; Ziekiewicz and Taylor, 2000)

The energy principle can be categorized as a special form of the variational principle which is particularly suited for problems of the mechanics of solids and structures. The weighted residual method is a more general mathematical tool applicable, in principle, for solving all kinds of partial differential equations. Both methods are very easy to understand and apply. This book will demonstrate both methods used for creating FEM equations. An energy principle will be used for mechanics problems of solids and structures, and the weighted residual method will be used for formulating the heat transfer problems. It is also equally applicable to use the energy principle for heat transfer problems, and the weighted residual method for solid mechanics problems, and the procedure is very much the same.

The weak form is often an integral form and requires a weaker continuity on the field variables. Due to the weaker requirement on the field variables, and the integral operation, a formulation based on a weak form usually produces a set of discretized system equations that give much more accurate results, especially for problems of complex geometry. Hence, the weak form is preferred by many for obtaining an approximate solution. The finite element method is a typical example of successfully using weak form formulations. Using the weak form usually leads to a set of well-behaved algebraic system equations, if the problem domain is discretized properly into elements. As the problem domain can be discretized into different types of elements, the FEM can be applied for many practical engineering problems with most kinds of complex geometry and boundary conditions.

In the following section, Hamilton's principle, which is one of the most powerful energy principles, is introduced for FEM formulation of problems of mechanics of solids and structures. Hamilton's principle is chosen because it is simple and can be used for dynamic problems. The approach adopted in this book is to directly work out the dynamic system equations, after which the static system equations can be easily obtained by simply dropping out the dynamic terms. This can be done because of the simple fact that the dynamic system equations are the general system equations, and the static case can be considered to be just a special case of the dynamic equations.

3.3 HAMILTON'S PRINCIPLE

Hamilton's principle is a simple yet powerful tool that can be used to derive discretized dynamic system equations. It states simply that

> "Of all the *admissible* time histories of displacement the most accurate solution makes the Lagrangian functional a minimum."

An *admissible* displacement must satisfy the following conditions:

(a) the compatibility equations,
(b) the essential or the kinematic boundary conditions, and
(c) the conditions at initial (t_1) and final time (t_2).

Condition (a) ensures that the displacements are compatible (continuous) in the problem domain. As will be seen in Chapter 11, there are situations when incompatibility can occur at the edges between elements. Condition (b) ensures that the displacement constraints are satisfied; and condition (c) requires the displacement history to satisfy the constraints at the initial and final times.

Mathematically, Hamilton's principle states:

$$\delta \int_{t_1}^{t_2} L \, \mathrm{d}t = 0 \tag{3.1}$$

The Langrangian functional, L, is obtained using a set of admissible time histories of displacements, and it consists of

$$L = T - \Pi + W_f \tag{3.2}$$

where T is the kinetic energy, Π is the potential energy (for our purposes, it is the elastic strain energy), and W_f is the work done by the external forces. The kinetic energy of the entire problem domain is defined in the integral form

$$T = \frac{1}{2} \int_V \rho \dot{\mathbf{U}}^T \dot{\mathbf{U}} \, \mathrm{d}V \tag{3.3}$$

where V represents the whole volume of the solid, and $\mathbf{U}$ is the set of admissible time histories of displacements.

The strain energy in the entire domain of elastic solids and structures can be expressed as

$$\Pi = \frac{1}{2} \int_V \boldsymbol{\varepsilon}^T \boldsymbol{\sigma} \, \mathrm{d}V = \frac{1}{2} \int_V \boldsymbol{\varepsilon}^T \mathbf{c} \boldsymbol{\varepsilon} \, \mathrm{d}V \tag{3.4}$$

where $\boldsymbol{\varepsilon}$ are the strains obtained using the set of admissible time histories of displacements. The work done by the external forces over the set of admissible time histories of displacements can be obtained by

$$W_f = \int_V \mathbf{U}^T \mathbf{f}_b \, \mathrm{d}V + \int_{S_f} \mathbf{U}^T \mathbf{f}_s \, \mathrm{d}S_f \qquad (3.5)$$

where S_f represents the surface of the solid on which the surface forces are prescribed (see Figure 2.2).

Hamilton's principle allows one to simply assume any set of displacements, as long as it satisfies the three admissible conditions. The assumed set of displacements will not usually satisfy the strong form of governing system equations unless we are extremely lucky, or the problem is extremely simple and we know the exact solution. Application of Hamilton's principle will conveniently guarantee a combination of this assumed set of displacements to produce the most accurate solution for the system that is governed by the strong form of the system equations.

The power of Hamilton's principle (or any other variational principle) is that it provides the freedom of choice, opportunity and possibility. For practical engineering problems, one usually does not have to pursue the exact solution, which in most cases are usually unobtainable, because we now have a choice to quite conveniently obtain a good approximation using Hamilton's principle, by assuming the likely form, pattern or *shape* of the solutions. Hamilton's principle thus provides the foundation for the finite element methods. Furthermore, the simplicity of Hamilton's principle (or any other energy principle) manifests itself in the use of scalar energy quantities. Engineers and scientists like working with scalar quantities when it comes to numerical calculations, as they do not need to worry about the direction. All the mathematical tools required to derive the final discrete system equations are then basic integration, differentiation and variation, all of which are standard linear operations. Another plus point of Hamilton's principle is that the final discrete system equations produced are usually a set of linear algebraic equations that can be solved using conventional methods and standard computational routines. The following demonstrates how the finite element equations can be established using Hamilton's principle and its simple operations.

3.4 FEM PROCEDURE

The standard FEM procedure can be briefly summarized as follows.

3.4.1 Domain Discretization

The solid body is divided into N_e elements. The procedure is often called *meshing*, which is usually performed using so-called pre-processors. This is especially true for complex geometries. Figure 3.1 shows an example of a mesh for a two-dimensional solid.

The pre-processor generates unique numbers for all the elements and nodes for the solid or structure in a proper manner. An element is formed by connecting its nodes in a pre-defined consistent fashion to create the *connectivity* of the element. All the elements together form the entire domain of the problem without any gap or overlapping. It is possible for the domain to consist of different types of elements with different numbers of nodes, as

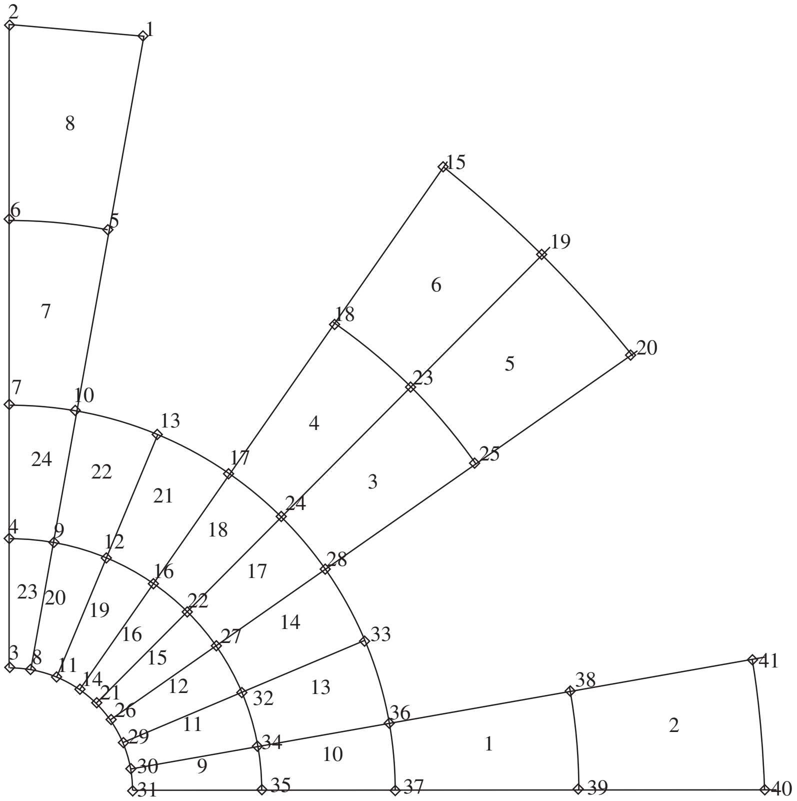


Figure 3.1. Example of a mesh with elements and node properly numbered.

long as they are *compatible* (no gaps and overlapping; the admissible condition (a) required by Hamilton's principle) on the boundaries between different elements. The density of the mesh depends upon the accuracy requirement of the analysis and the computational resources available. Generally, a finer mesh will yield results that are more accurate, but will increase the computational cost. As such, the mesh is usually not uniform, with a finer mesh being used in the areas where the displacement gradient is larger or where the accuracy is critical to the analysis. The purpose of the domain discretization is to make it easier in assuming the pattern of the displacement field.

3.4.2 Displacement Interpolation

The FEM formulation has to be based on a coordinate system. In formulating FEM equations for elements, it is often convenient to use a *local coordinate system* that is defined for an element in reference to the global coordination system that is usually defined for the entire structure, as shown in Figure 3.4. Based on the *local coordinate system* defined on an element, the displacement within the element is now assumed simply by polynomial interpolation using the displacements at its nodes (or *nodal displacements*) as

$$\mathbf{U}^h(x,y,z)=\sum_{i=1}^{n_d}\mathbf{N}_i(x,y,z)\mathbf{d}_i=\mathbf{N}(x,y,z)\mathbf{d}_e \tag{3.6}$$

where the superscript h stands for approximation, n_d is the number of nodes forming the element, and $\mathbf{d}_i$ is the nodal displacement at the ith node, which is the unknown the analyst wants to compute, and can be expressed in a general form of

$$\mathbf{d}_i=\begin{Bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{nf} \end{Bmatrix} \begin{matrix} \rightarrow \text{displacement component 1} \\ \rightarrow \text{displacement component 2} \\ \vdots \\ \rightarrow \text{displacement component } nf \end{matrix} \tag{3.7}$$

where nf is the number of Degrees Of Freedom (DOF) at a node. For 3D solids, $nf=3$, and

$$\mathbf{d}_i=\begin{Bmatrix} u_i \\ v_i \\ w_i \end{Bmatrix} \begin{matrix} \rightarrow \text{displacement in the } x\text{-direction} \\ \rightarrow \text{displacement in the } y\text{-direction} \\ \rightarrow \text{displacement in the } z\text{-direction} \end{matrix} \tag{3.8}$$

Note that the displacement components can also consist of rotations for structures of beams and plates. The vector $\mathbf{d}_e$ in Eq. (3.6) is the displacement vector for the entire element, and has the form of

$$\mathbf{d}_e=\begin{Bmatrix} \mathbf{d}_1 \\ \mathbf{d}_2 \\ \vdots \\ \mathbf{d}_{n_d} \end{Bmatrix} \begin{matrix} \rightarrow \text{ displacements at node 1} \\ \rightarrow \text{ displacements at node 2} \\ \vdots \qquad \vdots \\ \rightarrow \text{ displacements at node } n_d \end{matrix} \tag{3.9}$$

Therefore, the total DOF for the entire element is $n_d \times nf$.

In Eq. (3.6), $\mathbf{N}$ is a matrix of *shape functions* for the nodes in the element, which are predefined to assume the shapes of the displacement variations with respect to the coordinates. It has the general form of

$$\begin{matrix} \mathbf{N}(x,y,z)=[\mathbf{N}_1(x,y,z) & \mathbf{N}_2(x,y,z) & \cdots & \mathbf{N}_{n_d}(x,y,z)] \\ \downarrow & \downarrow & \cdots & \downarrow \\ \text{for node 1} & \text{for node 2} & \cdots & \text{for node } n_d \end{matrix} \tag{3.10}$$

where $\mathbf{N}_i$ is a sub-matrix of shape functions for displacement components, which is arranged as

$$\mathbf{N}_i=\begin{bmatrix} N_{i1} & 0 & 0 & 0 \\ 0 & N_{i2} & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & N_{inf} \end{bmatrix} \tag{3.11}$$

where N_{ik} is the shape function for the kth displacement component (DOF) at the ith node. For 3D solids, $nf=3$, and often $N_{i1}=N_{i2}=N_{i3}=N_i$. Note that it is not necessary to use the same shape function for all the displacement components at a node. For example, we often use different shape functions for translational and rotational displacements.

Note that this approach of assuming the displacements is often called the *displacement method*. There are FEM approaches that assume the stresses instead, but they will not be covered in this book.

3.4.3 Standard Procedure for Constructing Shape Functions

Consider an element with n_d nodes at $\mathbf{x}_i$ $(i = 1, 2, \ldots, n_d)$, where $\mathbf{x}^T = \{x\}$ for one-dimensional problems, $\mathbf{x}^T = \{x, y\}$ for two-dimensional problems, and $\mathbf{x}^T = \{x, y, z\}$ for three-dimensional problems. We should have n_d shape functions for each displacement component for an element. In the following, we consider only one displacement component in the explanation of the standard procedure for constructing the shape functions. The standard procedure is applicable for any other displacement components. First, the displacement component is approximated in the form of a linear combination of n_d linearly-independent basis functions $p_i(\mathbf{x})$, i.e.

$$u^h(\mathbf{x}) = \sum_{i=1}^{n_d} p_i(\mathbf{x})\alpha_i = \mathbf{p}^T(\mathbf{x})\boldsymbol{\alpha} \tag{3.12}$$

where u^h is the approximation of the displacement component, $p_i(\mathbf{x})$ is the basis function of monomials in the space coordinates $\mathbf{x}$, and α_i is the coefficient for the monomial $p_i(\mathbf{x})$. Vector α is defined as

$$\boldsymbol{\alpha}^T = \{\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_{n_d}\} \tag{3.13}$$

The $p_i(\mathbf{x})$ in Eq. (3.12) is built with n_d terms of one-dimensional monomials; based on the Pascal's triangle shown in Figure 3.2 for two-dimensional problems; or the well-known Pascal's pyramid shown in Figure 3.3 for three-dimensional problems. A basis of complete order of p in the one-dimensional domain has the form

$$\mathbf{p}^T(x) = \{1, x, x^2, x^3, x^4, \ldots, x^p\} \tag{3.14}$$

A basis of complete order of p in the two-dimensional domain is provided by

$$\mathbf{p}^T(\mathbf{x}) = \mathbf{p}^T(x, y) = \{1, x, y, xy, x^2, y^2, \ldots, x^p, y^p\} \tag{3.15}$$

and that in three-dimensional domain can be written as

$$\mathbf{p}^T(\mathbf{x}) = \mathbf{p}^T(x, y, z) = \{1, x, y, z, xy, yz, zx, x^2, y^2, z^2, \ldots, x^p, y^p, z^p\} \tag{3.16}$$

As a general rule, the n_d terms of $p_i(\mathbf{x})$ used in the basis should be selected from the constant term to higher orders symmetrically from the Pascal triangle shown in Figures 3.2 or 3.3. Some higher-order terms can be selectively included in the polynomial basis if there is a need in specific circumstances.

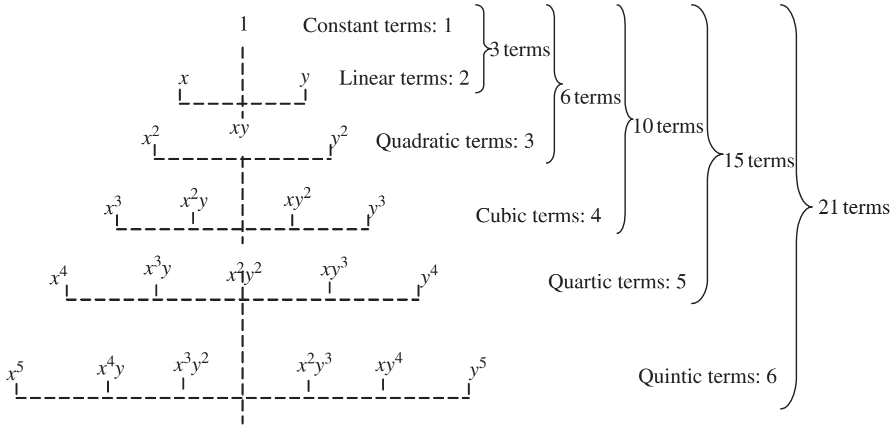


Figure 3.2. Pascal triangle of monomials (two-dimensional case).

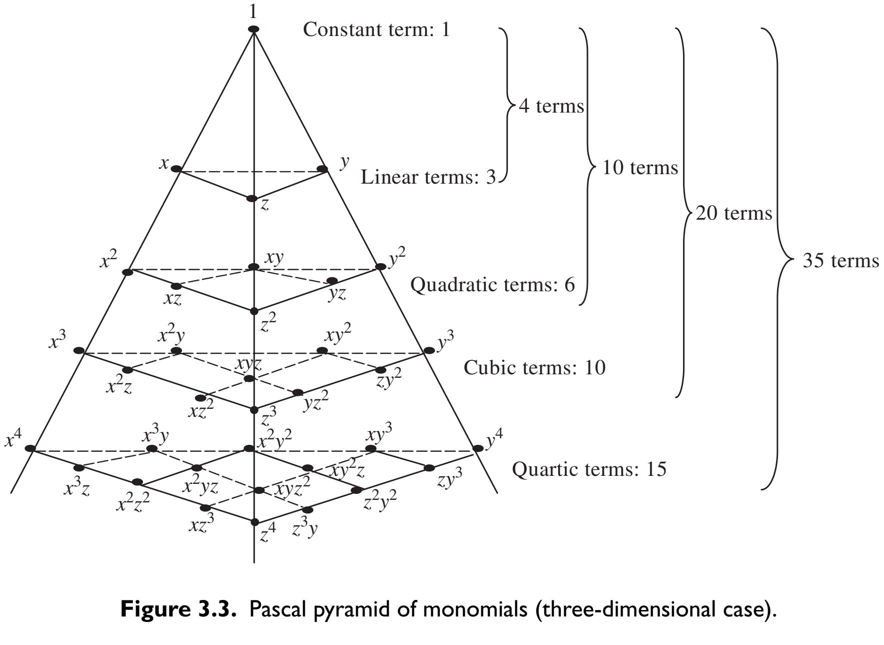


Figure 3.3. Pascal pyramid of monomials (three-dimensional case).

The coefficients α_i in Eq. (3.12) can be determined by enforcing the displacements calculated using Eq. (3.12) to be equal to the nodal displacements at the n_d nodes of the element. At node i we can have

$$d_i = \mathbf{p}^T(\mathbf{x}_i)\boldsymbol{\alpha} \quad i = 1, 2, 3, \ldots, n_d \tag{3.17}$$

where d_i is the nodal value of u^h at $\mathbf{x} = \mathbf{x}_i$. Equation (3.17) can be written in the following matrix form:

$$\mathbf{d}_e = \mathbf{P}\boldsymbol{\alpha} \tag{3.18}$$

where $\mathbf{d}_e$ is the vector that includes the values of the displacement component at all the n_d nodes in the element:

$$\mathbf{d}_e = \begin{Bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{n_d} \end{Bmatrix} \tag{3.19}$$

and $\mathbf{P}$ is given by

$$\mathbf{P} = \begin{bmatrix} \mathbf{p}^T(\mathbf{x}_1) \\ \mathbf{p}^T(\mathbf{x}_2) \\ \vdots \\ \mathbf{p}^T(\mathbf{x}_{n_d}) \end{bmatrix} \tag{3.20}$$

which is called the *moment matrix*. The expanded form of $\mathbf{P}$ is

$$\mathbf{P} = \begin{bmatrix} p_1(\mathbf{x}_1) & p_2(\mathbf{x}_1) & \cdots & p_{n_d}(\mathbf{x}_1) \\ p_1(\mathbf{x}_2) & p_2(\mathbf{x}_2) & \cdots & p_{n_d}(\mathbf{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ p_1(\mathbf{x}_{n_d}) & p_2(\mathbf{x}_{n_d}) & \cdots & p_{n_d}(\mathbf{x}_{n_d}) \end{bmatrix} \tag{3.21}$$

For two-dimensional polynomial basis functions, we have

$$\mathbf{P} = \begin{bmatrix} 1 & x_1 & y_1 & x_1y_1 & x_1^2 & y_1^2 & x_1^2y_1 & x_1y_1^2 & x_1^3 & \cdots \\ 1 & x_2 & y_2 & x_2y_2 & x_2^2 & y_2^2 & x_2^2y_2 & x_2y_2^2 & x_2^3 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n_d} & y_{n_d} & x_{n_d}y_{n_d} & x_{n_d}^2 & y_{n_d}^2 & x_{n_d}^2y_{n_d} & x_{n_d}y_{n_d}^2 & x_{n_d}^3 & \cdots \end{bmatrix} \tag{3.22}$$

Using Eq. (3.18), and assuming that the inverse of the moment matrix $\mathbf{P}$ exists, we can then have

$$\boldsymbol{\alpha} = \mathbf{P}^{-1}\mathbf{d}_e \tag{3.23}$$

Substituting Eq. (3.23) into Eq. (3.12), we then obtain

$$u^h(\mathbf{x}) = \sum_{i=1}^{n_d} N_i(\mathbf{x})d_i \tag{3.24}$$

or in matrix form

$$u^h(\mathbf{x}) = \mathbf{N}(\mathbf{x})\mathbf{d}_e \tag{3.25}$$

where $\mathbf{N}(\mathbf{x})$ is a matrix of shape functions $N_i(\mathbf{x})$ defined by

$$\begin{aligned}\mathbf{N}(\mathbf{x}) = \mathbf{p}^T(\mathbf{x})\mathbf{P}^{-1} &= \begin{bmatrix} \underbrace{\mathbf{p}^T(\mathbf{x})\mathbf{P}_1^{-1}}_{N_1(\mathbf{x})} & \underbrace{\mathbf{p}^T(\mathbf{x})\mathbf{P}_2^{-1}}_{N_2(\mathbf{x})} & \cdots & \underbrace{\mathbf{p}^T(\mathbf{x})\mathbf{P}_n^{-1}}_{N_n(\mathbf{x})} \end{bmatrix} \\ &= \begin{bmatrix} N_1(\mathbf{x}) & N_2(\mathbf{x}) & \cdots & N_n(\mathbf{x}) \end{bmatrix} \end{aligned} \tag{3.26}$$

where $\mathbf{P}_i^{-1}$ is the ith column of matrix $\mathbf{P}^{-1}$, and

$$N_i(\mathbf{x}) = \mathbf{p}^T(\mathbf{x})\mathbf{P}_i^{-1} \tag{3.27}$$

In obtaining Eq. (3.23), we have assumed the existence of the inverse of $\mathbf{P}$. There could be cases where $\mathbf{P}^{-1}$ does not exist, and the construction of shape functions will fail. The existence of $\mathbf{P}^{-1}$ depends upon (1) the basis function used, and (2) the nodal distribution of the element. The basis functions have to be chosen first from a linearly-independent set of bases, and then the inclusion of the basis terms should be based on the nodal distribution in the element. The discussion in this direction is more involved, and interested readers are referred to a monograph by Liu [2002]. In this book, we shall only discuss elements whose corresponding moment matrices are invertible.

Note that the derivatives of the shape functions can be obtained very easily, as all the functions involved are polynomials. The lth derivative of the shape functions is simply given by

$$N_i^{(l)}(\mathbf{x}) = \left[\mathbf{p}^{(l)}(\mathbf{x})\right]^T \mathbf{P}_i^{-1} \tag{3.28}$$

The issues related to the compatibility of element shape functions will be addressed in Chapter 11. Note that there are many other methods for creating shape functions which do not necessarily follow the standard procedure described above. Some of these often-used *shortcut* methods will be discussed in later chapters, when we develop different types of elements. These shortcut methods need to make use of the properties of shape functions detailed in the next section.

3.4.4 Properties of the Shape Functions[1]

Property 1. Reproduction property and consistency

The consistency of the shape function within the element depends upon the complete orders of the monomial $p_i(\mathbf{x})$ used in Eq. (3.12), and hence is also dependent upon the number of nodes of the element. If the complete order of monomial is k, the shape functions is said to

[1] The reader may skip the proof of these properties and lemmas.

possess C^k consistency. To demonstrate, we consider a field given by

$$f(\mathbf{x}) = \sum_{j}^{k} p_j(\mathbf{x})\beta_j, \quad k \le n_d \tag{3.29}$$

where $p_j(\mathbf{x})$ are monomials that are included in Eq. (3.12). Such a given field can always be written using Eq. (3.12) using all the basis terms, including those in Eq. (3.29):

$$f(\mathbf{x}) = \sum_{j}^{n_d} p_j(\mathbf{x})\beta_j = \mathbf{p}^T(\mathbf{x})\boldsymbol{\alpha} \tag{3.30}$$

where

$$\boldsymbol{\alpha}^T = [\beta_1, \beta_2, \ldots, \beta_k, 0, \ldots, 0] \tag{3.31}$$

Using n nodes in the support domain of $\mathbf{x}$, we can obtain the vector of nodal function value $\mathbf{d}_e$ as:

$$\mathbf{d}_e = \begin{Bmatrix} f_1 \\ f_2 \\ \vdots \\ f_k \\ f_{k+1} \\ \vdots \\ f_n \end{Bmatrix} = \mathbf{P} = \begin{bmatrix} p_1(\mathbf{x}_1) & p_2(\mathbf{x}_1) & \cdots & p_k(\mathbf{x}_1) & p_{k+1}(\mathbf{x}_1) & p_{n_d}(\mathbf{x}_1) \\ p_1(\mathbf{x}_2) & p_2(\mathbf{x}_2) & \cdots & p_k(\mathbf{x}_2) & p_{k+1}(\mathbf{x}_2) & p_{n_d}(\mathbf{x}_2) \\ \vdots & \vdots & \ldots & \vdots & \vdots & \vdots \\ p_1(\mathbf{x}_k) & p_2(\mathbf{x}_k) & \cdots & p_k(\mathbf{x}_k) & p_{k+1}(\mathbf{x}_k) & p_{n_d}(\mathbf{x}_k) \\ p_1(\mathbf{x}_{k+1}) & p_2(\mathbf{x}_{k+1}) & \cdots & p_k(\mathbf{x}_{k+1}) & p_{k+1}(\mathbf{x}_{k+1}) & p_{n_d}(\mathbf{x}_{k+1}) \\ \vdots & \vdots & \ldots & \vdots & \vdots & \vdots \\ p_1(\mathbf{x}_{n_d}) & p_2(\mathbf{x}_{n_d}) & \cdots & p_k(\mathbf{x}_{n_d}) & p_{k+1}(\mathbf{x}_{n_d}) & p_{n_d}(\mathbf{x}_{n_d}) \end{bmatrix} \begin{Bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_k \\ 0 \\ \vdots \\ 0 \end{Bmatrix}$$

$$= \mathbf{P}\boldsymbol{\alpha} \tag{3.32}$$

Substituting Eq. (3.32) into Eq. (3.25), we have the approximation of

$$u^h(\mathbf{x}) = \mathbf{p}^T(\mathbf{x})\mathbf{P}^{-1}\mathbf{d}_e = \mathbf{p}^T(\mathbf{x})\mathbf{P}^{-1}\mathbf{P}\boldsymbol{\alpha} = \mathbf{p}^T(\mathbf{x})\boldsymbol{\alpha} = \sum_{j}^{k} p_j(\mathbf{x})\alpha_j = f(\mathbf{x}) \tag{3.33}$$

which is exactly what is given in Eq. (3.30). This proves that any field given by Eq. (3.29) will be exactly reproduced in the element by the approximation using the shape functions, as long as the given field function is included in the basis functions used for constructing the shape functions. This feature of the shape function is in fact also very easy to understand by intuition: any function given in the form of $f(\mathbf{x}) = \sum_j^k p_j(\mathbf{x})\beta_j$ can be produced exactly by letting $\alpha_j = \beta_j$ $(j = 1, 2, \ldots, k)$ and $\alpha_j = 0$ $(j = k + 1, \ldots, n_d)$. This can always be done as long as the moment matrix $\mathbf{P}$ is invertible so as to ensure the uniqueness of the solution for $\boldsymbol{\alpha}$.

The proof of the consistency of the shape function implies another important feature of the shape function: that is the *reproduction property*, which states that any function that appears in the basis can be reproduced exactly. This important property can be used for creating fields of special features. To ensure that the shape functions have linear consistency, all one needs to do is include the constant (unit) and linear monomials into the basis. We can make use of the feature of the shape function to compute accurate results for problems by including terms in the basis functions that are good approximations of the problem solution. The difference between consistency and reproduction is

- consistency depends upon the complete order of the basis functions; and
- reproduction depends upon whatever is included in the basis functions.

Property 2. Linear independence

Shape functions are linearly-independent. This is because basis functions are of linear independence and $\mathbf{P}^{-1}$ is assumed to exist. The existence of $\mathbf{P}^{-1}$ implies that the shape functions are equivalent to the basis functions in the function space, as shown in Eq. (3.26). Because the basis functions are linearly-independent, the shape functions are hence linearly-independent. Many FEM users do not pay much attention to this linear independence property; however, it is the foundation for the shape functions to have the delta function property stated below.

Property 3. Delta function properties

$$N_i(\mathbf{x}_j) = \delta_{ij} = \begin{cases} 1 & i = j,\ j = 1, 2, \ldots, n_d \\ 0 & i \neq j,\ i, j = 1, 2, \ldots, n_d \end{cases} \tag{3.34}$$

where δ_{ij} is the delta function. The delta function property implies that the shape function N_i should be unit at its *home node i*, and vanishes at the *remote nodes* $j \neq i$ of the element.

The delta function property can be proven easily as follows: because the shape functions $N_i(\mathbf{x})$ are linearly-independent, any vector of length n_d should be uniquely produced by linear combination of these n_d shape functions. Assume that the displacement at node i is d_i and the displacements at other nodes are zero, i.e.

$$\mathbf{d}_e = \{0, 0, \ldots, d_i, \ldots, 0\}^T \tag{3.35}$$

and substitute the above equation into Eq. (3.24), we have at $\mathbf{x} = \mathbf{x}_j$, that

$$u^h(\mathbf{x}_j) = \sum_{k=1}^{n_d} N_k(\mathbf{x}_j) d_k = N_i(\mathbf{x}_j) d_i \tag{3.36}$$

and when $i = j$, we must have

$$u_i = d_i = N_i(\mathbf{x}_i) d_i \tag{3.37}$$

which implies that

$$N_i(\mathbf{x}_i) = 1 \tag{3.38}$$

This proves the first row of Eq. (3.34). When $i \neq j$, we must have

$$u_j = 0 = N_i(\mathbf{x}_j) d_i \tag{3.39}$$

which requires

$$N_i(\mathbf{x}_j) = 0 \tag{3.40}$$

This proves the second row of Eq. (3.34). We can then conclude that the shape functions possess the delta function property, as depicted by Eq. (3.34). Note that there are elements, such as the thin beam and plate elements, whose shape functions may not possess the delta function property (see Section 5.2.1 for details).

Property 4. Partitions of unity property
Shape functions are partitions of unity:

$$\sum_{i=1}^{n} N_i(\mathbf{x}) = 1 \tag{3.41}$$

if the constant is included in the basis. This can be proven easily from the reproduction feature of the shape function. Let $u(\mathbf{x}) = c$, where c is a constant; we should have

$$\mathbf{d}_e = \begin{Bmatrix} d_1 \\ d_2 \\ \vdots \\ d_{n_d} \end{Bmatrix} = \begin{Bmatrix} c \\ c \\ \vdots \\ c \end{Bmatrix} \tag{3.42}$$

which implies the same constant displacement for all the nodes. Substituting the above equation into Eq. (3.24), we obtain

$$u(\mathbf{x}) = c \underset{\text{reproduction}}{=} u^h(\mathbf{x}) \underset{\text{approximation}}{=} \sum_{i=1}^{n_d} N_i(\mathbf{x})d_i = \sum_{i=1}^{n_d} N_i(\mathbf{x})c = c\sum_{i=1}^{n_d} N_i(\mathbf{x}) \tag{3.43}$$

which gives Eq. (3.41). This shows that the partitions of unity of the shape functions in the element allows a constant field or rigid body movement to be reproduced. Note that Eq. (3.41) does not require $0 \le N_i(\mathbf{x}) \le 1$.

Property 5. Linear field reproduction
If the first order monomial is included in the basis, the shape functions constructed reproduce the linear field, i.e.

$$\sum_{i=1}^{n_d} N_i(x)x_i = x \tag{3.44}$$

where x_i is the nodal values of the linear field. This can be proven easily from the reproduction feature of the shape function in exactly the same manner for proving Property 4. Let $u(x) = x$, we should have

$$\mathbf{d}_e = \left\{x_1, x_2, \ldots, x_{n_d}\right\}^T \tag{3.45}$$

Substituting the above equation into Eq. (3.24), we obtain

$$u^h(x) = x = \sum_{1}^{n_d} N_i(x)x_i \tag{3.46}$$

which is Eq. (3.44).

Lemma 1. *Condition for shape functions being partitions of unity.* For a set of shape functions in the general form

$$N_i(\mathbf{x}) = c_{1i} + c_{2i}p_2(\mathbf{x}) + c_{3i}p_3(\mathbf{x}) + \cdots + c_{n_d i}p_{n_d}(\mathbf{x}) \tag{3.47}$$

where $p_i(\mathbf{x})$ $(p_1(\mathbf{x}) = 1, i = 2, n_d)$ is a set of independent base functions, the sufficient and necessary condition for this set of shape functions being partitions of unity is

$$\begin{aligned} C_1 &= 1 \\ C_2 &= C_3 = \cdots = C_{n_d} = 0 \end{aligned} \tag{3.48}$$

where

$$C_k = \sum_{i=1}^{n_d} c_{ki} \tag{3.49}$$

Proof. Using Eq. (3.47), the summation of the shape functions is

$$\begin{aligned} \sum_{i=1}^{n_d} N_i(\mathbf{x}) &= \sum_{i=1}^{n_d} c_{1i} + p_2(\mathbf{x})\sum_{i=1}^{n_d} c_{2i} + p_3(\mathbf{x})\sum_{i=1}^{n_d} c_{3i} + \cdots + p_{n_d}(\mathbf{x})\sum_{i=1}^{n_d} c_{n_d i} \\ &= \underbrace{C_1}_{1} + \underbrace{C_2}_{0} p_2(\mathbf{x}) + \underbrace{C_2}_{0} p_3(\mathbf{x}) + \cdots + \underbrace{C_{n_d}}_{0} p_{n_d}(\mathbf{x}) = 1 \end{aligned} \tag{3.50}$$

which proofs the sufficient condition. To proof the necessary condition, we argue that, to have the partitions of unity, we have

$$\sum_{i=1}^{n_d} N_i(\mathbf{x}) = C_1 + C_2p_2(\mathbf{x}) + C_2p_3(\mathbf{x}) + \cdots + C_{n_d}p_{n_d}(\mathbf{x}) = 1 \tag{3.51}$$

or

$$(C_1 - 1) + C_2p_2(\mathbf{x}) + C_2p_3(\mathbf{x}) + \cdots + C_{n_d}p_{n_d}(\mathbf{x}) = 0 \tag{3.52}$$

Because $p_i(\mathbf{x})$ $(p_1(\mathbf{x}) = 1, i = 2, n_d)$ is a set of independent base functions. The necessary condition for Eq. (3.52) to be satisfied is Eq. (3.48).

Lemma 2. *Condition for shape functions being partitions of unity.* Any set of n_d shape functions will automatically satisfy the partitions of unity property if it satisfies:

- *Condition 1*: it is given in a linear combination of the same linearly-independent set of n_d basis functions that contain the constant basis, and the moment matrix defined by Eq. (3.21) is of full rank;
- *Condition 2*: it possesses the delta function property.

Proof. From Eq. (3.26), we can see that all the n_d shape functions are formed via a combination of the same basis function $p_i(\mathbf{x})$ $(i = 1, 2, \ldots, n_d)$. This feature, together with the delta function property, can ensure the property of partitions of unity. To prove this, we write a set of shape functions in the general form

$$N_i(\mathbf{x}) = c_{1i} + c_{2i}p_2(\mathbf{x}) + c_{3i}p_3(\mathbf{x}) + \cdots + c_{n_d i}p_{n_d}(\mathbf{x}) \tag{3.53}$$

where we ensured the inclusion of the constant basis of $p_1(\mathbf{x}) = 1$. The other basis function $p_i(\mathbf{x})$ $(i = 2, \ldots, n_d)$ in Eq. (3.53) can be monomials or any other type of basis functions as long as all the basis functions (including $p_1(\mathbf{x})$) are linearly-independent.

From the Condition 2, the shape functions possess a delta function property that leads to

$$\sum_{i=1}^{n_d} N_i(\mathbf{x}_j) = 1 \quad \text{for } j = 1, 2, \ldots, n_d \tag{3.54}$$

Substituting Eq. (3.53) into the previous equations, we have

$$\sum_{i=1}^{n_d} c_{1i} + p_2(\mathbf{x}_j)\sum_{i=1}^{n_d} c_{2i} + p_3(\mathbf{x}_j)\sum_{i=1}^{n_d} c_{3i} + \cdots + p_{n_d}(\mathbf{x}_j)\sum_{i=1}^{n_d} c_{n_d i} = 1 \quad \text{for } j = 1, 2, \ldots, n_d \tag{3.55}$$

or

$$C_1 + p_2(\mathbf{x}_j)C_2 + p_3(\mathbf{x}_j)C_3 + \cdots + p_{n_d}(\mathbf{x}_j)C_{n_d} = 1 \quad \text{for } j = 1, 2, \ldots, n_d \tag{3.56}$$

Expanding Eq. (3.56) gives

$$\begin{aligned} &C_1 + p_2(\mathbf{x}_1)C_2 + p_3(\mathbf{x}_1)C_3 + \cdots + p_{n_d}(\mathbf{x}_1)C_{n_d} = 1 \\ &C_1 + p_2(\mathbf{x}_2)C_2 + p_3(\mathbf{x}_2)C_3 + \cdots + p_{n_d}(\mathbf{x}_2)C_{n_d} = 1 \\ &\vdots \\ &C_1 + p_2(\mathbf{x}_{n_d})C_2 + p_3(\mathbf{x}_{n_d})C_3 + \cdots + p_{n_d}(\mathbf{x}_{n_d})C_{n_d} = 1 \end{aligned} \tag{3.57}$$

or in the matrix form

$$\begin{bmatrix} 1 & p_2(\mathbf{x}_2) & p_3(\mathbf{x}_2) & \cdots & p_{n_d}(\mathbf{x}_2) \\ 1 & p_2(\mathbf{x}_3) & p_3(\mathbf{x}_3) & \cdots & p_{n_d}(\mathbf{x}_3) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & p_2(\mathbf{x}_{n_d}) & p_3(\mathbf{x}_{n_d}) & \cdots & p_{n_d}(\mathbf{x}_{n_d}) \end{bmatrix} \begin{Bmatrix} C_1 - 1 \\ C_2 \\ C_3 \\ \vdots \\ C_{n_d} \end{Bmatrix} = 0 \tag{3.58}$$

Note that the coefficient matrix of Eq. (3.58) is the moment matrix that has a full rank (Condition 1); we then have

$$\begin{aligned} &C_1 = 1 \\ &C_2 = C_3 = \cdots = C_{n_d} = 0 \end{aligned} \tag{3.59}$$

The use of Lemma 1 proves the partitions of the unity property of shape functions.

Lemma 3. *Condition for shape functions being linear field reproduction.* Any set of n_d shape functions will automatically satisfy the linear reproduction property, if it satisfies

- *Condition 1*: it is given in a linear combination of the same linearly-independent set of n_d basis functions that contain the linear basis function, and the moment matrix defined by Eq. (3.21) is of full rank;
- *Condition 2*: it possesses the delta function property.

To prove this, we write a set of shape functions in the following general form of

$$N_i(\mathbf{x}) = c_{1i}p_1(\mathbf{x}) + c_{2i}x + c_{3i}p_3(\mathbf{x}) + \cdots + c_{ni}p_{n_d}(\mathbf{x}) \tag{3.60}$$

where we ensure inclusion of the complete linear basis functions of $p_2(\mathbf{x}) = x$. The other basis function $p_i(\mathbf{x})$ $(i = 1, 3, \ldots, n_d)$ in Eq. (3.53) can be monomials or any other type of basis function as long as all the basis functions are linearly-independent.

Consider a linear field of $u(\mathbf{x}) = x$, we should have the nodal vector as follows:

$$\mathbf{d}_e = \left\{x_1, x_2, \ldots, x_{n_d}\right\}^T \tag{3.61}$$

Substituting the above equation into Eq. (3.24), we obtain

$$\begin{aligned}
u^h(\mathbf{x}) &= \sum_{i=1}^{n_d} N_i(\mathbf{x})x_i \\
&= \sum_{i=1}^{n_d}[c_{1i}p_1(\mathbf{x}) + c_{2i}x + c_{3i}p_3(\mathbf{x}) + \cdots + c_{n_di}p_{n_d}(\mathbf{x})]x_i \\
&= \sum_{i=1}^{n_d} c_{1i}p_1(\mathbf{x})x_i + \sum_{i=1}^{n_d} c_{2i}xx_i + \sum_{i=1}^{n_d} c_{3i}p_3(\mathbf{x})x_i + \cdots + \sum_{i=1}^{n_d} c_{n_di}p_{n_d}(\mathbf{x})x_i \\
&= \sum_{i=1}^{n_d} c_{1i}p_1(\mathbf{x})x_i + x\sum_{i=1}^{n_d} c_{2i}x_i + p_3(\mathbf{x})\sum_{i=1}^{n_d} c_{3i}x_i + \cdots + p_{n_d}(\mathbf{x})\sum_{i=1}^{n_d} c_{n_di}x_i \\
&= p_1(\mathbf{x})C_{x1} + xC_{x2} + p_3(\mathbf{x})C_{x3} + \cdots + p_{n_d}(\mathbf{x})C_{xn_d}
\end{aligned} \tag{3.62}$$

At the n_d nodes of the element, we have n_d equations:

$$\begin{aligned}
u^h(\mathbf{x}_1) &= p_1(\mathbf{x})C_{x1} + x_1C_{x2} + p_3(\mathbf{x}_1)C_{x3} + \cdots + p_{n_d}(\mathbf{x}_1)C_{xn_d} \\
u^h(\mathbf{x}_2) &= p_1(\mathbf{x})C_{x1} + x_2C_{x2} + p_3(\mathbf{x}_2)C_{x3} + \cdots + p_{n_d}(\mathbf{x}_2)C_{xn_d} \\
&\vdots \\
u^h(\mathbf{x}_{n_d}) &= p_1(\mathbf{x})C_{x1} + x_{n_d}C_{x2} + p_3(\mathbf{x}_{n_d})C_{x3} + \cdots + p_{n_d}(\mathbf{x}_{n_d})C_{xn_d}
\end{aligned} \tag{3.63}$$

Using the delta function property of the shape functions, we have

$$\begin{aligned} u^h(\mathbf{x}_j) &= \sum_{i=1}^{n_d} N_i(\mathbf{x}_j)x_i \\ &= \underbrace{N_1(\mathbf{x}_j)}_{0}x_1 + \underbrace{N_1(\mathbf{x}_j)}_{0}x_2 + \cdots + \underbrace{N_1(\mathbf{x}_j)}_{1}x_j + \cdots + \underbrace{N_1(\mathbf{x}_j)}_{0}x_{n_d} \\ &= x_j \end{aligned} \tag{3.64}$$

Hence, Eq. (3.63) becomes

$$\begin{aligned} 0 &= p_1(\mathbf{x})C_{x1} + x_1(C_{x2} - 1) + p_3(\mathbf{x}_2)C_{x3} + \cdots + p_{n_d}(\mathbf{x}_2)C_{xn_d} \\ 0 &= p_1(\mathbf{x}_3)C_{x1} + x_2(C_{x2} - 1) + p_3(\mathbf{x}_3)C_{x3} + \cdots + p_{n_d}(\mathbf{x}_3)C_{xn_d} \\ &\vdots \\ 0 &= p_1(\mathbf{x}_{n_d})C_{x1} + x_{n_d}(C_{x2} - 1) + p_3(\mathbf{x}_{n_d})C_{x3} + \cdots + p_{n_d}(\mathbf{x}_{n_d})C_{xn_d} \end{aligned} \tag{3.65}$$

Or in matrix form,

$$\begin{bmatrix} p_1(\mathbf{x}_1) & x_1 & p_3(\mathbf{x}_1) & \cdots & p_{n_d}(\mathbf{x}_1) \\ p_1(\mathbf{x}_1) & x_2 & p_3(\mathbf{x}_2) & \cdots & p_{n_d}(\mathbf{x}_2) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_1(\mathbf{x}_1) & x_{n_d} & p_3(\mathbf{x}_{n_d}) & \cdots & p_{n_d}(\mathbf{x}_{n_d}) \end{bmatrix} \begin{Bmatrix} C_{x1} \\ C_{x2} - 1 \\ C_{x3} \\ \vdots \\ C_{xn_d} \end{Bmatrix} = 0 \tag{3.66}$$

Note that the coefficient matrix of Eq. (3.66) is the moment matrix that has a full rank. We thus have

$$\begin{aligned} C_{x1} &= 0 \\ (C_{x2} - 1) &= 0 \\ C_{x3} = \cdots = C_{n_d} &= 0 \end{aligned} \tag{3.67}$$

Substituting the previous equation back into Eq. (3.62), we obtain

$$u^h(x) = \sum_{i=1}^{n_d} N_i(x)x_i = x \tag{3.68}$$

This proves the property of linear field reproduction.

The delta function property (Property 3) ensures convenient imposition of the essential boundary conditions (admissible condition (b) required by Hamilton's principle), because the nodal displacement at a node is independent of that at any other nodes. The constraints can often be written in the form of a so-called *Single Point Constraint* (SPC). If the displacement at a node is fixed, all one needs to do is to remove corresponding rows and columns without affecting the other rows and columns.

The proof of Property 4 gives a convenient way to confirm the partitions of unity property of shape functions. As long as the constant (unit) basis is included in the basis functions, the shape functions constructed are partitions of unity. Properties 4 and 5 are essential for the FEM to pass the standard *patch test*, used for decades in the finite element method for validating the elements. In the standard patch test, the patch is meshed with a number of elements, with at least one interior node. Linear displacements are then enforced on the boundary (edges) of the patch. A successful patch test requires the FEM solution to produce the linear displacement (or constant strain) field at any interior node. Therefore, the property of reproduction of a linear field of shape function provides the foundation for passing the patch test. Note that the property of reproducing the linear field of the shape function does not guarantee successful patch tests, as there could be other sources of numerical error, such as numerical integration, which can cause failure.

Lemma 1 seems to be redundant, since we already have Property 4. However, Lemma 1 is a very convenient property to use for checking the property of partitions of unity of shape functions that are constructed using other *shortcut* methods, rather than the standard procedure described in Section 3.4.3. Using Lemma 1, one only needs to make sure whether the shape functions satisfy Eq. (3.48).

Lemma 2 is another very convenient property to use for checking the property of partitions of unity of shape functions. Using Lemma 2, we only need to make sure that the constructed n_d shape functions are of the delta function property, and they are linear combinations of the same n_d basis functions that are linearly-independent and contain the constant basis function. The conformation of full-rank of the moment matrix of the basis functions can sometimes be difficult. In this book, we usually assume that the rank is full for the normal elements, as long as the basis functions are linearly-independent. In usual situations, one will not be able to obtain the shape functions if the rank of the moment matrix is not full. If we somehow obtained the shape functions successfully, we can usually be sure that the rank of the corresponding moment matrix is full.

Lemma 3 is a very convenient property to use for checking the property of linear field reproduction of shape functions. Using Lemma 3, we only need to make sure that the constructed n_d shape functions are of the delta function property, and they are linear combinations of the same n_d basis functions that are linearly-independent and contain the linear basis function.

3.4.5 Formation of FE Equations in Local Coordinate System

Once the shape functions are constructed, the FE equation for an element can be formulated using the following process. By substituting the interpolation of the nodes, Eq. (3.6), and the strain–displacement equation, say Eq. (2.5), into the strain energy term (Eq. (3.4)), we have

$$\Pi = \frac{1}{2}\int_{V_e} \boldsymbol{\varepsilon}^T \mathbf{c}\boldsymbol{\varepsilon}\, \mathrm{d}V = \frac{1}{2}\int_{V_e} \mathbf{d}_e^T \mathbf{B}^T \mathbf{c}\mathbf{B}\, \mathbf{d}_e\, \mathrm{d}V = \frac{1}{2}\mathbf{d}_e^T \left(\int_{V_e} \mathbf{B}^T \mathbf{c}\mathbf{B}\, \mathrm{d}V \right) \mathbf{d}_e \tag{3.69}$$

where the subscript e stands for the element. Note that the volume integration over the global domain has been changed to that over the elements. This can be done because we assume that the assumed displacement field satisfies the compatibility condition (see Section 3.3) on all the edges between the elements. Otherwise, techniques discussed in Chapter 11 are needed. In Eq.(3.69), **B** is often called the *strain matrix*, defined by

$$\mathbf{B} = \mathbf{L}\mathbf{N} \tag{3.70}$$

where **L** is the differential operator that is defined for different problems in Chapter 2. For 3D solids, it is given by Eq. (2.7). By denoting

$$\mathbf{k}_e = \int_{V_e} \mathbf{B}^T \mathbf{c} \mathbf{B} \, \mathrm{d}V \tag{3.71}$$

which is called the element stiffness matrix, Eq. (3.69) can be rewritten as

$$\Pi = \tfrac{1}{2} \mathbf{d}_e^T \mathbf{k}_e \mathbf{d}_e \tag{3.72}$$

Note that the stiffness matrix $\mathbf{k}_e$ is symmetrical, because

$$[\mathbf{k}_e]^T = \int_{V_e} [\mathbf{B}^T \mathbf{c} \mathbf{B}]^T \, \mathrm{d}V = \int_{V_e} \mathbf{B}^T \mathbf{c}^T [\mathbf{B}^T]^T \, \mathrm{d}V = \int_{V_e} \mathbf{B}^T \mathbf{c} \mathbf{B} \, \mathrm{d}V = \mathbf{k}_e \tag{3.73}$$

which shows that the transpose of matrix $\mathbf{k}_e$ is itself. In deriving the above equation, $\mathbf{c} = \mathbf{c}^T$ has been employed. Making use of the symmetry of the stiffness matrix, only half of the terms in the matrix need to be evaluated and stored.

By substituting Eq. (3.6) into Eq. (3.3), the kinetic energy can be expressed as

$$T = \frac{1}{2} \int_{V_e} \rho \dot{\mathbf{U}}^T \dot{\mathbf{U}} \, \mathrm{d}V = \frac{1}{2} \int_{V_e} \rho \dot{\mathbf{d}}_e^T \mathbf{N}^T \mathbf{N} \dot{\mathbf{d}}_e \, \mathrm{d}V = \frac{1}{2} \dot{\mathbf{d}}_e^T \left(\int_{V_e} \rho \mathbf{N}^T \mathbf{N} \, \mathrm{d}V \right) \dot{\mathbf{d}}_e \tag{3.74}$$

By denoting

$$\mathbf{m}_e = \int_{V_e} \rho \mathbf{N}^T \mathbf{N} \, \mathrm{d}V \tag{3.75}$$

which is called the mass matrix of the element, Eq. (3.74) can be rewritten as

$$T = \tfrac{1}{2} \dot{\mathbf{d}}_e^T \mathbf{m}_e \dot{\mathbf{d}}_e \tag{3.76}$$

It is obvious that the element mass matrix is also symmetrical.

Finally, to obtain the work done by the external forces, Eq. (3.6) is substituted into Eq. (3.5):

$$W_f = \int_{V_e} \mathbf{d}_e^T \mathbf{N}^T \mathbf{f}_b \, \mathrm{d}V + \int_{S_e} \mathbf{d}_e^T \mathbf{N}^T \mathbf{f}_s \, \mathrm{d}S = \mathbf{d}_e^T \left(\int_{V_e} \mathbf{N}^T \mathbf{f}_b \, \mathrm{d}V \right) + \mathbf{d}_e^T \left(\int_{S_e} \mathbf{N}^T \mathbf{f}_s \, \mathrm{d}S \right) \tag{3.77}$$

where the surface integration is performed only for elements on the force boundary of the problem domain. By denoting

$$\mathbf{F}_b = \int_{V_e} \mathbf{N}^T \mathbf{f}_b \, \mathrm{d}V \tag{3.78}$$

and

$$\mathbf{F}_s = \int_{S_e} \mathbf{N}^T \mathbf{f}_s \,\mathrm{d}S \tag{3.79}$$

Eq. (3.77) can then be rewritten as

$$W_f = \mathbf{d}_e^T \mathbf{F}_b + \mathbf{d}_e^T \mathbf{F}_s = \mathbf{d}_e^T \mathbf{f}_e \tag{3.80}$$

$\mathbf{F}_b$ and $\mathbf{F}_s$ are the nodal forces acting on the nodes of the elements, which are equivalent to the body forces and surface forces applied on the element in terms of the work done on a virtual displacement. These two nodal force vectors can then be added up to form the total nodal force vector $\mathbf{f}_e$:

$$\mathbf{f}_e = \mathbf{F}_b + \mathbf{F}_s \tag{3.81}$$

Substituting Eqs. (3.72), (3.76) and (3.80) into Lagrangian functional L (Eq. (3.2)), we have

$$L = \tfrac{1}{2}\dot{\mathbf{d}}_e^T \mathbf{m}_e \dot{\mathbf{d}}_e - \tfrac{1}{2}\mathbf{d}_e^T \mathbf{k}_e \mathbf{d}_e + \mathbf{d}_e^T \mathbf{f}_e \tag{3.82}$$

Applying Hamilton's principle (Eq. (3.1)), we have

$$\delta \int_{t_1}^{t_2} \left(\frac{1}{2}\dot{\mathbf{d}}_e^T \mathbf{m}_e \dot{\mathbf{d}}_e - \frac{1}{2}\mathbf{d}_e^T \mathbf{k}_e \mathbf{d}_e + \mathbf{d}_e^T \mathbf{f}_e \right) \mathrm{d}t = 0 \tag{3.83}$$

Note that the variation and integration operators are interchangeable, hence we obtain

$$\int_{t_1}^{t_2} (\delta\dot{\mathbf{d}}_e^T \mathbf{m}_e \dot{\mathbf{d}}_e - \delta\mathbf{d}_e^T \mathbf{k}_e \mathbf{d}_e + \delta\mathbf{d}_e^T \mathbf{f}_e)\,\mathrm{d}t = 0 \tag{3.84}$$

To explicitly illustrate the process of deriving Eq. (3.84) from Eq. (3.83), we use a two-degree of freedom system as an example. Here, we show the procedure for deriving the second term in Eq. (3.84):

$$\begin{aligned}
\delta\left(\frac{1}{2}\mathbf{d}_e^T \mathbf{k}_e \mathbf{d}_e\right) &= \delta\left(\frac{1}{2}\{d_1 \quad d_2\} \begin{bmatrix} k_{11} & k_{12} \\ k_{12} & k_{22} \end{bmatrix} \begin{Bmatrix} d_1 \\ d_2 \end{Bmatrix}\right) \\
&= \frac{1}{2}\delta\left(\left\{ d_1k_{11} + d_2k_{12} \quad d_1k_{12} + d_2k_{22} \right\} \begin{Bmatrix} d_1 \\ d_2 \end{Bmatrix} \right) \\
&= \frac{1}{2}\delta\left(d_1^2k_{11} + 2d_1d_2k_{12} + d_2^2k_{22} \right) \\
&= \frac{1}{2}\left[\frac{\partial\left(d_1^2k_{11} + 2d_1d_2k_{12} + d_2^2k_{22}\right)}{\partial d_1}\delta d_1 + \frac{\partial\left(d_1^2k_{11} + 2d_1d_2k_{12} + d_2^2k_{22}\right)}{\partial d_2}\delta d_2 \right] \\
&= (d_1k_{11} + d_2k_{12})\delta d_1 + (d_1k_{12} + d_2k_{22})\delta d_2 \\
&= \left\{ \delta d_1 \quad \delta d_2 \right\} \begin{Bmatrix} d_1k_{11} + d_2k_{12} \\ d_1k_{12} + d_2k_{22} \end{Bmatrix} = \left\{ \delta d_1 \quad \delta d_2 \right\} \begin{bmatrix} k_{11} & k_{12} \\ k_{12} & k_{22} \end{bmatrix} \begin{Bmatrix} d_1 \\ d_2 \end{Bmatrix} = \delta\mathbf{d}_e^T \mathbf{k}_e \mathbf{d}_e
\end{aligned}$$

In Eq. (3.84), the variation and differentiation with time are also interchangeable, i.e.

$$\delta \dot{\mathbf{d}}_e^T = \delta \left(\frac{\mathrm{d}\mathbf{d}_e^T}{\mathrm{d}t} \right) = \frac{\mathrm{d}}{\mathrm{d}t} \left(\delta \mathbf{d}_e^T \right) \tag{3.85}$$

Hence, by substituting Eq. (3.85) into Eq. (3.84), and integrating the first term by parts, we obtain

$$\int_{t_1}^{t_2} \delta \dot{\mathbf{d}}_e^T \mathbf{m}_e \dot{\mathbf{d}}_e \,\mathrm{d}t = \underbrace{\delta \mathbf{d}_e^T \mathbf{m}_e \; \dot{\mathbf{d}}_e \big|_{t_1}^{t_2}}_{=0} - \int_{t_1}^{t_2} \delta \mathbf{d}_e^T \mathbf{m}_e \ddot{\mathbf{d}}_e \,\mathrm{d}t = - \int_{t_1}^{t_2} \delta \mathbf{d}_e^T \mathbf{m}_e \ddot{\mathbf{d}}_e \,\mathrm{d}t \tag{3.86}$$

Note that in deriving Eq. (3.86) as above, the condition $\delta \mathbf{d}_e = 0$ at t_1 and t_2 have been used, which leads to the vanishing of the first term on the right-hand side. This is because the initial condition at t_1 and final condition at t_2 have to be satisfied for any $\mathbf{d}_e$ (admissible conditions (c) required by Hamilton's principle), and no variation at t_1 and t_2 is allowed.

Substituting Eq. (3.86) into Eq. (3.84) leads to

$$\int_{t_1}^{t_2} \delta \mathbf{d}_e^T (-\mathbf{m}_e \ddot{\mathbf{d}}_e - \mathbf{k}\mathbf{d}_e + \mathbf{f}_e) \,\mathrm{d}t = 0 \tag{3.87}$$

To have the integration in Eq. (3.87) as zero for an arbitrary integrand, the integrand itself has to vanish, i.e.

$$\delta \mathbf{d}_e^T (-\mathbf{m}_e \ddot{\mathbf{d}}_e - \mathbf{k}\mathbf{d}_e + \mathbf{f}_e) = 0 \tag{3.88}$$

Due to the arbitrary nature of the variation of the displacements, the only insurance for Eq. (3.88) to be satisfied is

$$\mathbf{k}_e \mathbf{d}_e + \mathbf{m}_e \ddot{\mathbf{d}}_e = \mathbf{f}_e \tag{3.89}$$

Equation (3.89) is the FEM equation for an element, while $\mathbf{k}_e$ and $\mathbf{m}_e$ are the *stiffness* and *mass* matrices for the element, and $\mathbf{f}_e$ is the element force vector of the total external forces acting on the nodes of the element. All these element matrices and vectors can be obtained simply by integration for the given shape functions of displacements.

3.4.6 Coordinate transformation

The element equation given by Eq. (3.89) is formulated based on the local coordinate system defined on an element. In general, the structure is divided into many elements with different orientations (see Figure 3.4). To assemble all the element equations to form the global system equations, a coordinate transformation has to be performed for each element in order to formulate its element equation in reference to the *global coordinate system* which is defined on the whole structure.

The coordinate transformation gives the relationship between the displacement vector $\mathbf{d}_e$ based on the local coordinate system and the displacement vector $\mathbf{D}_e$ for the same element,

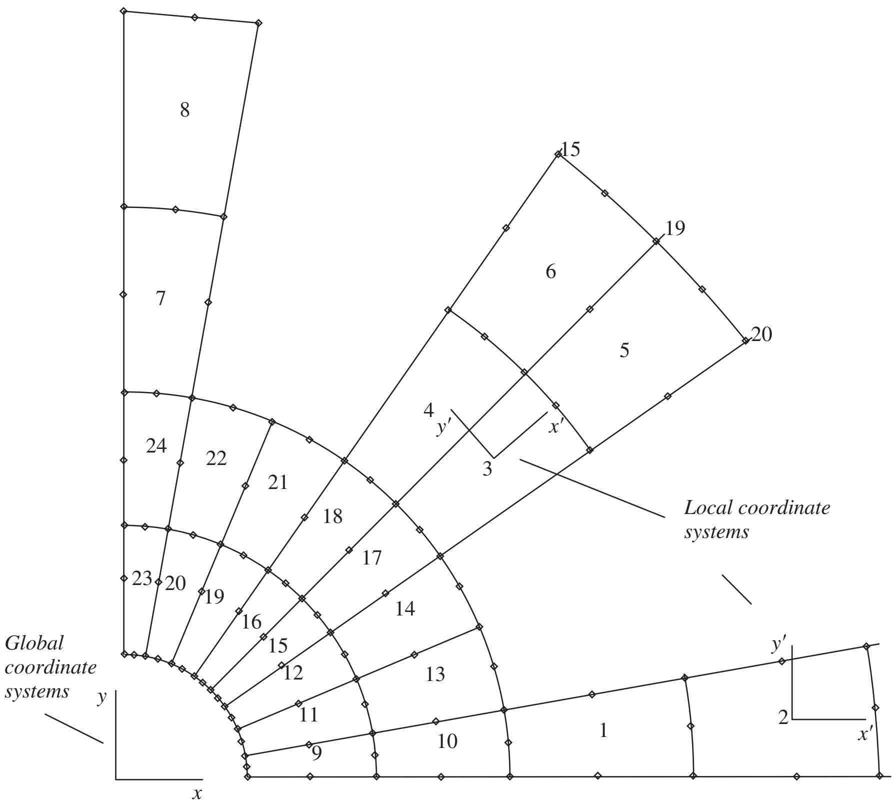


Figure 3.4. Local and global coordinate system.

but based on the global coordinate system:

$$\mathbf{d}_e = \mathbf{T}\mathbf{D}_e \tag{3.90}$$

T is the transformation matrix, which has different forms depending upon the type of element, and will be discussed in detail in later chapters. The transformation matrix can also be applied to the force vectors between the local and global coordinate systems:

$$\mathbf{f}_e = \mathbf{T}\mathbf{F}_e \tag{3.91}$$

in which $\mathbf{F}_e$ stands for the force vector at node i on the global coordinate system. Substitution of Eq. (3.90) into Eq. (3.89) leads to the element equation based on the global coordinate system:

$$\mathbf{K}_e\mathbf{D}_e + \mathbf{M}_e\ddot{\mathbf{D}}_e = \mathbf{F}_e \tag{3.92}$$

where

$$\mathbf{K}_e = \mathbf{T}^T \mathbf{k}_e \mathbf{T} \tag{3.93}$$

$$\mathbf{M}_e = \mathbf{T}^T \mathbf{m}_e \mathbf{T} \tag{3.94}$$

$$\mathbf{F}_e = \mathbf{T}^T \mathbf{f}_e \tag{3.95}$$

3.4.7 Assembly of Global FE Equation

The FE equations for all the individual elements can be assembled together to form the global FE system equation:

$$\mathbf{KD} + \mathbf{M}\ddot{\mathbf{D}} = \mathbf{F} \tag{3.96}$$

where **K** and **M** are the globe stiffness and mass matrix, **D** is a vector of all the displacements at all the nodes in the entire problem domain, and **F** is a vector of all the equivalent nodal force vectors. The process of assembly is one of simply adding up the contributions from all the elements connected at a node. The detailed process will be demonstrated in Chapter 4 using an example. It may be noted here that the assembly of the global matrices can be skipped by combining assembling with the equation solving. This means that the assembling of a term in the global matrix is done only when the equation solver is operating on this term.

3.4.8 Imposition of Displacement Constraints

The global stiffness matrix **K** in Eq. (3.96) does not usually have a full rank, because displacement constraints (supports) are not yet imposed, and it is non-negative definite or positive semi-definite. Physically, an unconstrained solid or structure is capable of performing rigid movement. Therefore, if the solid or structure is free of support, Eq. (3.96) gives the behavior that includes the rigid body dynamics, if it is subjected to dynamic forces. If the external forces applied are static, the displacements cannot be uniquely determined from Eq. (3.96) for any given force vector. It is meaningless to try to determine the static displacements of an unconstrained solid or structure that can move freely.

For constrained solids and structures, the constraints can be imposed by simply removing the rows and columns corresponding to the constrained nodal displacements. We shall demonstrate this method in an example problem in later chapters. After the treatment of constraints (and if the constraints are sufficient), the stiffness matrix **K** in Eq. (3.96) will be of full rank, and will be Positive Definite (PD). Since we have already proven that **K** is symmetric, **K** is of a Symmetric Positive Definite (SPD) property.

3.4.9 Solving the Global FE Equation

By solving the global FE equation, displacements at the nodes can be obtained. The strain and stress in any element can then be retrieved using Eq. (3.6) in Eqs. (2.5) and (2.8).

3.5 STATIC ANALYSIS

Static analysis involves the solving of Eq. (3.96) without the term with the global mass matrix, **M**. Hence, as discussed, the static system of equations takes the form

$$\mathbf{KD} = \mathbf{F} \tag{3.97}$$

There are numerous methods and algorithms to solve the above matrix equation. The methods often used are Gauss elimination and LU decompositions for small systems, and iterative methods for large systems. These methods are all routinely available in any math library of any computer system.

3.6 ANALYSIS OF FREE VIBRATION (EIGENVALUE ANALYSIS)

For a structural system with a total DOF of N, the stiffness matrix **K** and mass matrix **M** in Eq. (3.96) have a dimension of $N \times N$. By solving the above equation we can obtain the displacement field, and the stress and strain can then be calculated. The question now is how to solve this equation, as N is usually very large for practical engineering structures. One way to solve this equation is using the so-called direct integration method, discussed in the next section. An alternative way of solving Eq. (3.96) is using the so-called modal analysis technique (or mode superposition technique). In this technique, we first have to solve the homogenous equation of Eq. (3.96). The homogeneous equation is when we consider the case of $\mathbf{F} = 0$, therefore it is also called *free vibration* analysis, as the system is free of external forces. For a solid or structure that undergoes a free vibration, the discretized system equation Eq. (3.96) becomes

$$\mathbf{KD} + \mathbf{M}\ddot{\mathbf{D}} = 0 \tag{3.98}$$

This solution for the free vibration problem can be assumed as

$$\mathbf{D} = \phi \exp(\mathrm{i}\omega t) \tag{3.99}$$

where $\boldsymbol{\phi}$ is the amplitude of the nodal displacement, ω is the frequency of the free vibration, and t is the time. By substituting Eq. (3.99) into Eq. (3.98), we obtain

$$[\mathbf{K} - \omega^2\mathbf{M}]\boldsymbol{\phi} = 0 \tag{3.100}$$

or

$$[\mathbf{K} - \lambda\mathbf{M}]\boldsymbol{\phi} = 0 \tag{3.101}$$

where

$$\lambda = \omega^2 \tag{3.102}$$

Equation (3.100) (or (3.101)) is called the *eigenvalue equation*. To have a non-zero solution for ϕ, the determinate of the matrix must vanish:

$$\det[\mathbf{K} - \lambda\mathbf{M}] = |\mathbf{K} - \lambda\mathbf{M}| = 0 \tag{3.103}$$

The expansion of the above equation will lead to a polynomial of λ of order N. This polynomial equation will have N roots, $\lambda_1, \lambda_2, \ldots, \lambda_N$, called *eigenvalues*, which relate to the *natural frequency* of the system by Eq. (3.100). The natural frequency is a very important characteristic of the structure carrying dynamic loads. It has been found that if a structure is excited by a load with a frequency of one of the structure's natural frequencies, the structure can undergo extremely violent vibration, which often leads to catastrophic failure of the structural system. Such a phenomenon is called *resonance*. Therefore, an eigenvalue analysis has to be performed in designing a structural system that is to be subjected to dynamic loadings.

By substituting an eigenvalue λ_i back into the eigenvalue equation, Eq. (3.101), we have

$$[\mathbf{K} - \lambda_i \mathbf{M}]\boldsymbol{\phi} = 0 \tag{3.104}$$

which is a set of algebraic equations. Solving the above equation for $\boldsymbol{\phi}$, a vector denoted by $\boldsymbol{\phi}_i$ can then be obtained. This vector corresponding to the ith eigenvalue λ_i is called the ith *eigenvector* that satisfies the following equation:

$$[\mathbf{K} - \lambda_i \mathbf{M}]\boldsymbol{\phi}_i = 0 \tag{3.105}$$

An eigenvector $\boldsymbol{\phi}_i$ corresponds to a *vibration mode* that gives the shape of the vibrating structure of the ith mode. Therefore, analysis of the eigenvalue equation also gives very important information on possible vibration modes experienced by the structure when it undergoes a vibration. Vibration modes of a structure are therefore another important characteristic of the structure. Mathematically, the eigenvectors can be used to construct the displacement fields. It has been found that using a few of the lowest modes can obtain very accurate results for many engineering problems. Modal analysis techniques have been developed to take advantage of these properties of natural modes.

In Eq. (3.101), since the mass matrix $\mathbf{M}$ is symmetric positive definite and the stiffness matrix $\mathbf{K}$ is symmetric and either positive or positive semi-definite, the eigenvalues are all real and either positive or zero. It is possible that some of the eigenvalues may coincide. The corresponding eigenvalue equation is said to have multiple eigenvalues. If there are m coincident eigenvalues, the eigenvalue is said to be of a multiplicity of m. For an eigenvalue of multiplicity m, there are m vectors satisfying Eq. (3.105).

Methods for the effective computation of the eigenvalues and eigenvectors for an eigenvalue equations system like Eq. (3.100) or (3.101) are outside the scope of this book. Intensive research has been conducted to-date, and many sophisticated algorithms are already well established and readily available in the open literature, and routinely in computational libraries. The commonly used methods are (see, e.g. Petyt, 1990):

- Jacobi's method;
- Given's method and householder's method;
- the bisection method (using Sturm sequences);
- inverse iteration;
- QR method;
- subspace iteration;
- Lanczos' method.

3.7 TRANSIENT RESPONSE

Structural systems are very often subjected to *transient* excitation. A transient excitation is a highly dynamic, time-dependent force exerted on the solid or structure, such as earthquake, impact and shocks. The discrete governing equation system for such a structure is still Eq. (3.96), but it often requires a different solver from that used in eigenvalue analysis. The widely used method is the so-called *direct integration method*.

The direct integration method basically uses the *finite difference method* for time stepping to solve Eq. (3.96). There are two main types of direct integration method: *implicit* and *explicit*. Implicit methods are generally more efficient for a relatively slow phenomenon, and explicit methods are more efficient for a very fast phenomenon, such as impact and explosion. The literature on the various algorithms available to solving transient problems is vast. This section introduces the idea of time stepping used in finite difference methods, which are employed in solving transient problems.

Before discussing the equations used for the time stepping techniques, it should be mentioned that the general system equation for a structure can be re-written as

$$\mathbf{KD} + \mathbf{C}\dot{\mathbf{D}} + \mathbf{M}\ddot{\mathbf{D}} = \mathbf{F} \tag{3.106}$$

where $\dot{\mathbf{D}}$ is the vector of velocity components, and $\mathbf{C}$ is the matrix of damping coefficients that are determined experimentally. The damping coefficients are often expressed as proportions of the mass and stiffness matrices, called *proportional damping* (e.g. Petyt, 1990; Clough and Penzien, 1975). For a proportional damping system, C can be simply determined in the form

$$\mathbf{C} = c_K\mathbf{K} + c_M\mathbf{M} \tag{3.107}$$

where c_K and c_M are determined by experiments.

3.7.1 Central Difference Algorithm

We first write the system equation in the form

$$\mathbf{M}\ddot{\mathbf{D}} = \mathbf{F} - [\mathbf{C}\dot{\mathbf{D}} + \mathbf{KD}] = \mathbf{F} - \mathbf{F}^{\text{int}} = \mathbf{F}^{\text{residual}} \tag{3.108}$$

where $\mathbf{F}^{\text{residual}}$ is the residual force vector, and

$$\mathbf{F}^{\text{int}} = [\mathbf{C}\dot{\mathbf{D}} + \mathbf{KD}] \tag{3.109}$$

is defined as the internal force at time t. The acceleration, $\ddot{\mathbf{D}}$, can be simply obtained by

$$\ddot{\mathbf{D}} = \mathbf{M}^{-1}\mathbf{F}^{\text{residual}} \tag{3.110}$$

In practice, the above equation does not usually require solving of the matrix equation, since lumped masses are usually used which forms a diagonal mass matrix [Petyt, 1990].

The solution to Eq. (3.110) is thus trivial, and the matrix equation is the set of independent equations for each degree of freedom i as follows:

$$d_i = \frac{f_i^{\text{residual}}}{m_i} \tag{3.111}$$

where f_i^{residual} is the residual force, and m_i is the lumped mass corresponding to the ith DOF.

We now introduce the following finite central difference equations:

$$\mathbf{D}_{t+\Delta t} = 2\,(\Delta t)\,\dot{\mathbf{D}}_t + \mathbf{D}_{t-\Delta t} \tag{3.112}$$

$$\dot{\mathbf{D}}_{t+\Delta t} = 2\,(\Delta t)\,\ddot{\mathbf{D}}_t + \dot{\mathbf{D}}_{t-\Delta t} \tag{3.113}$$

$$\ddot{\mathbf{D}}_t = \frac{1}{(\Delta t)^2}(\mathbf{D}_{t+\Delta t} - 2\mathbf{D}_t + \mathbf{D}_{t-\Delta t}) \tag{3.114}$$

By eliminating $\mathbf{D}_{t+\Delta t}$ from Eqs. (3.112) and (3.114), we have

$$\mathbf{D}_{t-\Delta t} = \mathbf{D}_t - (\Delta t)\dot{\mathbf{D}}_t + \frac{(\Delta t)^2}{2}\ddot{\mathbf{D}}_t \tag{3.115}$$

To explain the time stepping procedure, refer to Figure 3.5, which shows an arbitrary plot of either displacement or velocity against time. The time stepping/marching procedure in the central difference method starts at $t = 0$, and computes the acceleration $\ddot{\mathbf{D}}_0$ using Eq. (3.110):

$$\ddot{\mathbf{D}}_0 = \mathbf{M}^{-1}\mathbf{F}_0^{\text{residual}} \tag{3.116}$$

For given initial conditions, $\mathbf{D}_0$ and $\dot{\mathbf{D}}_0$ are known. Substituting $\mathbf{D}_0, \dot{\mathbf{D}}_0$ and $\ddot{\mathbf{D}}_0$ into Eq. (3.115), we find $\mathbf{D}_{-\Delta t}$. Considering a half of the time step and using the central difference equations (3.112) and (3.113), we have

$$\mathbf{D}_{t+\Delta t/2} = (\Delta t)\dot{\mathbf{D}}_t + \mathbf{D}_{t-\Delta t/2} \tag{3.117}$$

$$\dot{\mathbf{D}}_{t+\Delta t/2} = (\Delta t)\ddot{\mathbf{D}}_t + \dot{\mathbf{D}}_{t-\Delta t/2} \tag{3.118}$$

The velocity, $\dot{\mathbf{D}}_{-\Delta t/2}$ at $t = -\Delta t/2$ can be obtained by Eq. (3.117) by performing the central differencing at $t = -\Delta t/2$ and using values of $\mathbf{D}_{-\Delta t}$ and $\mathbf{D}_0$:

$$\dot{\mathbf{D}}_{-\Delta t/2} = \frac{\mathbf{D}_0 - \mathbf{D}_{-\Delta t}}{(\Delta t)} \tag{3.119}$$

After this, Eq. (3.118) is used to compute $\dot{\mathbf{D}}_{\Delta t/2}$ using $\ddot{\mathbf{D}}_0$ and $\dot{\mathbf{D}}_{-\Delta t/2}$:

$$\dot{\mathbf{D}}_{\Delta t/2} = (\Delta t)\ddot{\mathbf{D}}_0 + \dot{\mathbf{D}}_{-\Delta t/2} \tag{3.120}$$

Then, Eq. (3.117) is used once again to compute $\mathbf{D}_{\Delta t}$ using $\dot{\mathbf{D}}_{\Delta t/2}$ and $\mathbf{D}_0$:

$$\mathbf{D}_{\Delta t} = (\Delta t)\dot{\mathbf{D}}_{\Delta t/2} + \mathbf{D}_0 \tag{3.121}$$

Once $\mathbf{D}_{\Delta t}$ is determined, Eq. (3.118) at $t = \Delta t/2$ can be used to obtain $\dot{\mathbf{D}}_{\Delta t}$ by assuming that the acceleration is constant over the step Δt and $\ddot{\mathbf{D}}_{\Delta t/2} = \ddot{\mathbf{D}}_0$, and using $\dot{\mathbf{D}}_0$, which

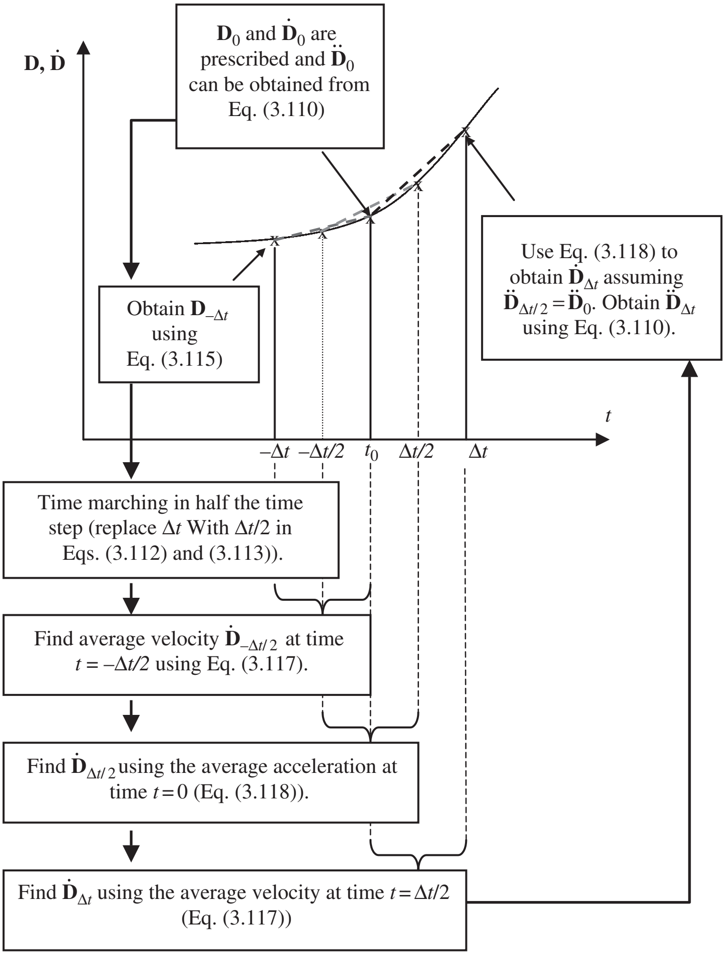

$\mathbf{D}, \dot{\mathbf{D}}$
$\mathbf{D}_0$ and $\dot{\mathbf{D}}_0$ are prescribed and $\ddot{\mathbf{D}}_0$ can be obtained from Eq. (3.110)
Obtain $\mathbf{D}_{-\Delta t}$ using Eq. (3.115)
Use Eq. (3.118) to obtain $\dot{\mathbf{D}}_{\Delta t}$ assuming $\ddot{\mathbf{D}}_{\Delta t/2} = \ddot{\mathbf{D}}_0$. Obtain $\ddot{\mathbf{D}}_{\Delta t}$ using Eq. (3.110).
t
$-\Delta t$
$-\Delta t/2$
t_0
$\Delta t/2$
Δt
Time marching in half the time step (replace Δt With $\Delta t/2$ in Eqs. (3.112) and (3.113)).
Find average velocity $\dot{\mathbf{D}}_{-\Delta t/2}$ at time $t = -\Delta t/2$ using Eq. (3.117).
Find $\dot{\mathbf{D}}_{\Delta t/2}$ using the average acceleration at time $t = 0$ (Eq. (3.118)).
Find $\dot{\mathbf{D}}_{\Delta t}$ using the average velocity at time $t = \Delta t/2$ (Eq. (3.117))

Figure 3.5. Time marching in the central difference algorithm: explicitly advancing in time.

is the prescribed initial velocity. At the next step in time, $\ddot{\mathbf{D}}_{\Delta t}$ is again computed using Eq. (3.110). The above process is then repeated. The time marching is continued until it reaches the final desired time.

Note that in the above process, the solution (displacement, velocity and acceleration) are obtained without solving any matrix form of system equation, but repeatedly using Eqs. (3.110), (3.117) and (3.118). The central difference method is therefore an *explicit method*. The time marching in explicit methods is therefore extremely fast, and the coding is also very straightforward. It is particularly suited for simulating highly nonlinear, large deformation, contact and extremely fast events of mechanics.

The central difference method, like most explicit methods, is *conditionally stable*. This means that if the time step, Δt, becomes too large to exceed a critical time step, Δt_{cr}, then the computed solution will become unstable and might grow without limit. The critical time step Δt_{cr} should be the time taken for the fastest stress wave in the solids/structure to cross the smallest element in the mesh. Therefore, the time steps used in the explicit methods are typically 100 to 1000 times smaller than those used with implicit methods, outlined in the next subsection. The need to use a small time step, and especially its dependence on the smallest element size, makes the explicit codes lose out to implicit codes for some of the problems, especially for those of slow time variation.

3.7.2 Newmark's Method (Newmark, 1959)

Newmark's method is the most widely used implicit algorithm.The example software used in this book, ABAQUS, also uses the Newmark's method as its implicit solver except that the equilibrium equation defined in Eq. (3.106) is modified with the introduction of an operator defined by Hilber, Hughes and Taylor [1978]. In this book, we will introduce the standard Newmark's method as follows. It is first assumed that

$$\mathbf{D}_{t+\Delta t} = \mathbf{D}_t + (\Delta t)\dot{\mathbf{D}}_t + (\Delta t)^2\left[\left(\tfrac{1}{2}-\beta\right)\ddot{\mathbf{D}}_t + \beta\ddot{\mathbf{D}}_{t+\Delta t}\right] \tag{3.122}$$

$$\dot{\mathbf{D}}_{t+\Delta t} = \dot{\mathbf{D}}_t + (\Delta t)\left[(1-\gamma)\ddot{\mathbf{D}}_t + \gamma\ddot{\mathbf{D}}_{t+\Delta t}\right] \tag{3.123}$$

where β and γ are constants chosen by the analyst. Equations (3.122) and (3.123) are then substituted into the system equation (3.106) to give

$$\begin{aligned}&\mathbf{K}\left\{\mathbf{D}_t + (\Delta t)\dot{\mathbf{D}}_t + (\Delta t)^2\left[\left(\tfrac{1}{2}-\beta\right)\ddot{\mathbf{D}}_t + \beta\ddot{\mathbf{D}}_{t+\Delta t}\right]\right\}\\ &\quad + \mathbf{C}\left\{\dot{\mathbf{D}}_t + (\Delta t)\left[(1-\gamma)\ddot{\mathbf{D}}_t + \gamma\ddot{\mathbf{D}}_{t+\Delta t}\right]\right\} + \mathbf{M}\ddot{\mathbf{D}}_{t+\Delta t} = \mathbf{F}_{t+\Delta t}\end{aligned} \tag{3.124}$$

If we group all the terms involving $\ddot{\mathbf{D}}_{t+\Delta t}$ on the left and shift the remaining terms to the right, we can write

$$\mathbf{K}_{\mathrm{cm}}\ddot{\mathbf{D}}_{t+\Delta t} = \mathbf{F}_{t+\Delta t}^{\mathrm{residual}} \tag{3.125}$$

where

$$\mathbf{K}_{\mathrm{cm}} = \left[\mathbf{K}\beta(\Delta t)^2 + \mathbf{C}\gamma\Delta t + \mathbf{M}\right] \tag{3.126}$$

and

$$\begin{aligned}\mathbf{F}_{t+\Delta t}^{\mathrm{residual}} &= \mathbf{F}_{t+\Delta t} - \mathbf{K}\left\{\mathbf{D}_t + (\Delta t)\dot{\mathbf{D}}_t + (\Delta t)^2\left(\tfrac{1}{2}-\beta\right)\ddot{\mathbf{D}}_t\right\}\\ &\quad - \mathbf{C}\left\{\dot{\mathbf{D}}_t + (\Delta t)(1-\gamma)\ddot{\mathbf{D}}_t\right\}\end{aligned} \tag{3.127}$$

The accelerations $\ddot{\mathbf{D}}_{t+\Delta t}$ can then be obtained by solving matrix system equation (3.125):

$$\ddot{\mathbf{D}}_{t+\Delta t} = \mathbf{K}_{\mathrm{cm}}^{-1}\mathbf{F}_{t+\Delta t}^{\mathrm{residual}} \tag{3.128}$$

Note that the above equation involves matrix inversion, and hence it is analogous to solving a matrix equation. This makes it an implicit method.

The algorithm normally starts with a prescribed initial velocity and displacements, $\mathbf{D}_0$ and $\dot{\mathbf{D}}_0$. The initial acceleration $\ddot{\mathbf{D}}_0$ can then be obtained by substituting $\mathbf{D}_0$ and $\dot{\mathbf{D}}_0$ into Eq. (3.106), if $\ddot{\mathbf{D}}_0$ is not prescribed initially. Then Eq. (3.128) can be used to obtain the acceleration at the next time step, $\ddot{\mathbf{D}}_{\Delta t}$. The displacements $\mathbf{D}_{\Delta t}$ and velocities $\dot{\mathbf{D}}_{\Delta t}$ can then be calculated using Eqs. (3.122) and (3.123), respectively. The procedure then repeats to march forward in time until arriving at the final desired time. At each time step, the matrix system Eq. (3.125) has to be solved, which can be very time consuming, and leads to a very slow time stepping process.

Newmark's method, like most implicit methods, is *unconditionally stable* if $\gamma \geq 0.5$ and $\beta \geq (2\gamma + 1)^2/16$. Unconditionally stable methods are those in which the size of the time step, Δt, will not affect the stability of the solution, but rather it is governed by accuracy considerations. The unconditionally stable property allows the implicit algorithms to use significantly larger time steps when the external excitation is of a slow time variation.

3.8 REMARKS

3.8.1 Summary of Shape Function Properties

The properties of the shape functions are summarized in Table 3.1.

3.8.2 Sufficient Requirements for FEM Shape Functions

Properties 3 and 4 are the minimum requirements for shape functions workable for the FEM. In mesh free methods [Liu, 2002], Property 3 is not a necessary condition for shape functions. Property 5 is a sufficient requirement for shape functions workable for the FEM for solid mechanics problems. It is possible for shape functions that do not possess Property 5 to produce convergent FEM solutions. In this book, however, we generally require all the FEM shape functions to satisfy Properties 3, 4 and 5. These three requirements are called the sufficient requirements in this book for FEM shape functions; they are the *delta function property*, *partitions of unity*, and *linear field reproduction*.

3.8.3 Recap of FEM Procedure

In finite element methods, the displacement field **U** is expressed by displacements at *nodes* using *shape functions* defined over *elements*. Once the shape functions are found, the mass matrix and force vector can be obtained using Eqs. (3.75), (3.78), (3.79) and (3.81). The stiffness matrix can also be obtained using Eq. (3.71), once the shape functions and the strain matrix have been found. Therefore, to develop FE equations for any type of structure components, all one need do is formulate the shape function N and then establish the strain matrix **B**. The other procedures are very much the same. Hence, in the following chapters, the focus will be mainly on the derivation of the shape function and strain matrix for various types of solids and structural components.

Table 3.1. List of properties of shape functions

Item	Name	Significant
Property 1	Reproduction property and consistency	Ensures shape functions produce all the functions that can be formed using basis functions used to create the shape function. It is useful for constructing shape functions with desired accuracy and consistency in displacement field approximation.
Property 2	Shape functions are linearly independent	Ensures the shape functions have Delta function properties.
Property 3	**Delta function properties**	Facilitate an easy imposition of essential boundary conditions. This is a minimum requirement for shape functions workable for the FEM.
Property 4	**Partitions of unity property**	Ensures the shape functions to produce the rigid body movement. This is a minimum requirement for shape functions.
Property 5	**Linear field reproduction**	Ensures shape functions to produce the linear displacement field. This is a sufficient requirement for shape functions capable of passing the patch test, and hence that for the FEM workable for solid mechanics problems.
Lemma 1	Condition for shape functions being partitions of unity	Provides an alternative tool for checking the property of partitions of unity of shape functions.
Lemma 2	Condition for shape functions being partitions of unity	Provides an alternative tool for checking the property of partitions of unity of shape functions.
Lemma 3	Condition for shape functions being linear field reproduction	Provides an alternative tool for checking the property of linear field reproduction of shape functions.

Properties 3, 4 and 5 constitute the sufficient requirement for FEM shape functions

3.9 REVIEW QUESTIONS

1. What is the difference between the strong and weak forms of system equations?
2. What are the conditions that a summed displacement has to satisfy in order to apply Hamilton's principle?
3. Briefly describe the standard steps involved in the finite element method.
4. Do we have to discretize the problem domain in order to apply Hamilton's principle? What is the purpose of dividing the problem domain into elements?
5. For a function defined as

$$f(x) = a_0 + a_1 x + a_2 x^2$$

show that

(1) the variation operator and integral operator are exchangeable, i.e.

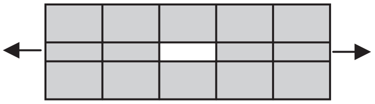
$$\int [\delta f(x)]\,\mathrm{d}x = \delta\left[\int f(x)\,\mathrm{d}x\right]$$

(2) the variation operator and the differential operator are exchangeable, i.e.

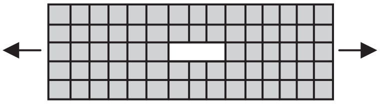
$$\delta\frac{\mathrm{d}f(x)}{\mathrm{d}x} = \frac{\mathrm{d}}{\mathrm{d}x}[\delta f(x)]$$

6. What are the properties of a shape function? Can we use shape functions that do not possess these properties?

7.



A



B

(a) Which mesh will yield more accurate results?
(b) Which will be more computationally expensive?
(c) Suggest a way of meshing which will yield relatively accurate results and, at the same time be less computationally expensive than B?

8. Why is there a need to perform coordinate transformation for each element?
9. Describe how element matrices can be assembled together to form the global system matrix.

4

FEM FOR TRUSSES

4.1 INTRODUCTION

A truss is one of the simplest and most widely used structural members. It is a straight *bar* that is designed to take only axial forces, therefore it deforms only in its axial direction. A typical example of its usage can be seen in Figure 2.7. The cross-section of the bar can be arbitrary, but the dimensions of the cross-section should be much smaller than that in the axial direction. Finite element equations for such truss members will be developed in this chapter. The element developed is commonly known as the *truss element* or *bar element*. Such elements are applicable for analysis of the skeletal type of truss structural systems both in two-dimensional planes and in three-dimensional space. The basic concepts, procedures and formulations can also be found in many existing textbooks (see, e.g. Reddy, 1993; Rao, 1999; Zienkiewicz and Taylor, 2000; etc.).

In planar trusses there are two components in the x and y directions for the displacements as well as for the forces. For space trusses, however, there will be three components in the x, y and z directions for both displacements and forces. In skeletal structures consisting of truss members, the truss members are joined together by pins or hinges (not by welding), so that there are only forces (not moments) transmitted between the bars. For the purpose of explaining the concepts more clearly, this book will assume that the truss elements have a uniform cross-section. Therefore, to deal with bars with varying cross-sections, one should develop equations for a truss element with a varying cross-section, which can also be done very easily following the procedure for uniform truss elements. Note that there is no reason from the mechanics viewpoint to use bars with a varying cross-section, as the force in a bar is uniform.

4.2 FEM EQUATIONS

4.2.1 Shape Function Construction

Consider a structure consisting of a number of trusses or bar members. Each of the members can be considered as a truss/bar element of uniform cross-section bounded by two nodes ($n_d = 2$). Consider a bar element with nodes 1 and 2 at each end of the element, as shown in Figure 4.1. The length of the element is l_e. The local x-axis is taken in the axial direction

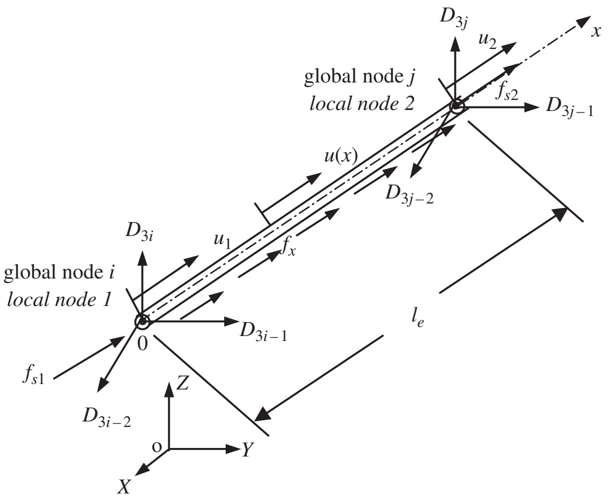


Figure 4.1. Truss element and the coordinate system.

of the element with the origin at node 1. In the local coordinate system, there is only one DOF at each node of the element, and that is the axial displacement. Therefore, there is a total of two DOFs for the element, i.e. $n_e = 2$. In the FEM discussed in the previous chapter, the displacement in an element should be written in the form

$$u^h(x) = \mathbf{N}(x)\mathbf{d}_e \tag{4.1}$$

where u^h is the *approximation* of the axial displacement within the element, $\mathbf{N}$ is a matrix of shape functions that possess the properties described in Chapter 3, and $\mathbf{d}_e$ should be the vector of the displacements at the two nodes of the element:

$$\mathbf{d}_e = \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} \tag{4.2}$$

The question now is, how can we determine the shape functions for the truss elements?

We follow the standard procedure described in Section 3.4.3 for constructing shape functions, and assume that the axial displacement in the truss element can be given in a general form

$$u^h(x) = \alpha_0 + \alpha_1 x = \underbrace{\begin{Bmatrix} 1 & x \end{Bmatrix}}_{\mathbf{p}^T} \underbrace{\begin{Bmatrix} \alpha_0 \\ \alpha_1 \end{Bmatrix}}_{\boldsymbol{\alpha}} = \mathbf{p}^T \boldsymbol{\alpha} \tag{4.3}$$

where u^h is the approximation of the displacement, α is the vector of two unknown constants, α_0 and α_1, and $\mathbf{p}$ is the vector of polynomial basis functions (or monomials). For this particular problem, we use up to the first order of polynomial basis. Depending upon the problem, we could use a higher order of polynomial basis functions. The order of polynomial basis functions up to the nth order can be given by

$$\mathbf{P}^T = \{1 \quad x \quad \cdots \quad x^n\} \tag{4.4}$$

The number of terms of basis functions or monomials we should use depends upon the number of nodes and degrees of freedom in the element. Since we have two nodes with a total of two DOFs in the element, we choose to have two terms of basis functions, which gives Eq. (4.3).

Note that we usually use polynomial basis functions complete of orders, meaning we do not skip any lower terms in constructing Eq. (4.3). This is to ensure that the shape functions constructed will be able to reproduce complete polynomials up to an order of n. If a polynomial basis of the kth order is skipped, the shape function constructed will only be able to ensure a consistency of $(k-1)$th order, regardless of how many higher orders of monomials are included in the basis. This is because of the consistency property of the shape function (Property 1), discussed in Section 3.4.4. From Properties 3, 4 and 5 discussed in Chapter 3, we can expect that the complete linear basis functions used in Eq. (4.3) guarantee that the shape function to be constructed satisfies the *sufficient requirements* for the FEM shape functions: the delta function property, partition of unity and linear field reproduction.

In deriving the shape function, we use the fact that

$$\begin{aligned} &\text{at } x = 0, \quad u(x=0) = u_1 \\ &\text{at } x = l_e, \quad u(x=l_e) = u_2 \end{aligned} \tag{4.5}$$

Using Eq. (4.3), we then have

$$\begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & l_e \end{bmatrix} \begin{Bmatrix} \alpha_0 \\ \alpha_1 \end{Bmatrix} \tag{4.6}$$

Solving the above equation for α, we have

$$\begin{Bmatrix} \alpha_0 \\ \alpha_1 \end{Bmatrix} = \begin{bmatrix} 1 & 0 \\ -1/l_e & 1/l_e \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} \tag{4.7}$$

Substituting the above equation into Eq. (4.3), we obtain

$$u(x) = \mathbf{P}^T\boldsymbol{\alpha} = \{1 \quad x\} \begin{bmatrix} 1 & 0 \\ -1/l_e & 1/l_e \end{bmatrix} \begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \underbrace{\begin{Bmatrix} \underbrace{1 - x/l_e}_{N_1(x)} & \underbrace{x/l_e}_{N_2(x)} \end{Bmatrix}}_{\mathbf{N}(x)} \underbrace{\begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix}}_{\mathbf{d}_e} = \mathbf{N}(x)\mathbf{d}_e \tag{4.8}$$

which is Eq. (4.1), that we wanted. The matrix of shape functions is then obtained in the form

$$\mathbf{N}(x) = [N_1(x) \quad N_2(x)] \tag{4.9}$$

where the shape functions for a truss element can be written as

$$\begin{aligned} N_1(x) &= 1 - \frac{x}{l_e} \\ N_2(x) &= \frac{x}{l_e} \end{aligned} \tag{4.10}$$

We obtained two shape functions because we have two DOFs in the truss elements.

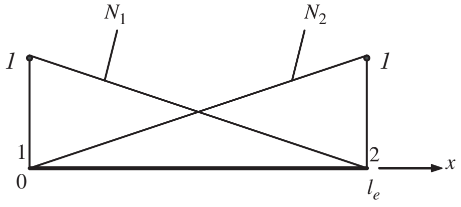


Figure 4.2. Linear shape functions.

It is easy to confirm that these two shape functions satisfy the delta function property defined by Eq. (3.34), and the partitions of unity in Eq. (3.41). We leave this confirmation to the reader as a simple exercise. The graphic representation of the linear shape functions is shown in Figure 4.2. It is clearly shown that N_i gives the *shape* of the contribution from nodal displacement at node i, and that is why it is called a shape function. In this case, the shape functions vary linearly across the element, and they are termed linear shape functions. Substituting Eqs. (4.9), (4.10) and (4.2) into Eq. (4.1), we have

$$u(x) = N_1(x)u_1 + N_2(x)u_2 = u_1 + \frac{u_2 - u_1}{l_e}x \tag{4.11}$$

which clearly states that the displacement within the element varies linearly. The element is therefore called a *linear element.*

4.2.2 Strain Matrix

As discussed in Chapter 2, there is only one stress component σ_x in a truss, and the corresponding strain can be obtained by

$$\varepsilon_x = \frac{\partial u}{\partial x} = \frac{u_2 - u_1}{l_e} \tag{4.12}$$

which is a direct result from differentiating Eq. (4.11) with respect to x. Note that the strain in Eq. (4.12) is a constant value in the element.

It was mentioned in the previous chapter that we would need to obtain the strain matrix, **B**, after which we can obtain the stiffness and mass matrices. In this case, this can be easily done. Equation (4.12) can be re-written in a matrix form as

$$\varepsilon_x = \frac{\partial u}{\partial x} = L\mathbf{N}\mathbf{d}_e = \mathbf{B}\mathbf{d}_e \tag{4.13}$$

where the strain matrix **B** has the following form for the truss element:

$$\mathbf{B} = L\mathbf{N} = \frac{\partial}{\partial x}\begin{bmatrix} 1 - x/l_e & x/l_e \end{bmatrix} = \begin{bmatrix} -1/l_e & 1/l_e \end{bmatrix} \tag{4.14}$$

4.2.3 Element Matrices in the Local Coordinate System

Once the strain matrix **B** has been obtained, the stiffness matrix for truss elements can be obtained using Eq. (3.71) in the previous chapter:

$$\mathbf{k}_e = \int_{V_e} \mathbf{B}^T \mathbf{c} \mathbf{B} \,\mathrm{d}V = A_e \int_0^{l_e} \begin{bmatrix} -1/l_e \\ 1/l_e \end{bmatrix} E \begin{bmatrix} -1/l_e & 1/l_e \end{bmatrix} \mathrm{d}x = \frac{AE}{l_e} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \qquad (4.15)$$

where A is the area of the cross-section of the truss element. Note that the material constant matrix **c** reduces to the elastic modulus, E, for the one-dimensional truss element (see Eq. (2.39)). It is noted that the element stiffness matrix as shown in Eq. (4.15) is symmetrical. This confirms the proof given in Eq. (3.73). Making use of the symmetry of the stiffness matrix, only half of the terms in the matrix need to be evaluated and stored during computation.

The mass matrix for truss elements can be obtained using Eq. (3.75):

$$\mathbf{m}_e = \int_{V_e} \rho \mathbf{N}^T \mathbf{N} \,\mathrm{d}V = A\rho l \int_0^{l_e} \begin{bmatrix} N_1 N_1 & N_1 N_2 \\ N_2 N_1 & N_2 N_2 \end{bmatrix} \mathrm{d}x = \frac{A\rho l_e}{6} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \qquad (4.16)$$

Similarly, the mass matrix is found to be symmetrical. The nodal force vector for truss elements can be obtained using Eqs. (3.78), (3.79) and (3.81). Suppose the element is loaded by an evenly distributed force f_x along the x-axis, and two concentrated forces f_{s1} and f_{s2}, respectively, at two nodes 1 and 2, as shown in Figure 4.1; the total nodal force vector becomes

$$\mathbf{f}_e = \int_{V_e} \mathbf{N}^T f_b \,\mathrm{d}V + \int_{S_e} \mathbf{N}^T f_s \,\mathrm{d}S = f_x \int_0^{l_e} \begin{bmatrix} N_1 \\ N_2 \end{bmatrix} \mathrm{d}x + \begin{Bmatrix} f_{s1} \\ f_{s2} \end{Bmatrix} = \begin{Bmatrix} f_x l_e/2 + f_{s1} \\ f_x l_e/2 + f_{s1} \end{Bmatrix} \qquad (4.17)$$

4.2.4 Element Matrices in the Global Coordinate System

Element matrices in Eqs. (4.15), (4.16) and (4.17) were formulated based on the *local coordinate system*, where the x-axis coincides with the mid axis of the bar 1–2, shown in Figure 4.1. In practical trusses, there are many bars of different orientations and at different locations. To assemble all the element matrices to form the global system matrices, a coordinate transformation has to be performed for each element to formulate its element matrix based on the *global coordinate system* for the whole truss structure. The following performs the transformation for both spatial and planar trusses.

Spatial trusses

Assume that the local nodes 1 and 2 of the element correspond to the global nodes i and j, respectively, as shown in Figure 4.1. The displacement at a global node in space should have three components in the X, Y and Z directions, and numbered sequentially. For example, these three components at the ith node are denoted by D_{3i-2}, D_{3i-1} and D_{3i}. The coordinate transformation gives the relationship between the displacement vector $\mathbf{d}_e$ based on the local coordinate system and the displacement vector $\mathbf{D}_e$ for the same element, but based on the global coordinate system *XYZ*:

$$\mathbf{d}_e = \mathbf{T}\mathbf{D}_e \tag{4.18}$$

where

$$\mathbf{D}_e = \begin{Bmatrix} D_{3i-2} \\ D_{3i-1} \\ D_{3i} \\ D_{3j-2} \\ D_{3j-1} \\ D_{3j} \end{Bmatrix} \tag{4.19}$$

and $\mathbf{T}$ is the transformation matrix for the truss element, given by

$$\mathbf{T} = \begin{bmatrix} l_{ij} & m_{ij} & n_{ij} & 0 & 0 & 0 \\ 0 & 0 & 0 & l_{ij} & m_{ij} & n_{ij} \end{bmatrix}_e \tag{4.20}$$

in which

$$\begin{aligned} l_{ij} &= \cos(x, X) = \frac{X_j - X_i}{l_e} \\ m_{ij} &= \cos(x, Y) = \frac{Y_j - Y_i}{l_e} \\ n_{ij} &= \cos(x, Z) = \frac{Z_j - Z_i}{l_e} \end{aligned} \tag{4.21}$$

are the *direction cosines* of the axial axis of the element. It is easy to confirm that

$$\mathbf{T}\mathbf{T}^T = \mathbf{I} \tag{4.22}$$

where $\mathbf{I}$ is an identity matrix of 2 × 2. Therefore, matrix $\mathbf{T}$ is an *orthogonal matrix.* The length of the element, l_e, can be calculated using the global coordinates of the two nodes of the element by

$$l_e = \sqrt{(X_j - X_i)^2 + (Y_j - Y_i)^2 + (Z_j - Z_i)^2} \tag{4.23}$$

Equation (4.18) can be easily verified, as it simply says that at node i, d_1 equals the summation of all the projections of D_{3i-2}, D_{3i-1} and D_{3i} onto the local x axis, and the same can be said for node j. The matrix $\mathbf{T}$ for a truss element transforms a 6 × 1 vector in the global coordinate system into a 2 × 1 vector in the local coordinate system.

The transformation matrix also applies to the force vectors between the local and global coordinate systems:

$$\mathbf{f}_e = \mathbf{T}\mathbf{F}_e \tag{4.24}$$

where

$$\mathbf{F}_e = \begin{Bmatrix} F_{3i-2} \\ F_{3i-1} \\ F_{3i} \\ F_{3j-2} \\ F_{3j-1} \\ F_{3j} \end{Bmatrix} \tag{4.25}$$

in which F_{3i-2}, F_{3i-1} and F_{3i} stand for the three components of the force vector at node i based on the global coordinate system.

Substitution of Eq. (4.18) into Eq. (3.89) leads to the element equation based on the global coordinate system:

$$\mathbf{k}_e\mathbf{T}\mathbf{D}_e + \mathbf{m}_e\mathbf{T}\ddot{\mathbf{D}}_e = \mathbf{f}_e \tag{4.26}$$

Pre-multiply $\mathbf{T}^T$ to both sides in the above equation to obtain:

$$(\mathbf{T}^T\mathbf{k}_e\mathbf{T})\mathbf{D}_e + (\mathbf{T}^T\mathbf{m}_e\mathbf{T})\ddot{\mathbf{D}}_e = \mathbf{T}^T\mathbf{f}_e \tag{4.27}$$

or

$$\mathbf{K}_e\mathbf{D}_e + \mathbf{M}_e\ddot{\mathbf{D}}_e = \mathbf{F}_e \tag{4.28}$$

where

$$\mathbf{K}_e = \mathbf{T}^T\mathbf{k}_e\mathbf{T}$$
$$= \frac{AE}{l_e}\begin{bmatrix} l_{ij}^2 & l_{ij}m_{ij} & l_{ij}n_{ij} & -l_{ij}^2 & -l_{ij}m_{ij} & -l_{ij}n_{ij} \\ l_{ij}m_{ij} & m_{ij}^2 & m_{ij}n_{ij} & -l_{ij}m_{ij} & -m_{ij}^2 & -m_{ij}n_{ij} \\ l_{ij}n_{ij} & m_{ij}n_{ij} & n_{ij}^2 & -l_{ij}n_{ij} & -m_{ij}n_{ij} & -n_{ij}^2 \\ -l_{ij}^2 & -l_{ij}m_{ij} & -l_{ij}n_{ij} & l_{ij}^2 & l_{ij}m_{ij} & l_{ij}n_{ij} \\ -l_{ij}m_{ij} & -m_{ij}^2 & -m_{ij}n_{ij} & l_{ij}m_{ij} & m_{ij}^2 & m_{ij}n_{ij} \\ -l_{ij}n_{ij} & -m_{ij}n_{ij} & -n_{ij}^2 & l_{ij}n_{ij} & m_{ij}n_{ij} & n_{ij}^2 \end{bmatrix} \tag{4.29}$$

and

$$\mathbf{M}_e = \mathbf{T}^T\mathbf{m}_e\mathbf{T}$$
$$= \frac{A\rho l_e}{6}\begin{bmatrix} 2l_{ij}^2 & 2l_{ij}m_{ij} & 2l_{ij}n_{ij} & l_{ij}^2 & l_{ij}m_{ij} & l_{ij}n_{ij} \\ 2l_{ij}m_{ij} & 2m_{ij}^2 & 2m_{ij}n_{ij} & l_{ij}m_{ij} & m_{ij}^2 & m_{ij}n_{ij} \\ 2l_{ij}n_{ij} & 2m_{ij}n_{ij} & 2n_{ij}^2 & l_{ij}n_{ij} & m_{ij}n_{ij} & n_{ij}^2 \\ l_{ij}^2 & l_{ij}m_{ij} & l_{ij}n_{ij} & 2l_{ij}^2 & 2l_{ij}m_{ij} & 2l_{ij}n_{ij} \\ l_{ij}m_{ij} & m_{ij}^2 & m_{ij}n_{ij} & 2l_{ij}m_{ij} & 2m_{ij}^2 & 2m_{ij}n_{ij} \\ l_{ij}n_{ij} & m_{ij}n_{ij} & n_{ij}^2 & 2l_{ij}n_{ij} & 2m_{ij}n_{ij} & 2n_{ij}^2 \end{bmatrix} \tag{4.30}$$

Note that the coordinate transformation conserves the symmetrical properties of both stiffness and mass matrices.

For the forces given in Eq. (4.17), we have

$$\mathbf{F}_e = \mathbf{T}^T \mathbf{f}_e$$
$$= \begin{Bmatrix} (f_x l_e/2 + f_{s1})\, l_{ij} \\ (f_x l_e/2 + f_{s1})\, m_{ij} \\ (f_x l_e/2 + f_{s1})\, n_{ij} \\ \left(f_y l_e/2 + f_{s1}\right) l_{ij} \\ \left(f_y l_e/2 + f_{s1}\right) m_{ij} \\ \left(f_y l_e/2 + f_{s1}\right) n_{ij} \end{Bmatrix} \tag{4.31}$$

Note that the element stiffness matrix $\mathbf{K}_e$ and mass matrix $\mathbf{M}_e$ have a dimension of 6 × 6 in the three-dimensional global coordinate system, and the displacement $\mathbf{D}_e$ and the force vector $\mathbf{F}_e$ have a dimension of 6 × 1.

Planar trusses

For a planar truss, the global coordinates X–Y can be employed to represent the plane of the truss. All the formulations of coordinate transformation can be obtained from the counterpart of those for spatial trusses by simply removing the rows and/or columns corresponding to the z- (or Z-) axis. The displacement at the global node i should have two components in the X and Y directions only: D_{2i-1} and D_{2i}. The coordinate transformation, which gives the relationship between the displacement vector $\mathbf{d}_e$ based on the local coordinate system and the displacement vector $\mathbf{D}_e$, has the same form as Eq. (4.18), except that

$$\mathbf{D}_e = \begin{Bmatrix} D_{2i-1} \\ D_{2i} \\ D_{2j-1} \\ D_{2j} \end{Bmatrix} \tag{4.32}$$

and the transformation matrix $\mathbf{T}$ is given by

$$\mathbf{T} = \begin{bmatrix} l_{ij} & m_{ij} & 0 & 0 \\ 0 & 0 & l_{ij} & m_{ij} \end{bmatrix} \tag{4.33}$$

The force vector in the global coordinate system is

$$\mathbf{F}_e = \begin{Bmatrix} F_{2i-1} \\ F_{2i} \\ F_{2j-1} \\ F_{2j} \end{Bmatrix} \tag{4.34}$$

All the other equations for a planar truss have the same form as the corresponding equations for a space truss. The $\mathbf{K}_e$ and $\mathbf{M}_e$ for the planar truss have a dimension of 4 × 4 in the global coordinate system. They are listed as follows:

$$\mathbf{K}_e = \mathbf{T}^T\mathbf{k}_e\mathbf{T} = \begin{bmatrix} K_{11}^e & K_{12}^e & K_{13}^e & K_{14}^e \\ K_{12}^e & K_{22}^e & K_{23}^e & K_{24}^e \\ K_{13}^e & K_{23}^e & K_{33}^e & K_{34}^e \\ K_{14}^e & K_{24}^e & K_{34}^e & K_{44}^e \end{bmatrix}$$

$$= \frac{AE}{l_e}\begin{bmatrix} l_{ij}^2 & l_{ij}m_{ij} & -l_{ij}^2 & -l_{ij}m_{ij} \\ l_{ij}m_{ij} & m_{ij}^2 & -l_{ij}m_{ij} & -m_{ij}^2 \\ -l_{ij}^2 & -l_{ij}m_{ij} & l_{ij}^2 & l_{ij}m_{ij} \\ -l_{ij}m_{ij} & -m_{ij}^2 & l_{ij}m_{ij} & m_{ij}^2 \end{bmatrix} \tag{4.35}$$

$$\mathbf{M}_e = \mathbf{T}^T\mathbf{m}_e\mathbf{T} = \begin{bmatrix} M_{11}^e & M_{12}^e & M_{13}^e & M_{14}^e \\ M_{12}^e & M_{22}^e & M_{23}^e & M_{24}^e \\ M_{13}^e & M_{23}^e & M_{33}^e & M_{34}^e \\ M_{14}^e & M_{24}^e & M_{34}^e & M_{44}^e \end{bmatrix}$$

$$= \frac{A\rho l_e}{6}\begin{bmatrix} 2l_{ij}^2 & 2l_{ij}m_{ij} & l_{ij}^2 & l_{ij}m_{ij} \\ 2l_{ij}m_{ij} & 2m_{ij}^2 & l_{ij}m_{ij} & m_{ij}^2 \\ l_{ij}^2 & l_{ij}m_{ij} & 2l_{ij}^2 & 2l_{ij}m_{ij} \\ l_{ij}m_{ij} & m_{ij}^2 & 2l_{ij}m_{ij} & 2m_{ij}^2 \end{bmatrix} \tag{4.36}$$

4.2.5 Boundary Conditions

The stiffness matrix $\mathbf{K}_e$ in Eq. (4.28) is usually singular, because the whole structure can perform rigid body movements. There are two DOFs of rigid movement for planer trusses and three DOFs for space trusses. These rigid body movements are constrained by supports or displacement constraints. In practice, truss structures are fixed somehow to the ground or to a fixed main structure at a number of the nodes. When a node is fixed, the displacement at the node must be zero. This fixed displacement boundary condition can be imposed on Eq. (4.28). The imposition leads to a cancellation of the corresponding rows and columns in the stiffness matrix. The reduced stiffness matrix becomes Symmetric Positive Definite (SPD), if sufficient displacements are constrained.

4.2.6 Recovering Stress and Strain

Equation (4.28) can be solved using standard routines and the displacements at all the nodes can be obtained after imposing the boundary conditions. The displacements at any position other than the nodal positions can also be obtained using interpolation by the shape functions. The stress in a truss element can also be recovered using the following equation:

$$\sigma_x = E\mathbf{B}\mathbf{d}_e = E\mathbf{B}\mathbf{T}\mathbf{D}_e \tag{4.37}$$

In deriving the above equation, Hooke's law in the form of $\sigma = E\varepsilon$ is used, together with Eqs. (4.13) and (4.18).

4.3 WORKED EXAMPLES

Example 4.1: A uniform bar subjected to an axial force

Consider a bar of uniform cross-sectional area, shown in Figure 4.3. The bar is fixed at one end and is subjected to a horizontal load of P at the free end. The dimensions of the bar are shown in the figure, and the beam is made of an isotropic material with Young's modulus E.

Exact solution

We first derive the exact solution, as this problem is very simple. From the strong form of governing equation (2.43), we have

$$\frac{\partial^2 u}{\partial x^2} = 0 \tag{4.38}$$

Note that, for the current example problem, the bar is free of body forces and hence $f_x = 0$. The general form of solution for Eq. (4.38) can be obtained very easily as

$$u(x) = c_0 + c_1 x \tag{4.39}$$

where c_0 and c_1 are unknown constants to be determined by boundary conditions. The displacement boundary condition for this problem can be given as

$$u = 0, \quad \text{at } x = 0 \tag{4.40}$$

Therefore, we have $c_0 = 0$. Equation (4.39) now becomes

$$u(x) = c_1 x \tag{4.41}$$

Using Eqs. (2.38), (2.39) and (4.41), we obtain

$$\sigma_x = E\frac{\partial u}{\partial x} = c_1 E \tag{4.42}$$

The force boundary condition for this bar can be given as

$$\sigma_x = \frac{P}{A}, \quad \text{at } x = l \tag{4.43}$$

Equating the right-hand side of Eqs. (4.42) and (4.43), we obtain

$$c_1 = \frac{P}{EA} \tag{4.44}$$

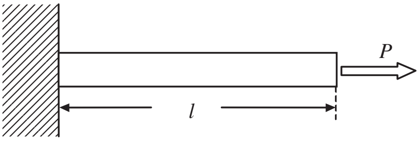


Figure 4.3. Clamped bar under static load.

The stress in the bar is obtained by substituting Eq. (4.44) back into Eq. (4.42), i.e.

$$\sigma_x = \frac{P}{A} \tag{4.45}$$

Substituting Eq. (4.44) back into Eq. (4.41), we finally obtain the solution of the displacement of the bar:

$$u(x) = \frac{P}{EA}x \tag{4.46}$$

At $x = l$, we have

$$u(x = l) = \frac{Pl}{EA} \tag{4.47}$$

FEM solution

Using one element, the bar is modelled as shown in Figure 4.4. Using Eq. (4.15), the stiffness matrix of the bars is given by

$$\mathbf{K} = \mathbf{k}_e = \frac{AE}{l}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{4.48}$$

There is no need to perform coordinate transformation, as the local and global coordinate systems are the same. There is also no need to perform assembly, because there is only one element. The finite element equation becomes

$$\frac{AE}{l}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \begin{Bmatrix} F_1 =? \\ F_2 = P \end{Bmatrix} \tag{4.49}$$

where F_1 is the reaction force applied at node 1, which is unknown at this stage. Instead, what we know is the displacement boundary condition Eq. (4.40) at node1. We can then simply remove the first equation in Eq. (4.49), i.e.

$$\frac{AE}{l}\begin{bmatrix} \cancel{1} & \cancel{-1} \\ \cancel{-1} & 1 \end{bmatrix}\begin{Bmatrix} \cancel{u_1} \\ u_2 \end{Bmatrix} = \begin{Bmatrix} \cancel{F_1 =?} \\ F_2 = P \end{Bmatrix} \tag{4.50}$$

which leads to

$$u_2 = \frac{Pl}{AE} \tag{4.51}$$

This is the finite element solution of the bar, which is exactly the same as the exact solution obtained in Eq. (4.47). The distribution of the displacement in the bar can be obtained by

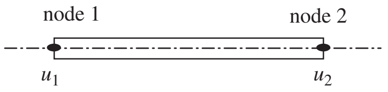


Figure 4.4. One truss element is used to model the clamped bar under static load.

substituting Eqs. (4.40) and (4.51) into Eq. (4.1),

$$u(x) = \mathbf{N}(x)\mathbf{d}_e = \left\{1 - x/l \quad x/l\right\}\begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \left\{1 - x/l \quad x/l\right\}\begin{Bmatrix} 0 \\ Pl/(EA) \end{Bmatrix} = \frac{P}{EA}x \tag{4.52}$$

which is also exactly the same as the exact solution obtained in Eq. (4.46).

Using Eqs. (4.37) and (4.14), we obtain the stress in the bar

$$\sigma_x = E\mathbf{B}\mathbf{d}_e = E\begin{bmatrix} -\frac{1}{l} & \frac{1}{l} \end{bmatrix}\begin{Bmatrix} 0 \\ u_2 \end{Bmatrix} = \frac{P}{A} \tag{4.53}$$

which is again exactly the same as the exact solution given in Eq. (4.45).

4.3.1 Properties of the FEM

Reproduction property of the FEM

Using the FEM, one can usually expect only an approximated solution. In Example 4.1, however, we obtained the exact solution. Why? This is because the exact solution of the deformation for the bar is a first order polynomial (see Eq. (4.46)). The shape functions used in our FEM analysis are also first order polynomials that are constructed using complete monomials up to the first order. Therefore, the exact solution of the problem is included in the set of assumed displacements in FEM shape functions. In Chapter 3, we understand that the FEM based on Hamilton's principle guarantees to choose the best possible solution that can be produced by the shape functions. In Example 4.1, the best possible solution that can be produced by the shape function is the exact solution, due to the reproduction property of the shape functions, and the FEM has indeed reproduced it exactly. We therefore confirmed the *reproduction* property of the FEM that if the exact solution can be formed by the basis functions used to construct the FEM shape function, the FEM will always produce the exact solution, provided there is no numerical error involved in computation of the FEM solution.

Making use of this property, one may try to add in basis functions that form the exact solution or part of the exact solution, if that is possible, so as to achieve better accuracy in the FEM solution.

Convergence property of the FEM

For complex problems, the solution cannot be written in the form of a combination of monomials. Therefore, the FEM using polynomial shape functions will not produce the exact solution for such a problem. The question now is, how can one ensure that the FEM can produce a *good* approximation of the solution of a complex problem? The insurance is given by the *convergence* property of the FEM, which states that the FEM solution will converge to the exact solution that is continuous at arbitrary accuracy when the element size becomes infinitely small, and as long as the complete linear polynomial basis is included in the basis to form the FEM shape functions. The theoretical background for this convergence feature of the FEM is due to the fact that any continuous function can always be approximated by a first order polynomial with a second order of refinement error. This fact can be revealed by using the local Taylor expansion, based on which a continuous (displacement) function $u(x)$ can always be approximated using the following equation:

$$u = u_i + \left.\frac{\partial u}{\partial x}\right|_i (x - x_i) + O(h^2) \tag{4.54}$$

where h is the characteristic size that relates to $(x - x_i)$, or the size of the element.

One may argue that the use of a constant can also reproduce the function u, but with an accuracy of $O(h^1)$, according to Eq. (4.54). However, the constant displacements produced by the elements will possibly not be continuous in between elements, unless the entire displacement field is constant (rigid movement), which is trivial. Therefore, to guarantee the convergence of a continuous solution, a complete polynomial up to at least the first order is used.

Rate of convergence of FEM results

The Taylor expansion up to the order of p can be given as

$$u = u_i + \left.\frac{\partial u}{\partial x}\right|_i (x - x_i) + \frac{1}{2!}\left.\frac{\partial^2 u}{\partial x^2}\right|_i (x - x_i)^2 + \cdots + \frac{1}{p!}\left.\frac{\partial^p u}{\partial x^p}\right|_i (x - x_i)^p + O(h^{p+1}) \tag{4.55}$$

If the complete polynomials up to the pth order are used for constructing the shape functions, the first $(p+1)$ terms in Eq. (4.55) will be reproduced by the FEM shape function. The error is of the order of $O(h^{p+1})$; the order of the rate of convergence is therefore $O(h^{p+1})$. For linear elements we have $p = 1$, and the order of the rate of convergence for the displacement is therefore $O(h^2)$. This implies that if the element sized is halved, the error of the results in displacement will be reduced by a rate of one quarter.

These properties of the FEM, reproduction and convergence, are the key for the FEM to provide reliable numerical results for mechanics problems, because we are assured as to what kind of results we are going to get. For simple problems whose exact solutions are of polynomial types, the FEM is capable of reproducing the exact solution using a minimum number of elements, as long as complete order of basis functions, including the order of the exact solution, is used. In Example 4.1, one element of first order is sufficient. For complex problems whose exact solution is of a very high order of polynomial type, or often a non-polynomial type, it is then up to the analyst to use a proper density of the element mesh to obtain FEM results of desired accuracy with a convergence rate of $O(h^{p+1})$ for the displacements.

As an extension of this discussion, we mention the concepts of so-called h-adaptivity and p-adaptivity that are intensively used in the recent development of FEM analyses. We conventionally use h to present the characteristic size of the elements, and p to represent the order of the polynomial basis functions. h-adaptive analysis uses finer element meshes (smaller h), and p-adaptivity analysis uses a higher order of shape functions (large p) to achieve the desired accuracy of FEM results.

Example 4.2: A triangular truss structure subjected to a vertical force

Consider the plane truss structure shown in Figure 4.5. The structure is made of three planar truss members as shown, and a vertical downward force of 1000 N is applied at node 2. The

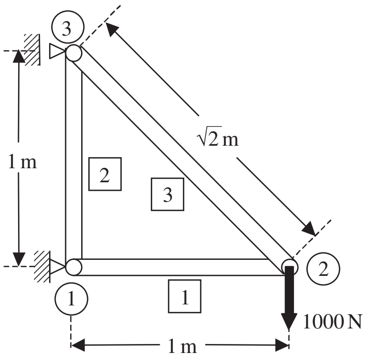


Figure 4.5. A three member truss structure.

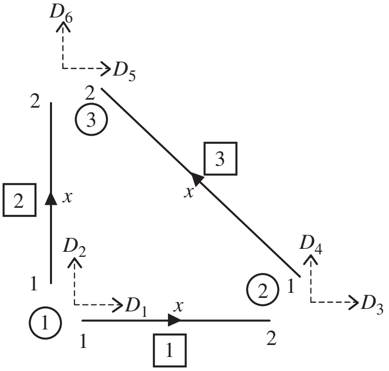


Figure 4.6. Local coordinates and degrees of freedom of truss elements.

figure also shows the numbering of the elements used (labelled in squares), as well as the numbering of the nodes (labelled in circles).

The local coordinates of the three truss elements are shown in Figure 4.6. The figure also shows the numbering of the global degrees of freedom, D_1, D_2, ..., D_6, corresponding to the three nodes in the structure. Note that there are six global degrees of freedom altogether, with each node having two degrees of freedom in the X and Y directions. However, there is actually only one degree of freedom in each node in the local coordinate system for each element. From the figure, it is shown clearly that the degrees of freedom at each node have contributions from more than one element. For example, at node 1, the global degrees of freedom D_1 and D_2 have a contribution from elements 1 and 2. These will play an important role in the assembly of the final finite element matrices. Table 4.1 shows the dimensions and material properties of the truss members in the structure.

Table 4.1. Dimensions and properties of truss members

Element number	Cross-sectional area, A_e m^2	Length l_e m	Young's modulus E N/m^2
1	0.1	1	70×10^9
2	0.1	1	70×10^9
3	0.1	$\sqrt{2}$	70×10^9

Table 4.2. Global coordinates of nodes and direction cosines of elements

Element number	Global node corresponding to		Coordinates in global coordinate system		Direction cosines	
	Local node 1 (i)	Local node 2 (j)	X_i, Y_i	X_j, Y_j	l_{ij}	m_{ij}
1	1	2	0, 0	1, 0	1	0
2	1	3	0, 0	0, 1	0	1
3	2	3	1, 0	0, 1	$-1/\sqrt{2}$	$1/\sqrt{2}$

Step 1: Obtaining the direction cosines of the elements Knowing the coordinates of the nodes in the global coordinate system, the first step would be to take into account the orientation of the elements with respect to the global coordinate system. This can be done by computing the direction cosines using Eq. (4.21). Since this problem is a planar problem, there is no need to compute n_{ij}. The coordinates of all the nodes and the direction cosines of l_{ij} and m_{ij} are shown in Table 4.2.

Step 2: Calculation of element matrices in global coordinate system After obtaining the direction cosines, the element matrices in the global coordinate system can be obtained. Note that the problem here is a static problem, hence the element mass matrices need not be computed. What is required is thus the stiffness matrix. Recall that the element stiffness matrix in the local coordinate system is a 2 × 2 matrix, since the total degrees of freedom is two for each element. However, in the transformation to the global coordinate system, the degrees of freedom for each element becomes four, therefore the stiffness matrix in the global coordinate system is a 4 × 4 matrix. The stiffness matrices can be computed using Eq. (4.35), and is shown below:

$$\mathbf{K}^{e1} = \frac{(0.1)(70 \times 10^9)}{1.0}\begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 7 & 0 & -7 & 0 \\ 0 & 0 & 0 & 0 \\ -7 & 0 & 7 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \times 10^9\,\text{Nm}^{-2} \qquad (4.56)$$

$$\mathbf{K}^{e2} = \frac{(0.1)(70\times 10^9)}{1}\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 7 & 0 & -7 \\ 0 & 0 & 0 & 0 \\ 0 & -7 & 0 & 7 \end{bmatrix} \times 10^9\,\text{Nm}^{-2} \tag{4.57}$$

$$\mathbf{K}^{e3} = \frac{(0.1)(70\times 10^9)}{\sqrt{2}}\begin{bmatrix} 1/2 & -1/2 & -1/2 & 1/2 \\ -1/2 & 1/2 & 1/2 & -1/2 \\ -1/2 & 1/2 & 1/2 & -1/2 \\ 1/2 & -1/2 & -1/2 & 1/2 \end{bmatrix}$$

$$= \begin{bmatrix} 7/2\sqrt{2} & -7/2\sqrt{2} & -7/2\sqrt{2} & 7/2\sqrt{2} \\ -7/2\sqrt{2} & 7/2\sqrt{2} & 7/2\sqrt{2} & -7/2\sqrt{2} \\ -7/2\sqrt{2} & 7/2\sqrt{2} & 7/2\sqrt{2} & -7/2\sqrt{2} \\ 7/2\sqrt{2} & -7/2\sqrt{2} & -7/2\sqrt{2} & 7/2\sqrt{2} \end{bmatrix} \times 10^9\,\text{Nm}^{-2} \tag{4.58}$$

Step 3: Assembly of global FE matrices The next step after getting the element matrices will be to assemble the element matrices into a global finite element matrix. Since the total global degrees of freedom in the structure is six, the global stiffness matrix will be a 6 × 6 matrix. The assembly is done by adding up the contributions for each node by the elements that share the node. For example, looking at Figure 4.6, it can be seen that element 1 contributes to the degrees of freedom D_1 and D_2 at node 1, and also to the degrees of freedom D_3 and D_4 at node 2. On the other hand, element 2 also contributes to degrees of freedom D_1 and D_2 at node 1, and also to D_5 and D_6 at node 3. By adding the contributions from the individual element matrices into the respective positions in the global matrix according the contributions to the degrees of freedom, the global matrix can be obtained. This assembly process is termed *direct assembly*.

At the beginning of the assembly, the entire global stiffness matrix is zeroed. By adding the element matrix for element $\boxed{1}$ into the global element, we have

$$\mathbf{K} = 10^9 \times \begin{array}{c} \\ \\ \left[\begin{array}{cccccc} \boxed{7} & \boxed{0} & \boxed{-7} & \boxed{0} & 0 & 0 \\ \boxed{0} & \boxed{0} & \boxed{0} & \boxed{0} & 0 & 0 \\ \boxed{-7} & \boxed{0} & \boxed{7} & \boxed{0} & 0 & 0 \\ \boxed{0} & \boxed{0} & \boxed{0} & \boxed{0} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \end{array}$$

$$\begin{array}{cccccc|c} D_1 & D_2 & D_3 & D_4 & & & \\ \uparrow & \uparrow & \uparrow & \uparrow & & & \\ \boxed{7} & \boxed{0} & \boxed{-7} & \boxed{0} & 0 & 0 & \rightarrow D_1 \\ \boxed{0} & \boxed{0} & \boxed{0} & \boxed{0} & 0 & 0 & \rightarrow D_2 \\ \boxed{-7} & \boxed{0} & \boxed{7} & \boxed{0} & 0 & 0 & \rightarrow D_3 \\ \boxed{0} & \boxed{0} & \boxed{0} & \boxed{0} & 0 & 0 & \rightarrow D_4 \\ 0 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 0 & 0 & 0 & 0 & \end{array} \tag{4.59}$$

Note that element 1 contributes to DOFs of D_1 to D_4. By adding the element matrix for element $\boxed{2}$ on top of the new global element, it becomes

$$\mathbf{K} = 10^9 \times \begin{array}{cc} \begin{array}{cccccc} D_1 & D_2 & & & D_5 & D_6 \\ \uparrow & \uparrow & & & \uparrow & \uparrow \end{array} & \\ \left[\begin{array}{cccccc} 7+\boxed{0} & 0+\boxed{0} & -7 & 0 & \boxed{0} & \boxed{0} \\ 0+\boxed{0} & 0+\boxed{7} & 0 & 0 & \boxed{0} & \boxed{-7} \\ -7 & 0 & 7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \boxed{0} & \boxed{0} & 0 & 0 & \boxed{0} & \boxed{0} \\ \boxed{0} & \boxed{-7} & 0 & 0 & \boxed{0} & \boxed{7} \end{array}\right] & \begin{array}{l} \to D_1 \\ \to D_2 \\ \\ \\ \to D_5 \\ \to D_6 \end{array} \end{array} \tag{4.60}$$

Element $\boxed{2}$ contributes to DOFs of D_1, D_2, D_5 and D_6. Finally, by adding the element matrix for element $\boxed{3}$ on top of the current global element, we obtain

$$\mathbf{K} = 10^9 \times \begin{array}{cc} \begin{array}{cccccc} & & D_3 & D_4 & D_5 & D_6 \\ & & \uparrow & \uparrow & \uparrow & \uparrow \end{array} & \\ \left[\begin{array}{cccccc} 7 & 0 & -7 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 & 0 & -7 \\ -7 & 0 & 7+\boxed{7/2\sqrt{2}} & \boxed{-7/2\sqrt{2}} & \boxed{-7/2\sqrt{2}} & \boxed{7/2\sqrt{2}} \\ 0 & 0 & \boxed{-7/2\sqrt{2}} & \boxed{7/2\sqrt{2}} & \boxed{7/2\sqrt{2}} & \boxed{-7/2\sqrt{2}} \\ 0 & 0 & \boxed{-7/2\sqrt{2}} & \boxed{7/2\sqrt{2}} & \boxed{7/2\sqrt{2}} & \boxed{-7/2\sqrt{2}} \\ 0 & -7 & \boxed{7/2\sqrt{2}} & \boxed{-7/2\sqrt{2}} & \boxed{-7/2\sqrt{2}} & 7+\boxed{7/2\sqrt{2}} \end{array}\right] & \begin{array}{l} \\ \\ \to D_3 \\ \to D_4 \\ \to D_5 \\ \to D_6 \end{array} \end{array} \tag{4.61}$$

Element $\boxed{3}$ contributes to DOFs of D_3 to D_6. In summary, we have the final global stiffness matrix:

$$\mathbf{K} = 10^9 \times \begin{array}{cc} \begin{array}{cccccc} D_1 & D_2 & D_3 & D_4 & D_5 & D_5 \\ \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \end{array} & \\ \left[\begin{array}{cccccc} 7 & 0 & -7 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 & 0 & -7 \\ -7 & 0 & 7+7/2\sqrt{2} & -7/2\sqrt{2} & -7/2\sqrt{2} & 7/2\sqrt{2} \\ 0 & 0 & -7/2\sqrt{2} & 7/2\sqrt{2} & 7/2\sqrt{2} & -7/2\sqrt{2} \\ 0 & 0 & -7/2\sqrt{2} & 7/2\sqrt{2} & 7/2\sqrt{2} & -7/2\sqrt{2} \\ 0 & -7 & 7/2\sqrt{2} & -7/2\sqrt{2} & -7/2\sqrt{2} & 7+7/2\sqrt{2} \end{array}\right] & \begin{array}{l} \to D_1 \\ \to D_2 \\ \to D_3 \\ \to D_4 \\ \to D_5 \\ \to D_6 \end{array} \end{array} \tag{4.62}$$

The direct assembly process shown above is very simple, and can be coded in a computer program very easily. All one needs to do is add entries of the element matrix to the corresponding entries in the global stiffness matrix. The correspondence is usually facilitated using a so-called index that gives the relation between the element number and the global nodal numbers.

One may now ask how one can simply add up element matrices into the global matrix like this, and prove that this will indeed lead to the global stiffness matrix. The answer is that we can, and the proof can be performed simply using the global equilibrium conditions at all of these nodes in the entire problem domain. The following gives a simple proof.

We choose to prove the assembled result of the entries of the third row in the global stiffness matrix given in Eq. (4.62). The proof process applies exactly to all other rows. For the third row of the equation, we consider the equilibrium of forces in the x-direction at node 2 of elements 1 and 3, which corresponds to the third global DOF of the truss structure that links elements 1 and 3. For static problems, the FE equation for element 1 can be written in the following general form (in the global coordinate system):

$$\begin{bmatrix} K_{11}^{e1} & K_{12}^{e1} & K_{13}^{e1} & K_{14}^{e1} \\ K_{12}^{e1} & K_{22}^{e1} & K_{23}^{e1} & K_{24}^{e1} \\ K_{13}^{e1} & K_{23}^{e1} & K_{33}^{e1} & K_{34}^{e1} \\ K_{14}^{e1} & K_{24}^{e1} & K_{34}^{e1} & K_{44}^{e1} \end{bmatrix} \begin{Bmatrix} D_1 \\ D_2 \\ D_3 \\ D_4 \end{Bmatrix} = \begin{Bmatrix} F_1^{e1} \\ F_2^{e1} \\ F_3^{e1} \\ F_4^{e1} \end{Bmatrix} \tag{4.63}$$

The third equation of Eq. (4.63), which corresponds to the third global DOF, is

$$K_{13}^{e1} D_1 + K_{23}^{e1} D_2 + K_{33}^{e1} D_3 + K_{34}^{e1} D_4 = F_3^{e1} \tag{4.64}$$

The FE equation for element 3 can be written in the following general form:

$$\begin{bmatrix} K_{11}^{e3} & K_{12}^{e3} & K_{13}^{e3} & K_{14}^{e3} \\ K_{12}^{e3} & K_{22}^{e3} & K_{23}^{e3} & K_{24}^{e3} \\ K_{13}^{e3} & K_{23}^{e3} & K_{33}^{e3} & K_{34}^{e3} \\ K_{14}^{e3} & K_{24}^{e3} & K_{34}^{e3} & K_{44}^{e3} \end{bmatrix} \begin{Bmatrix} D_3 \\ D_4 \\ D_5 \\ D_6 \end{Bmatrix} = \begin{Bmatrix} F_3^{e3} \\ F_4^{e3} \\ F_5^{e3} \\ F_6^{e3} \end{Bmatrix} \tag{4.65}$$

The first equation of Eq. (4.65), which corresponds to the third global DOF, is

$$K_{13}^{e3} D_3 + K_{23}^{e3} D_4 + K_{33}^{e3} D_5 + K_{34}^{e3} D_6 = F_3^{e3} \tag{4.66}$$

Forces in the x-direction applied at node 2 consist of element force F_3^{e3} from element 1 and F_3^{e3} from element 3, and the possible external force F_3. All these forces have to satisfy the following equilibrium equation:

$$F_3^{e1} + F_3^{e3} = F_3 \tag{4.67}$$

Substitution of Eqs. (4.64) and (4.66) into the foregoing equation leads to

$$K_{13}^{e1} D_1 + K_{23}^{e1} D_2 + \left(K_{33}^{e1} + K_{11}^{e3}\right) D_3 + \left(K_{34}^{e1} + K_{12}^{e3}\right) D_4 + K_{13}^{e3} D_5 + K_{14}^{e3} D_6 = F_3 \tag{4.68}$$

This confirms that the coefficients on the left-hand side of the above equations are the entries for the third row of the global stiffness matrix given in Eq. (4.62). The above proof process is also valid for all the other rows of entries in the global stiffness matrix.

Step 4: Applying boundary conditions The global matrix can normally be reduced in size after applying boundary conditions. In this case, D_1, D_2 and D_5 are constrained, and thus

$$D_1 = D_2 = D_5 = 0\ \mathrm{m} \tag{4.69}$$

This implies that the first, second and fifth rows and columns will actually have no effect on the solving of the matrix equation. Hence, we can simply remove the corresponding rows and columns:

$$\mathbf{K} = 10^9 \times \begin{array}{cc} \begin{array}{cccccc} \cancel{D_1} & \cancel{D_2} & D_3 & D_4 & \cancel{D_5} & D_5 \\ \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \end{array} & \\ \begin{bmatrix} \cancel{7} & \cancel{0} & \cancel{-7} & \cancel{0} & \cancel{0} & \cancel{0} \\ \cancel{0} & \cancel{7} & \cancel{0} & \cancel{0} & \cancel{0} & \cancel{-7} \\ \cancel{-7} & \cancel{0} & 7+7/2\sqrt{2} & -7/2\sqrt{2} & \cancel{-7/2\sqrt{2}} & 7/2\sqrt{2} \\ \cancel{0} & \cancel{0} & -7/2\sqrt{2} & 7/2\sqrt{2} & \cancel{7/2\sqrt{2}} & -7/2\sqrt{2} \\ \cancel{0} & \cancel{0} & \cancel{-7/2\sqrt{2}} & \cancel{7/2\sqrt{2}} & \cancel{7/2\sqrt{2}} & \cancel{-7/2\sqrt{2}} \\ \cancel{0} & \cancel{-7} & 7/2\sqrt{2} & -7/2\sqrt{2} & \cancel{-7/2\sqrt{2}} & 7+7/2\sqrt{2} \end{bmatrix} & \begin{array}{l} \cancel{\rightarrow D_1} \\ \cancel{\rightarrow D_2} \\ \rightarrow D_3 \\ \rightarrow D_4 \\ \cancel{\rightarrow D_5} \\ \rightarrow D_6 \end{array} \end{array} \tag{4.70}$$

The condensed global matrix becomes a 3 × 3 matrix, given as follows:

$$\mathbf{K} = \begin{bmatrix} 7+7/2\sqrt{2} & -7/2\sqrt{2} & 7/2\sqrt{2} \\ -7/2\sqrt{2} & 7/2\sqrt{2} & -7/2\sqrt{2} \\ 7/2\sqrt{2} & -7/2\sqrt{2} & 7+7/2\sqrt{2} \end{bmatrix} \times 10^9\,\mathrm{Nm}^{-2} \tag{4.71}$$

It can easily be confirmed that this condensed stiffness matrix is SPD. The constrained global FE equation is

$$\mathbf{KD} = \mathbf{F} \tag{4.72}$$

where

$$\mathbf{D}^T = \begin{bmatrix} D_3 & D_4 & D_6 \end{bmatrix} \tag{4.73}$$

and the force vector **F** is given as

$$\mathbf{F} = \begin{Bmatrix} 0 \\ -1000 \\ 0 \end{Bmatrix} \mathrm{N} \tag{4.74}$$

Note that the only force applied is at node 2 in the downward direction of D_4. Equation (4.72) is actually equivalent to three simultaneous equations involving the three unknowns D_3, D_4

and D_6, as shown below:

$$\begin{aligned}
&\left[\left(7+7/2\sqrt{2}\right)D_3-\left(7/2\sqrt{2}\right)D_4+\left(7/2\sqrt{2}\right)D_6\right]\times 10^9=0\\
&\left[\left(-7/2\sqrt{2}\right)D_3+\left(7/2\sqrt{2}\right)D_4-\left(7/2\sqrt{2}\right)D_6\right]\times 10^9=-1000\\
&\left[\left(7/2\sqrt{2}\right)D_3-\left(7/2\sqrt{2}\right)D_4+\left(7+7/2\sqrt{2}\right)D_6\right]\times 10^9=0
\end{aligned} \tag{4.75}$$

Step 5: Solving the FE matrix equation The final step would be to solve the FE equation, Eqs. (4.72) or (4.75), to obtain the solution for D_3, D_4 and D_6. Solving this equation manually is possible, since this only involves three unknowns in three equations. To this end, we obtain

$$\begin{aligned}
D_3&=-1.429\times 10^{-7}\ \text{m}\\
D_4&=-6.898\times 10^{-7}\ \text{m}\\
D_6&=-1.429\times 10^{-7}\ \text{m}
\end{aligned} \tag{4.76}$$

To obtain the stresses in the elements, Eq. (4.37) is used as follows:

$$\begin{aligned}
\sigma_x^1 &= E\mathbf{BTD}_e = 70\times 10^9\begin{bmatrix}-1 & 1\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&0&1&0\end{bmatrix}\begin{Bmatrix}0\\0\\-1.429\times 10^{-7}\\-6.898\times 10^{-7}\end{Bmatrix}\\
&=-10003\ \text{Pa}
\end{aligned} \tag{4.77}$$

$$\begin{aligned}
\sigma_x^2 &= E\mathbf{BTD}_e = 70\times 10^9\begin{bmatrix}-1 & 1\end{bmatrix}\begin{bmatrix}0&1&0&0\\0&0&0&1\end{bmatrix}\begin{Bmatrix}0\\0\\0\\-1.429\times 10^{-7}\end{Bmatrix}\\
&=-10003\ \text{Pa}
\end{aligned} \tag{4.78}$$

$$\begin{aligned}
\sigma_x^3 &= E\mathbf{BTD}_e = 70\times 10^9\begin{bmatrix}\frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\end{bmatrix}\begin{bmatrix}-\frac{1}{\sqrt{2}}&\frac{1}{\sqrt{2}}&0&0\\0&0&-\frac{1}{\sqrt{2}}&\frac{1}{\sqrt{2}}\end{bmatrix}\begin{Bmatrix}-1.429\times 10^{-7}\\-6.898\times 10^{-7}\\0\\-1.429\times 10^{-7}\end{Bmatrix}\\
&=14140\ \text{Pa}
\end{aligned} \tag{4.79}$$

In engineering practice, the problem can be of a much larger scale, and thus the unknowns or number of degrees of freedom will also be very much more. Therefore, numerical methods, or so-called solvers for solving the FEM equations, have to be used. Typical real life engineering problems might involve hundreds of thousands, and even millions, of degrees of freedom. Many kinds of such solvers are routinely available in math or numerical libraries in computer systems.

4.4 HIGH ORDER ONE-DIMENSIONAL ELEMENTS

For truss members that are free of body forces, there is no need to use higher order elements, as the linear element can already give the exact solution, as shown in Example 4.1. However, for truss members subjected to body forces arbitrarily distributed in the truss elements along its axial direction, higher order elements can be used for more accurate analysis. The procedure for developing such high order one-dimensional elements is the same as for the linear elements. The only difference is the shape functions.

In deriving high order shape functions, we usually use the natural coordinate ξ, instead of the physical coordinate x. The natural coordinate ξ is defined as

$$\xi = 2\frac{x - x_c}{l_e} \tag{4.80}$$

where x_c is the physical coordinate of the mid point of the one-dimensional element. In the natural coordinate system, the element is defined in the range of $-1 \leq \xi \leq 1$. Figure 4.7 shows a one-dimensional element of nth order with $(n + 1)$ nodes. The shape function of the element can be written in the following form using so-called Lagrange interpolants:

$$N_k(\xi) = l_k^n(\xi) \tag{4.81}$$

where $l_k^n(x)$ is the well-known Lagrange interpolants, defined as

$$l_k^n(\xi) = \frac{(\xi - \xi_0)(\xi - \xi_1)\cdots(\xi - \xi_{k-1})(\xi - \xi_{k+1})\cdots(\xi - \xi_n)}{(\xi_k - \xi_0)(\xi_k - \xi_1)\cdots(\xi_k - \xi_{k-1})(\xi_k - \xi_{k+1})\cdots(\xi_k - \xi_n)} \tag{4.82}$$

From Eq. (4.82), it is clear that

$$N_k(\xi) = \begin{cases} 1 & \text{at node } k \text{ where } \xi = \xi_k \\ 0 & \text{at other nodes} \end{cases} \tag{4.83}$$

Therefore, the high order shape functions defined by Eq. (4.81) are of the delta function property.

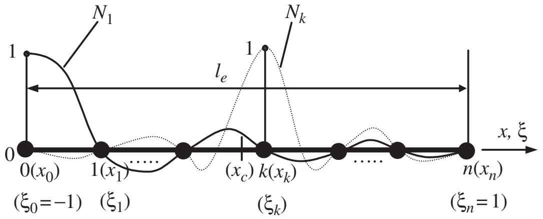


Figure 4.7. One-dimensional element of *n*th order with $(n + 1)$ nodes.

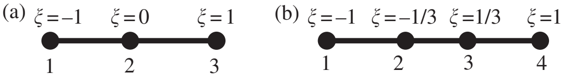

(a) $\xi=-1$ $\xi=0$ $\xi=1$
1 2 3
(b) $\xi=-1$ $\xi=-1/3$ $\xi=1/3$ $\xi=1$
1 2 3 4

Figure 4.8. One-dimensional quadratic and cubic element with evenly distributed nodes. (a) Quadratic element, (b) cubic element.

Using Eq. (4.82), *the quadratic one-dimensional element* with three nodes shown in Figure 4.8a can be obtained explicitly as

$$\begin{aligned} N_1(\xi) &= -\tfrac{1}{2}\xi(1-\xi) \\ N_2(\xi) &= \tfrac{1}{2}\xi(1+\xi) \\ N_3(\xi) &= (1+\xi)(1-\xi) \end{aligned} \tag{4.84}$$

The *cubic one-dimensional element* with four nodes shown in Figure 4.8b can be obtained as

$$\begin{aligned} N_1(\xi) &= -\tfrac{1}{16}(1-\xi)(1-9\xi^2) \\ N_2(\xi) &= -\tfrac{1}{16}(1+\xi)(1-9\xi^2) \\ N_3(\xi) &= \tfrac{9}{16}(1-3\xi)(1-\xi^2) \\ N_4(\xi) &= \tfrac{9}{16}(1+3\xi)(1-\xi^2) \end{aligned} \tag{4.85}$$

4.5 REVIEW QUESTIONS

1. What are the characteristics of the joints in a truss structure and what are the effects of this on the deformation and stress properties in a truss element?
2. How many DOFs does a two-nodal, planar truss element have in its local coordinate system, and in the global coordinate system? Why is there a difference in DOFs in these two coordinate systems?
3. How many DOFs does a two-nodal, space truss element have in its local coordinate system, and in the global coordinate system? Why is there such a difference?
4. Write down the expression for the element stiffness matrix, $\mathbf{k}_e$, with Young's modulus, E, length, l_e, and cross-sectional area, $A = 0.02x + 0.01$. (Note: non-uniform cross-sectional area.)
5. Write down the expression for the element mass matrix, $\mathbf{m}_e$, with the same properties as that in question 4 above.
6. Work out the displacements of the truss structure shown in Figure 4.9. All the truss members are of the same material ($E = 69.0$ GPa) and with the same cross-sectional area of $0.01\ \text{m}^2$.

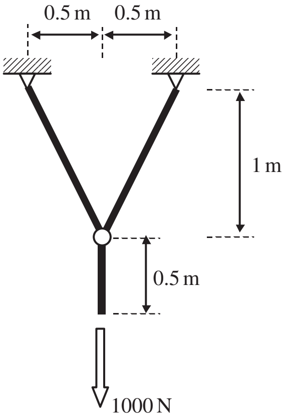


Figure 4.9. Three member planar truss structure.

5

FEM FOR BEAMS

5.1 INTRODUCTION

A *beam* is another simple but commonly used structural component. It is also geometrically a straight *bar* of an arbitrary cross-section, but it deforms only in directions perpendicular to its axis. Note that the main difference between the beam and the truss is the type of load they carry. Beams are subjected to transverse loading, including transverse forces and moments that result in transverse deformation. Finite element equations for beams will be developed in this chapter, and the element developed is known as the *beam element*. The basic concepts, procedures and formulations can also be found in many existing textbooks (see, e.g. Petyt,1990; Reddy, 1993; Rao, 1999; Zienkiewicz and Taylor, 2000; etc.).

In beam structures, the beams are joined together by welding (not by pins or hinges, as in the case of truss elements), so that both forces and moments can be transmitted between the beams. In this book, the cross-section of the beam structure is assumed uniform. If a beam has a varying cross-section, it is advised that the beam should be divided into shorter beams, where each can be treated as beam(s) with a uniform cross-section. Nevertheless, the FE matrices for varying cross-sectional area can also be developed with ease using the same concepts that are introduced. The beam element developed in this chapter is based on the Euler–Bernoulli beam theory that is applicable for thin beams.

5.2 FEM EQUATIONS

In planar beam elements there are two degrees of freedom (DOFs) at a node in its local coordinate system. They are deflection in the y direction, v, and rotation in the x–y plane, θ_z with respect to the z-axis (see Section 2.5). Therefore, each beam element has a total of four DOFs.

5.2.1 Shape Function Construction

Consider a beam element of length $l = 2a$ with nodes 1 and 2 at each end of the element, as shown in Figure 5.1. The local x-axis is taken in the axial direction of the element with its origin at the middle section of the beam. Similar to all other structures, to develop the FEM equations, shape functions for the interpolation of the variables from the nodal variables

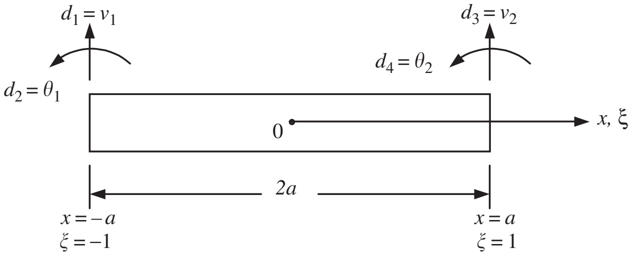


Figure 5.1. Beam element and its local coordinate systems: physical coordinates x, and natural coordinates ξ.

would first have to be developed. As there are four DOFs for a beam element, there should be four shape functions. It is often more convenient if the shape functions are derived from a special set of local coordinates, which is commonly known as the *natural coordinate system*. This natural coordinate system has its origin at the centre of the element, and the element is defined from -1 to $+1$, as shown in Figure 5.1.

The relationship between the natural coordinate system and the local coordinate system can be simply given as

$$\xi = \frac{x}{a} \tag{5.1}$$

To derive the four shape functions in the natural coordinates, the displacement in an element is first assumed in the form of a third order polynomial of ξ that contains four unknown constants:

$$v(\xi) = \alpha_0 + \alpha_1\xi + \alpha_2\xi^2 + \alpha_3\xi^3 \tag{5.2}$$

where α_0 to α_3 are the four unknown constants. The third order polynomial is chosen because there are four unknowns in the polynomial, which can be related to the four nodal DOFs in the beam element. The above equation can have the following matrix form:

$$v(\xi) = \begin{bmatrix} 1 & \xi & \xi^2 & \xi^3 \end{bmatrix} \begin{Bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{Bmatrix} \tag{5.3}$$

or

$$v(\xi) = \mathbf{p}^T(\xi)\boldsymbol{\alpha} \tag{5.4}$$

where $\mathbf{p}$ is the vector of basis functions and $\boldsymbol{\alpha}$ is the vector of coefficients, as discussed in Chapters 3 and 4. The rotation θ can be obtained from the differential of Eq. (5.2) with the use of Eq. (5.1):

$$\theta = \frac{\partial v}{\partial x} = \frac{\partial v}{\partial \xi}\frac{\partial \xi}{\partial x} = \frac{1}{a}\frac{\partial v}{\partial \xi} = \frac{1}{a}(\alpha_1 + 2\alpha_2\xi + 3\alpha_3\xi^2) \tag{5.5}$$

The four unknown constants α_0 to α_3 can be determined by utilizing the following four conditions:
At $x = -a$ or $\xi = -1$:

$$\begin{aligned} &(1) \quad v(-1) = v_1 \\ &(2) \quad \left.\frac{\mathrm{d}v}{\mathrm{d}x}\right|_{\xi=-1} = \theta_1 \end{aligned} \tag{5.6}$$

At $x = +a$ or $\xi = +1$:

$$\begin{aligned} &(3) \quad v(1) = v_2 \\ &(4) \quad \left.\frac{\mathrm{d}v}{\mathrm{d}x}\right|_{\xi=1} = \theta_2 \end{aligned} \tag{5.7}$$

The application of the above four conditions gives

$$\begin{Bmatrix} v_1 \\ \theta_1 \\ v_2 \\ \theta_2 \end{Bmatrix} = \begin{bmatrix} 1 & -1 & 1 & -1 \\ 0 & 1/a & -2/a & 3/a \\ 1 & 1 & 1 & 1 \\ 0 & 1/a & 2/a & 3/a \end{bmatrix} \begin{Bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{Bmatrix} \tag{5.8}$$

or

$$\mathbf{d}_e = \mathbf{A}_e \boldsymbol{\alpha} \tag{5.9}$$

Solving the above equation for α gives

$$\boldsymbol{\alpha} = \mathbf{A}_e^{-1}\,\mathbf{d}_e \tag{5.10}$$

where

$$\mathbf{A}_e^{-1} = \frac{1}{4}\begin{bmatrix} 2 & a & 2 & -a \\ -3 & -a & 3 & -a \\ 0 & -a & 0 & a \\ 1 & a & -1 & a \end{bmatrix} \tag{5.11}$$

Hence, substituting Eq. (5.10) into Eq. (5.4) will give

$$v = \mathbf{N}(\xi)\mathbf{d}_e \tag{5.12}$$

where $\mathbf{N}$ is a *matrix of shape functions* given by

$$\mathbf{N}(\xi) = \mathbf{P}\mathbf{A}_e^{-1} = \begin{bmatrix} N_1(\xi) & N_2(\xi) & N_3(\xi) & N_4(\xi) \end{bmatrix} \tag{5.13}$$

in which the shape functions are found to be

$$\begin{aligned} N_1(\xi) &= \tfrac{1}{4}(2 - 3\xi + \xi^3) \\ N_2(\xi) &= \tfrac{1}{4}a(1 - \xi - \xi^2 + \xi^3) \\ N_3(\xi) &= \tfrac{1}{4}(2 + 3\xi - \xi^3) \\ N_4(\xi) &= \frac{a}{4}(-1 - \xi + \xi^2 + \xi^3) \end{aligned} \tag{5.14}$$

It can be easily confirmed that the two translational shape functions N_1 and N_3 satisfy conditions defined by Eqs. (3.34) and (3.41). However, the two rotational shape functions

N_2 and N_4 do not satisfy the conditions of Eqs. (3.34) and (3.41). This is because these two shape functions relate to rotational degrees of freedom, which are derived from the deflection functions. Satisfaction of N_1 and N_3 to Eq. (3.34) has already ensured the correct representation of the rigid body movement of the beam element.

5.2.2 Strain Matrix

Having now obtained the shape functions, the next step would be to obtain the element strain matrix. Substituting Eq. (5.12) into Eq. (2.47), which gives the relationship between the strain and the deflection, we have

$$\varepsilon_{xx} = \mathbf{B}\,\mathbf{d}_e \tag{5.15}$$

where the strain matrix **B** is given by

$$\mathbf{B} = -yL\mathbf{N} = -y\frac{\partial^2}{\partial x^2}\mathbf{N} = -\frac{y}{a^2}\frac{\partial^2}{\partial \xi^2}\mathbf{N} = -\frac{y}{a^2}\mathbf{N}'' \tag{5.16}$$

In deriving the above equation, Eqs. (2.48) and (5.1) have been used. From Eq. (5.14), we have

$$\mathbf{N}'' = \begin{bmatrix} N_1'' & N_2'' & N_3'' & N_4'' \end{bmatrix} \tag{5.17}$$

where

$$\begin{aligned} N_1'' &= \frac{3}{2}\xi, \quad N_2'' = \frac{a}{2}(-1+3\xi) \\ N_3'' &= -\frac{3}{2}\xi, \quad N_4'' = \frac{a}{2}(1+3\xi) \end{aligned} \tag{5.18}$$

5.2.3 Element Matrices

Having obtained the strain matrix, we are now ready to obtain the element stiffness and mass matrices. By substituting Eq. (5.16) into Eq. (3.71), the stiffness matrix can be obtained as

$$\begin{aligned} \mathbf{k}_e &= \int_V \mathbf{B}^T \mathbf{c} \mathbf{B}\,\mathrm{d}V = E\int_A y^2\,\mathrm{d}A \int_{-a}^{a}\left(\frac{\partial^2}{\partial x^2}\mathbf{N}\right)^T\left(\frac{\partial^2}{\partial x^2}\mathbf{N}\right)\mathrm{d}x \\ &= EI_z\int_{-1}^{1}\frac{1}{a^4}\left[\frac{\partial^2}{\partial \xi^2}\mathbf{N}\right]^T\left[\frac{\partial^2}{\partial \xi^2}\mathbf{N}\right]a\,\mathrm{d}\xi = \frac{EI_z}{a^3}\int_{-1}^{1}\mathbf{N}''^T\mathbf{N}''\,\mathrm{d}\xi \end{aligned} \tag{5.19}$$

where $I_z = \int_A y^2\,\mathrm{d}A$ is the second moment of area (or moment of inertia) of the cross-section of the beam with respect to the z axis. Substituting Eq. (5.17) into (5.19), we obtain

$$\mathbf{k}_e = \frac{EI_z}{a^3}\int_{-1}^{1}\begin{bmatrix} N_1''N_1'' & N_1''N_2'' & N_1''N_3'' & N_1''N_4'' \\ N_2''N_1'' & N_2''N_2'' & N_2''N_3'' & N_2''N_4'' \\ N_3''N_1'' & N_3''N_2'' & N_3''N_3'' & N_1''N_4'' \\ N_4''N_1'' & N_4''N_2'' & N_4''N_3'' & N_1''N_4'' \end{bmatrix} \mathrm{d}x \tag{5.20}$$

Evaluating the integrals in the above equation leads to

$$\mathbf{k}_e = \frac{EI_z}{2a^3}\begin{bmatrix} 3 & 3a & -3 & 3a \\ & 4a^2 & -3a & 2a^2 \\ & & 3 & -3a \\ sy. & & & 4a^2 \end{bmatrix} \tag{5.21}$$

To obtain the mass matrix, we substitute Eq. (5.13) into Eq. (3.75):

$$\mathbf{m}_e = \int_V \rho\mathbf{N}^T\mathbf{N}\,\mathrm{d}V = \rho\int_A \mathrm{d}A\int_{-a}^{a}\mathbf{N}^T\mathbf{N}\,\mathrm{d}x = \rho A\int_{-1}^{1}\mathbf{N}^T\mathbf{N}a\,\mathrm{d}\xi$$

$$= \rho Aa\int_{-1}^{1}\begin{bmatrix} N_1N_1 & N_1N_2 & N_1N_3 & N_1N_4 \\ N_2N_1 & N_2N_2 & N_2N_3 & N_2N_4 \\ N_3N_1 & N_3N_2 & N_3N_3 & N_3N_4 \\ N_4N_1 & N_4N_2 & N_4N_3 & N_4N_4 \end{bmatrix} \mathrm{d}x \tag{5.22}$$

where A is the area of the cross-section of the beam. Evaluating the integral in the above equation leads to

$$\mathbf{m}_e = \frac{\rho Aa}{105}\begin{bmatrix} 78 & 22a & 27 & -13a \\ & 8a^2 & 13a & -6a^2 \\ & & 78 & -22a \\ sy. & & & 8a^2 \end{bmatrix} \tag{5.23}$$

The other element matrix would be the force vector. The nodal force vector for beam elements can be obtained using Eqs. (3.78), (3.79) and (3.81). Suppose the element is loaded by an external distributed force f_y along the x-axis, two concentrated forces f_{s1} and f_{s2}, and concentrated moments m_{s1} and m_{s2}, respectively, at nodes 1 and 2; the total nodal force vector becomes

$$\mathbf{f}_e = \int_V \mathbf{N}^T f_b\,\mathrm{d}V + \int_{S_f}\mathbf{N}^T f_s\,\mathrm{d}S_f$$

$$= f_y a\int_{-1}^{1}\begin{bmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \end{bmatrix}\mathrm{d}\xi + \begin{Bmatrix} f_{s1} \\ m_{s1} \\ f_{s2} \\ m_{s1} \end{Bmatrix} = \begin{Bmatrix} f_y a + f_{s1} \\ f_y a^2/3 + m_{s1} \\ f_y a + f_{s2} \\ -f_y a^2/3 + m_{s1} \end{Bmatrix} \tag{5.24}$$

The final FEM equation for beams has the form of Eq. (3.89), but the element matrices are defined by Eqs. (5.21), (5.23) and (5.24).

5.3 REMARKS

Theoretically, coordinate transformation can also be used to transform the beam element matrices from the local coordinate system into the global coordinate system. However, the transformation is necessary only if there is more than one beam element in the beam structure, and of which there are at least two beam elements of different orientations. A beam structure with at least two beam elements of different orientations is commonly termed a *frame* or *framework*. To analyse frames, frame elements, which carry both axial and bending forces, have to be used, and coordinate transformation is generally required. Detailed formulation for frames is discussed in the next chapter.

5.4 WORKED EXAMPLES

Example 5.1: A uniform cantilever beam subjected to a downward force

Consider the cantilever beam as shown in Figure 5.2. The beam is fixed at one end, and it has a uniform cross-sectional area as shown. The beam undergoes static deflection by a downward load of $P = 1000\,\text{N}$ applied at the free end. The dimensions of the beam are shown in the figure, and the beam is made of aluminium whose properties are shown in Table 5.1.

To make clear the steps involved in solving this simple example, we first used just one beam element to solve for the deflection. The beam element would have degrees of freedom as shown in Figure 5.1.

Step 1: Obtaining the element matrices The first step in formulating the finite element equations is to form the element matrices and, in this case, being the only element used, the element matrices are actually the global finite element matrices, since no assembly is required. The shape functions for the four degrees of freedom are given in Eq. (5.14). The element stiffness matrix can be obtained using Eq. (5.21). Note that as this is a static problem, the mass matrix is not required here. The second moment of area of the cross-sectional area

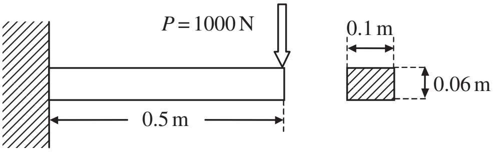


Figure 5.2. Cantilever beam under static load.

Table 5.1. Material properties of aluminium

Young's modulus, E GPa	Poisson's ratio, v
69.0	0.33

about the z-axis can be given as

$$I_z = \tfrac{1}{12}bh^3 = \tfrac{1}{12}\,(0.1)\,(0.06)^3 = 1.8 \times 10^{-6}\,\text{m}^4 \tag{5.25}$$

Since only one element is used, the stiffness matrix of the beam is thus the same as the element stiffness matrix:

$$\mathbf{K} = \mathbf{k}_e = \frac{\left(69\times10^9\right)\left(1.8\times10^{-6}\right)}{2\times0.25^3}\begin{bmatrix} 3 & 0.75 & -3 & 0.75 \\ 0.75 & 0.25 & -0.75 & 0.125 \\ -3 & -0.75 & 3 & -0.75 \\ 0.75 & 0.125 & -0.75 & 0.25 \end{bmatrix}$$

$$= 3{,}974\times10^6\begin{bmatrix} 3 & 0.75 & -3 & 0.75 \\ 0.75 & 0.25 & -0.75 & 0.125 \\ -3 & -0.75 & 3 & -0.75 \\ 0.75 & 0.125 & -0.75 & 0.25 \end{bmatrix}\text{Nm}^{-2} \tag{5.26}$$

The finite element equation becomes

$$\underbrace{3{,}974\times10^6\begin{bmatrix} 3 & 0.75 & -3 & 0.75 \\ 0.75 & 0.25 & -0.75 & 0.125 \\ -3 & -0.75 & 3 & -0.75 \\ 0.75 & 0.125 & -0.75 & 0.25 \end{bmatrix}}_{\mathbf{K}}\underbrace{\begin{Bmatrix} v_1 \\ \theta_1 \\ v_2 \\ \theta_2 \end{Bmatrix}}_{\mathbf{D}}$$

$$= \underbrace{\begin{Bmatrix} Q_1 = ? \\ M_1 = ? \\ Q_2 = P \\ M_2 = 0 \end{Bmatrix}}_{\mathbf{F}}\begin{matrix} \rightarrow \text{unknown reaction shear force} \\ \rightarrow \text{unknown reaction moment} \\ \\ \\ \end{matrix} \tag{5.27}$$

Note that, at node 1, the beam is clamped. Therefore, the shear force and moment at this node should be the reaction force and moment, which are unknowns before the FEM equation is solved for the displacements. To solve Eq. (5.27), we need to impose the displacement boundary condition at the clamped node.

Step 2: Applying boundary conditions The beam is fixed or clamped at one end. This implies that at that end, the deflection, v_1, and the slope, θ_1, are both equal to zero:

$$v_1 = \theta_1 = 0 \tag{5.28}$$

The imposition of the above displacement boundary condition leads to the removal of the first and second rows and columns of the stiffness matrix:

$$3.974\times10^6\begin{bmatrix} \cancel{3} & \cancel{0.75} & \cancel{-3} & \cancel{0.75} \\ \cancel{0.75} & \cancel{0.25} & \cancel{0.75} & \cancel{0.125} \\ \cancel{-3} & \cancel{-0.75} & 3 & -0.75 \\ \cancel{0.75} & \cancel{0.125} & -0.75 & 0.25 \end{bmatrix}\begin{Bmatrix} \cancel{v_1 = 0} \\ \cancel{\theta_1 = 0} \\ v_2 \\ \theta_2 \end{Bmatrix} = \begin{Bmatrix} \cancel{Q_1} \\ \cancel{M_1} \\ Q_2 = P \\ M_2 = 0 \end{Bmatrix} \tag{5.29}$$

The reduced stiffness matrix becomes a 2 × 2 matrix of

$$\mathbf{K} = 3.974 \times 10^6 \begin{bmatrix} 3 & -0.75 \\ -0.75 & 0.25 \end{bmatrix} \text{Nm}^{-2} \tag{5.30}$$

The finite element equation, after the imposition of the displacement condition, is thus

$$\mathbf{Kd} = \mathbf{F} \tag{5.31}$$

where

$$\mathbf{d}^T = [v_2 \quad \theta_2] \tag{5.32}$$

and the force vector **F** is given as

$$\mathbf{F} = \begin{Bmatrix} -1000 \\ 0 \end{Bmatrix} \text{N} \tag{5.33}$$

Note that, although we do not know the reaction shear force Q_1 and the moment M_1, it does not affect our solving of the FEM equation, because we know v_1 and θ_1 instead. This allows us to remove the unknowns of Q_1 and M_1 from the original FEM equation. We will come back to calculate the unknowns of Q_1 and M_1, after we have solved the FEM equations for all the displacements (deflections and rotations).

Step 3: Solving the FE matrix equation The last step in this simple example would be to solve Eq. (5.31) to obtain v_2 and θ_2. In this case, Eq. (5.31) is actually two simultaneous equations involving two unknowns, and can be easily solved manually. Of course, when we have more unknowns or degrees of freedom, some numerical methods of solving the matrix equation might be required. The solution to Eq. (5.31) is

$$\begin{aligned} v_2 &= -3.355 \times 10^{-4} \text{ m} \\ \theta_2 &= -1.007 \times 10^{-3} \text{ rad} \end{aligned} \tag{5.34}$$

After v_2 and θ_2 have been obtained, they are substituted back into the first two equations of Eq. (5.27) to obtain the reaction shear force at node 1:

$$\begin{aligned} Q_1 &= 3.974 \times 10^6(-3v_2 + 0.75\theta_2) \\ &= 3.974 \times 10^6[-3 \times (-3.355 \times 10^{-4}) + 0.75 \times (-1.007 \times 10^{-3})] \\ &= 998.47\,\text{N} \end{aligned} \tag{5.35}$$

and the reaction moment at node 1:

$$\begin{aligned} M_1 &= 3.974 \times 10^6(-0.75v_2 + 0.125\theta_2) \\ &= 3.974 \times 10^6[-0.75 \times (-3.355 \times 10^{-4}) + 0.125 \times (-1.007 \times 10^{-3})] \\ &= 499.73\,\text{Nm} \end{aligned} \tag{5.36}$$

This completes the solution process of this problem.

Note that this solution is exactly the same as the analytical solution. We again observe the reproduction feature of the FEM that was revealed in Example 4.1. In this case, it is because the exact solution of the deflection for the cantilever thin beam is a third order polynomial, which can be obtained easily by solving the strong form of the system equation of beam given by Eq. (5.59) with $f_y = 0$. On the other hand, the shape functions used in our FEM analysis are also third order polynomials (see Eq. (5.14) or Eq. (5.2)). Therefore, the exact solution of the problem is included in the set of assumed deflections. The FEM based on Hamilton's principle has indeed reproduced the exact solution. This is, of course, also true if we were to calculate the deflection at anywhere else other than the nodes. For example, to compute the deflection at the centre of the beam, we can use Eq. (5.12) with $x = 0.25$, or in the natural coordinate system, $\xi = 0$, and substituting the values calculated at the nodes:

$$v_{\xi=0} = \mathbf{N}_{\xi=0}\,\mathbf{d}_e = \begin{bmatrix} \frac{1}{2} & \frac{1}{16} & \frac{1}{2} & -\frac{1}{16} \end{bmatrix} \begin{Bmatrix} 0 \\ 0 \\ -3.355 \times 10^{-4} \\ -1.007 \times 10^{-3} \end{Bmatrix} = -1.048 \times 10^{-4}\,\text{m} \tag{5.37}$$

To calculate the rotation at the centre of the beam, the derivatives of the shape functions are used as follows:

$$\begin{aligned} \theta_{\xi=0} &= \left(\frac{\mathrm{d}v}{\mathrm{d}x}\right)_{\xi=0} = \left(\frac{\mathrm{d}\mathbf{N}}{\mathrm{d}x}\right)_{\xi=0} \mathbf{d}_e = \begin{bmatrix} -3 & -\frac{1}{4} & 3 & -\frac{1}{4} \end{bmatrix} \begin{Bmatrix} 0 \\ 0 \\ -3.355 \times 10^{-4} \\ -1.007 \times 10^{-3} \end{Bmatrix} \\ &= -7.548 \times 10^{-4}\,\text{rad} \end{aligned} \tag{5.38}$$

Note that in obtaining $\mathrm{d}\mathbf{N}/\mathrm{d}x$ above, the chain rule of differentiation is used together with the relationship between x and ξ as depicted in Eq. (5.1).

5.5 CASE STUDY: RESONANT FREQUENCIES OF MICRO RESONANT TRANSDUCER

Making machines as small as insects, or even smaller, has been a dream of scientists for many years. Made possible by present lithographic techniques, such micro-systems are now being produced and applied in our daily lives. Such machines are called micro-electro-mechanical systems (MEMS), usually composed of mechanical and electrical devices. There are many MEMS devices, from micro actuators and sensors to micro fluidic devices, being designed and manufactured today. The technology has very wide applications in communication, medical, aerospace, robotics, and so on.

One of the most common micro-electro-mechanical (MEMS) devices is the resonant transducer. Resonant transducers convert externally induced beam strain into a beam resonant frequency change. This change in resonant frequency is then typically detected by implanted piezoresistors or optical techniques. Such resonant transducers are used for the

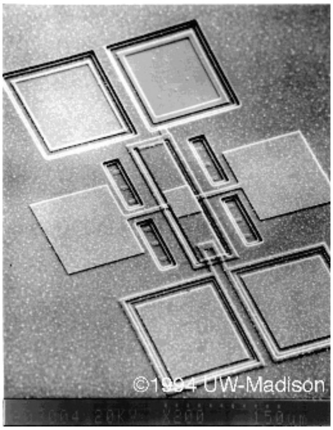


Figure 5.3. Resonant micro-beam strain transducer (Courtesy of Professor Henry Guckel and the University of Wisconsin-Madison).

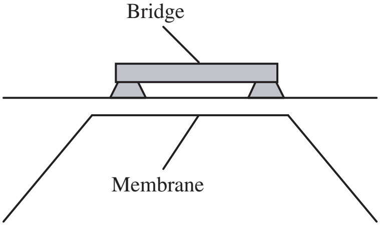


Figure 5.4. Bridge in a micro resonant transducer.

measurement of pressure, acceleration, strain, vibration, and so on. Figure 5.3 shows a micrograph of a micro polysilicon resonant microbeam transducer.

Figure 5.3 shows an overall view of the transducer, but the principle of the resonant transducer actually lies in the clamped–clamped bridge on top of a membrane. This bridge is actually located at the centre of the micrograph. Figure 5.4 shows a schematic side view of the bridge structure. The resonant frequency of the bridge is related to the force applied to it (between anchor points), its material properties, cross-sectional area and length. When the membrane deforms, for example, due to a change in pressure, the force applied to the bridge also changes, resulting in a change in the resonant frequency of the bridge.

It is thus important to analyse the resonant frequency of this bridge structure in the design of the resonant transducer. We use the beam element in the software ABAQUS to solve for the first three resonant frequencies of the bridge. The dimensions of the clamped–clamped bridge structure shown in Figure 5.5 are used to model a bridge in a micro resonant transducer. The material properties of polysilicon, of which the resonant transducer is normally made, are shown in Table 5.2.

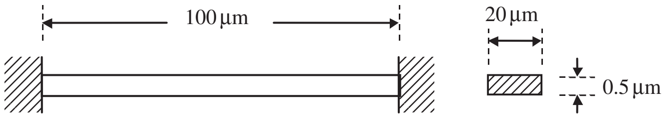


Figure 5.5. Geometrical dimensions of clamped–clamped bridge.

Table 5.2. Elastic properties of polysilicon

Young's Modulus, E	169 GPa
Poisson's ratio, ν	0.262
Density, ρ	2300 kg m^{-3}

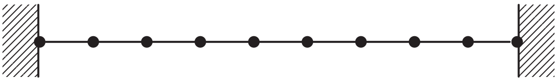


Figure 5.6. Ten element mesh of clamped–clamped bridge.

5.5.1 Modelling

The modelling of the bridge is done using one-dimensional beam elements developed in this chapter. The beam is assumed to be clamped at two ends of the beam. The meshing of the structure should not pose any difficulty, but what is important here is the choice of how many elements to use to give sufficient accuracy. Because the exact solution of free vibration modes of the beam is no longer of a polynomial type, the FEM will not be able to produce the exact solution, but an approximated solution. One naturally becomes concerned with whether the results converge and whether they are accurate.

To start, the first analysis will mesh the beam uniformly into ten two-nodal beam elements, as shown in Figure 5.6. This simple mesh will serve to show clearly the steps used in ABAQUS. Refined uniform meshes of 20, 40 and 60 elements will then be used to check the accuracy of the results obtained. This is a simplified way of performing what is commonly known as a convergence test. Remember that usually the greater the number of elements, the greater the accuracy. However, we can't simply use as many elements as possible all the time, since, there is usually a limit to the computer resources available. Hence, convergence tests are carried out to determine the optimum number of elements or nodes to be used for a certain problem. What is meant by 'optimum' means the least number of elements or nodes to yield a desired accuracy within the acceptable tolerance.

5.5.2 ABAQUS Input File

The ABAQUS input file for the above described finite element model is shown below. In the early days, the analyst had to write these cards manually, but now it is generated by the preprocessors of FEM packages. Understanding the input file is very important both for undersanding the FEM and to effectively use the FEM packages. The text boxes to the right of the input file are not part of the input file, but explain what the sections of the file meant.

```
* HEADING, SPARSE
ABAQUS job to calculate eigenvalues of beam
**
*NODE
1, 0., 0.
2, 10., 0.
3, 20., 0.
4, 30., 0.
5, 40., 0.
6, 50., 0.
7, 60., 0.
8, 70., 0.
9, 80., 0.
10, 90., 0.
11, 100., 0.
**
**
```

Nodal cards

Define the coordinates of the nodes in the model. The first entry is the node ID, while the second and third entries are the *x* and *y* coordinates of the position of the node, respectively.

```
* ELEMENT, TYPE=B23, ELSET=BEAM
1, 1, 2
2, 2, 3
3, 3, 4
4, 4, 5
5, 5, 6
6, 6, 7
7, 7, 8
8, 8, 9
9, 9, 10
10, 10, 11
**
** beam
**
```

Element (connectivity) cards

Define the element type and what nodes make up the element. B23 represents that it is a planar, cubic, Euler–Bernoulli beam element. There are many other beam element types in the ABAQUS element library. The “ELSET = BEAM” statement is simply for naming this set of elements so that it can be referenced when defining the material properties. In the subsequent data entry, the first entry is the element ID, and the following two entries are the nodes making up the element.

```
* BEAM SECTION, ELSET=BEAM, SECTION=RECT, MATERIAL=POLYSILI
20., 0.5,
0.0, 0.0, -1.0
**
**
**
**
**
**
**
** polysilicon
**
* MATERIAL, NAME=POLYSILI
**
```

Property cards

Define properties to the elements of set “BEAM”. “SECT = RECT” describes the cross-section as a rectangle. ABAQUS provides a choice of other cross-sections. The first data line under “BEAM SECTION” defines the geometry of the cross-section. The second data line defines the normal, which in this case is for a planar beam. It will have the material properties defined under “POLYSILI”.

```
* DENSITY
2.3E-15,
**
* ELASTIC, TYPE=ISO
169000., 0.262
**
**
```

Material cards

Define material properties under the name "POLYSILI". Density and elastic properties are defined. TYPE=ISO represents isotropic properties.

```
* BOUNDARY, OP=NEW
1, 1,, 0.
1, 2,, 0.
2, 1,, 0.
3, 1,, 0.
4, 1,, 0.
5, 1,, 0.
6, 1,, 0.
7, 1,, 0.
8, 1,, 0.
9, 1,, 0.
10, 1,, 0.
11, 1,, 0.
11, 2,, 0.
**
*BOUNDARY, OP=NEW
1, 6,, 0.
11, 6,, 0.
**
```

Boundary (BC) cards

Define boundary conditions. For nodes 1 and 11, the DOFs 1, 2 and 6 are constrained. For the rest, DOF 1 is constrained. Note that in ABAQUS, a planar beam has x and y translational displacements, as well as rotation about the z-axis.

```
** Step 1, eigen
** LoadCase, Default
**
* STEP, NLGEOM
This load case is the default load case that always appears
*FREQUENCY
3, 0.,,, 30
**
**
**
```

Control cards

Indicate the analysis step. In this case it is a "FREQUENCY" analysis or an eigenvalue analysis.

```
* NODE PRINT, FREQ=1
U,
* NODE FILE, FREQ=1
U,
**
**
**
* END STEP
```

Output control cards

Define the ouput required. For example, in this case, we require the nodal output, displacement "U".

The input file above shows how a basic ABAQUS input file is set up. Note that all the input file does is provide the information necessary so that the program can utilize them to formulate and solve the finite element equations. It may also be noticed that in the input file, there is no mention of the units of measurement used. This implies that the units must definitely be consistent throughout the input file in all the information provided. For example, if the coordinate values of the nodes are in micrometres, the units for other values like the Young's modulus, density, forces and so on must also undergo the necessary conversions in order to be consistent, before they are keyed into the preprocessor of ABAQUS. It is noted that in this case study, all the units are converted into micrometres to be consistent with the geometrical dimensions, as can be seen from the values of Young's modulus and density. This is the case for most finite element software, and many times, errors in analysis occur due to negligence in ensuring the units' consistency. More details regarding the setting up of an ABAQUS input file will be provided in Chapter 13.

5.5.3 Solution Process

Let us now try to relate the information provided in the input file with what is formulated in this chapter. The first part of the ABAQUS input normally describes the nodes and their coordinates (position). These lines are often called 'nodal cards[1]'. The second part of the input file are the so-called 'element cards'. Information regarding the definition of the elements using nodes is provided. For example, element 1 is formed by nodes 1 and 2. The element cards give the connectivity of the element or the order of the nodal number that forms the element. The connectivity is very important, because a change in the order of the nodal numbers may lead to a breakdown of the computation. The connectivity is also used as the index for the direct assembly of the global matrices (see Example 4.2). This element and nodal information is required for determining the stiffness matrix (Eq. (5.21)) and the mass matrix (Eq. (5.23)).

The property cards define the properties (type of element, cross-sectional property, etc.) of the elements, as well as the material which the element is made of. The cross-section of the element is defined here as it is required for computation of the moment of area about the z-axis, which is in turn used in the stiffness matrix. The material properties defined are also a necessity for the computation of both the stiffness (elastic properties) and mass matrices (density).

The boundary cards (BC cards) define the boundary conditions for the model. In ABAQUS, a node of a *general beam element* (equivalent to the frame element, to be discussed in the next chapter) in the XY plane has three DOFs: translational displacements in the x and y directions (1, 2), and the rotation about the z-axis (6). To model just the transverse displacements and rotation as depicted in the formulation in this chapter, the x-displacement DOFs are constrained here. Hence it can be seen from the input file that the DOF '1' is constrained for all nodes. In addition to this, the two nodes at the ends, nodes 1 and 11, also have their '2' and '6' DOFs constrained to simulate clamped ends. Just as in the worked example previously, constraining these DOFs would effectively reduce the dimension of the matrix.

[1] In the early 1980s, the input files were recorded by pieces of card, each of which recorded one line.

We should usually also have *load cards*. Because this case study is an eigenvalue analysis, there are no external loadings, and hence there is no need to define any loadings in the input file.

The control cards are used to control the analysis, such as defining the type of analysis required. ABAQUS uses the subspace iteration scheme by default to evaluate the eigenvalues of the equation of motion. This method is a very effective method of determining a number of lowest eigenvalues and corresponding eigenvectors for a very large system of several thousand DOFs. The procedure is as follows:

(i) To determine the n lowest eigenvalues and eigenvectors, select a starting matrix $\mathbf{X}_1$ having m $(>n)$ columns.
(ii) Solve the equation $\mathbf{K}\overline{\mathbf{X}}_{k+1} = \mathbf{M}\mathbf{X}_k$ for $\overline{\mathbf{X}}_{k+1}$.
(iii) Calculate $\mathbf{K}_{k+1} = \overline{\mathbf{X}}_{k+1}^T\mathbf{K}\overline{\mathbf{X}}_{k+1}$ and $\mathbf{M}_{k+1} = \overline{\mathbf{X}}_{k+1}^T\mathbf{M}\overline{\mathbf{X}}_{k+1}$, where the dimension of $\mathbf{K}_{k+1}$ and $\mathbf{M}_{k+1}$ are of m by m.
(iv) Solve the reduced eigenvalue problem $\mathbf{K}_{k+1}\Psi_{k+1} - \mathbf{M}\Psi_{k+1}\Lambda_{k+1} = 0$ for m eigenvalues, which are the diagonal terms of the diagonal matrix Λ_{k+1}, and for eigenvectors Ψ_{k+1}.
(v) Calculate the improved approximation to the eigenvectors of the original system using $\mathbf{X}_{k+1} = \overline{\mathbf{X}}_{k+1}\Psi_{k+1}$.
(vi) Repeat the process until the eigenvalues and eigenvectors converge to the lowest eigenvectors to desired accuracy.

By specifying the line '*FREQUENCY' in the analysis step, ABAQUS will carry out a similar algorithm as that briefly explained above. The line after the '*FREQUENCY' contains some data which ABAQUS uses to aid the procedure. The first entry refers to the number of eigenvalues required (in this case, 3). The second refers to the maximum frequency of interest. This will limit the frequency range, and therefore anything beyond this frequency will not be calculated. In this case, no maximum frequency range will be specified. The third is to specify shift points, which is used to ensure that the stiffness matrix is not singular. Here, again, it is left blank since it is not necessary. The fourth is the number of columns of the starting matrix $\mathbf{X}_1$ to be used. It is left blank again, and thus ABAQUS will use its default setting. The last entry is the number of iterations, which in this case is 30.

Output control cards are used for selecting the data that needs to be output. This is very useful for large scale computation that produces huge data files; one needs to limit the output to what is really needed.

Once the input file is created, one can then invoke ABAQUS to execute the analysis, and the results will be written into an output file that can be read by the post-processor.

5.5.4 Result and Discussion

Using the above input file, an analysis to calculate the eigenvalues, and hence the natural resonant frequencies of the bridge structure, is carried out using ABAQUS. Other than the 10-element mesh as shown in Figure 5.6, which is also depicted in the input file, a simple

Table 5.3. Resonant frequencies of bridge using FEA and analytical calculations

Number of two-node beam elements	Natural frequency (Hz)		
	Mode 1	Mode 2	Mode 3
10	4.4058×10^5	1.2148×10^6	2.3832×10^6
20	4.4057×10^5	1.2145×10^6	2.3809×10^6
40	4.4056×10^5	1.2144×10^6	2.3808×10^6
60	4.4056×10^5	1.2144×10^6	2.3808×10^6
Analytical calculations	4.4051×10^5	1.2143×10^6	2.3805×10^6

convergence test is carried out. Hence, there are similar uniform meshes using 20, 40 and 60 elements. All the frequencies obtained are given in Table 5.3. Because the clamped–clamped beam structure is a simple problem, it is possible to evaluate the natural frequencies analytically. The results obtained from analytical calculations are also shown in Table 5.3 for comparison.

From the table, it can be seen that the finite element results give very good approximations as compared to the analytical results. Even with just 10 elements, the error of mode 1 frequency is about 0.016% from the analytical calculations. It can also be seen that as the number of elements increases, the finite element results gets closer and closer to the analytical calculations, and converges such that the results obtained for 40 and 60 elements show no difference up to the fourth decimal place. What this implies is that in finite element analyses, the finer the mesh or the greater the number of elements used, the more accurate the results. However, using more elements will use up more computer resources, and it will take a longer time to execute. Hence, it is advised to use the minimum number of elements which give the results of desired accuracy.

Other than the resonant frequencies, the mode shapes can also be obtained. Mode shapes can be considered to be the way in which the structure vibrates at a particular natural frequency. It corresponds to the eigenvector of the finite element equation, just like the resonant frequencies corresponds to the eigenvalues of the finite element equation. Mode shapes can be important in some applications, where the points of zero displacements, like the centre of the beam in Figure 5.8, need to be identified for the installation of devices which should not undergo huge vibration.

The data for constructing the mode shape for each eigenvalue or natural frequency can be obtained from the displacement output for that natural frequency. Figures 5.7 to Figure 5.9 show the mode shapes obtained by plotting the displacement components using 10 elements. The figures show how the clamped–clamped beam will vibrate at the natural frequencies. Note that the output data usually consists of only the output at the nodes, and these are then used by the post-processor or any graph plotting applications to form a smooth curve. Most post-processors thus contain curve fitting functions to properly plot the curves using the data values.

This simple case study points out some of the basic requirements needed in a finite element analysis. Like ABAQUS, most finite element software works on the same finite

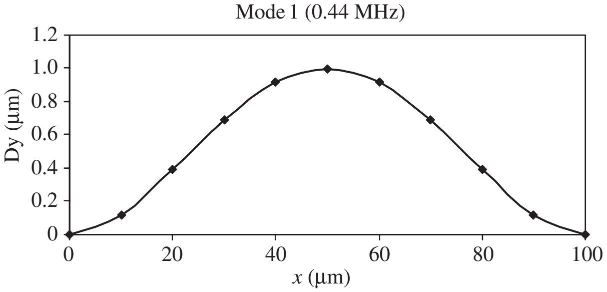


Figure 5.7. Mode 1 using 10 elements at 4.4285 × 10^5 Hz.

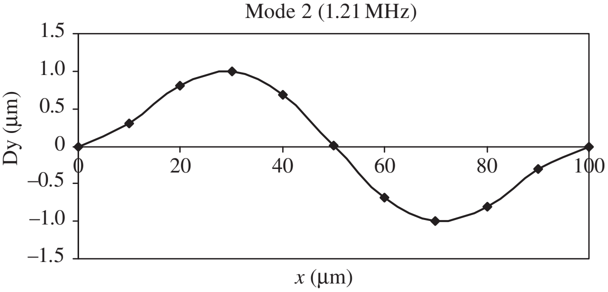


Figure 5.8. Mode 2 using 10 elements at 1.2284 × 10^6 Hz.

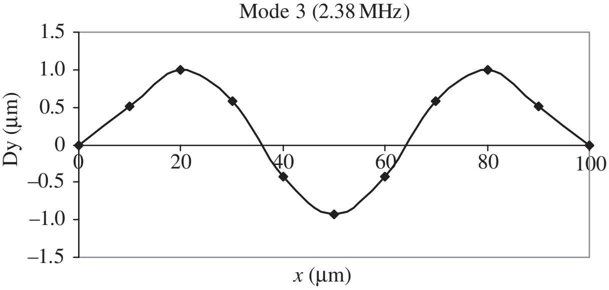


Figure 5.9. Mode 3 using 10 elements at 2.4276 × 10^6 Hz.

element principles. All that is needed is just to provide the necessary information for the software to use the necessary type of elements, and hence the shape functions; to build up the necessary element matrices; followed by the assembly of all the elements to form the global matrices; and finally, to solve the finite element equations.

5.6 REVIEW QUESTIONS

1. How would you formulate a beam element that also carries axial forces?
2. Calculate the force vector for a simply supported beam subjected to a vertical force at the middle of the span, when only one beam element is used to model the beam, as shown in Figure 5.10.

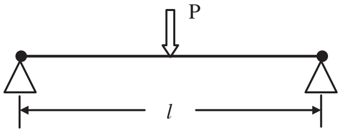


Figure 5.10. Simply supported beam modelled using one beam element.

3. Calculate the force vector for a simply supported beam subjected to a vertical force at the middle of the span, when two beam elements of equal length are used to model the beam, as shown in Figure 5.11.

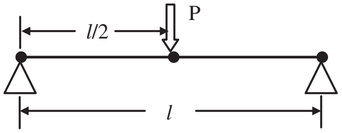


Figure 5.11. Simply supported beam modelled using two beam elements.

4. For a cantilever beam subjected to a vertical force at its free end, how many elements should be used to obtain the exact solution for the deflection of the beam?

6

FEM FOR FRAMES

6.1 INTRODUCTION

A frame element is formulated to model a straight *bar* of an arbitrary cross-section, which can deform not only in the axial direction but also in the directions perpendicular to the axis of the bar. The bar is capable of carrying both axial and transverse forces, as well as moments. Therefore, a frame element is seen to possess the properties of both truss and beam elements. In fact, the frame structure can be found in most of our real world structural problems, for there are not many structures that deform and carry loadings purely in axial directions nor purely in transverse directions. The development of FEM equations for beam elements facilitates the development of FEM equations for frame structures in this chapter.

The frame element developed is also known in many commercial software packages as the *general beam element*, or even simply the *beam element*. Commercial software packages usually offer both pure beam and frame elements, but frame structures are more often used in actual engineering applications. A three-dimensional spatial frame structure can practically take forces and moments of all directions. Hence, it can be considered to be the most general form of element with a one-dimensional geometry.

Frame elements are applicable for the analysis of skeletal type systems of both *planar frames* (two-dimensional frames) and *space frames* (three-dimensional frames). A typical three-dimensional frame structure is shown in Figure 6.1. Frame members in a frame structure are joined together by welding so that both forces and moments can be transmitted between members. In this book, it is assumed that the frame elements have a uniform cross-sectional area. If a structure of varying cross-section is to be modelled using the formulation in this chapter, then it is advised that the structure is to be divided into smaller elements of different constant cross-sectional area so as to simulate the varying cross-section. Of course, if the variation in cross-section is too severe for accurate approximation, then the equations for a varying cross-sectional area can also be formulated without much difficulty using the same concepts and procedure given in this chapter. The basic concepts, procedures and formulations can also be found in many existing textbooks (see, e.g. Petyt,1990; Rao, 1999; etc.).

Figure 6.1. Example of a space frame structure.

6.2 FEM EQUATIONS FOR PLANAR FRAMES

Consider a frame structure whereby the structure is divided into frame elements connected by nodes. Each element is of length $l_e = 2a$, and has two nodes at its two ends. The elements and nodes are numbered separately in a convenient manner. In a planar frame element, there are three degrees of freedom (DOFs) at one node in its local coordinate system, as shown in Figure 6.2. They are the axial deformation in the x direction, u; deflection in the y direction, v; and the rotation in the x–y plane and with respect to the z-axis, θ_z. Therefore, each element with two nodes will have a total of six DOFs.

6.2.1 Equations in Local Coordinate System

Considering the frame element shown in Figure 6.2 with nodes labelled 1 and 2 at each end of the element, it can be seen that the local x-axis is taken as the axial direction of the element with its origin at the middle of the element. As mentioned, a frame element contains both the properties of the truss element and the beam element. Therefore, the element matrices for a frame element can be simply formulated by combining element matrices for truss and

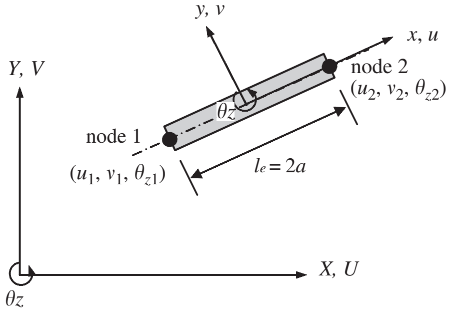


Figure 6.2. Planar frame element and the DOFs.

beam elements, without going through the detailed process of formulating shape functions and using the constitutive equations for a frame. Recall that the truss element has only one degree of freedom at each node (axial deformation), and the beam element has two degrees of freedom at each node (transverse deformation and rotation). Combining these will give the degrees of freedom of a frame element, and the element displacement vector for a frame element can thus be written as

$$
\mathbf{d}_e = \begin{Bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{Bmatrix} = \begin{Bmatrix} u_1 \\ v_1 \\ \theta_{z1} \\ u_2 \\ v_2 \\ \theta_{z2} \end{Bmatrix} \begin{matrix} \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} \text{displacement components at node 1} \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} \text{displacement components at node 2} \end{matrix} \tag{6.1}
$$

To construct the stiffness matrix, the stiffness matrix for truss elements, Eq. (4.16), is first extended to a 6 × 6 matrix corresponding to the order of the degrees of freedom of the truss element in the element displacement vector in Eq. (6.1):

$$
\mathbf{k}_e^{\text{truss}} = \begin{array}{cc}
\begin{matrix} d_1 = u_1 & & & d_4 = u_2 & & \\ \uparrow & & & \uparrow & & \end{matrix} & \\
\begin{bmatrix} AE/(2a) & 0 & 0 & -AE/(2a) & 0 & 0 \\ & 0 & 0 & 0 & 0 & 0 \\ & & 0 & 0 & 0 & 0 \\ & & & AE/(2a) & 0 & 0 \\ & sy. & & & 0 & 0 \\ & & & & & 0 \end{bmatrix} & \begin{matrix} \rightarrow d_1 = u_1 \\ \\ \\ \rightarrow d_4 = u_2 \\ \\ \\ \end{matrix}
\end{array} \tag{6.2}
$$

Next, the stiffness matrix for the beam element, Eq. (5.21), is also extended to a 6 × 6 matrix corresponding to the order of the degrees of freedom of the beam element in Eq. (6.1):

$$\mathbf{k}_e^{\text{beam}} = \begin{array}{cccccccc}
 & d_2(v_1) & d_3(\theta_{z1}) & & d_5(v_2) & d_6(\theta_{z2}) & & \\
 & \uparrow & \uparrow & & \uparrow & \uparrow & & \\
\lceil\, 0 & 0 & 0 & 0 & 0 & 0 \,\rceil & \\
 & \frac{3EI_z}{2a^3} & \frac{3EI_z}{2a^2} & 0 & -\frac{3EI_z}{2a^3} & \frac{3EI_z}{2a^2} & & \rightarrow d_2 = v_1 \\
 & & \frac{2EI_z}{a} & 0 & -\frac{3EI_z}{2a^2} & \frac{EI_z}{a} & & \rightarrow d_3 = \theta_{z1} \\
 & & & 0 & 0 & 0 & & \\
 & sy. & & & \frac{3EI_z}{2a^3} & -\frac{3EI_z}{2a^2} & & \rightarrow d_5 = v_2 \\
\lfloor & & & & & \frac{2EI_z}{a} \,\rfloor & & \rightarrow d_6 = \theta_{z2}
\end{array} \qquad (6.3)$$

The two matrices in Eqs. (6.2) and (6.3) are now superimposed together to obtain the stiffness matrix for the frame element:

$$\mathbf{k}_e = \begin{bmatrix}
\frac{AE}{2a} & 0 & 0 & -\frac{AE}{2a} & 0 & 0 \\
 & \frac{3EI_z}{2a^3} & \frac{3EI_z}{2a^2} & 0 & -\frac{3EI_z}{2a^3} & \frac{3EI_z}{2a^2} \\
 & & \frac{2EI_z}{a} & 0 & -\frac{3EI_z}{2a^2} & \frac{EI_z}{a} \\
 & & & \frac{AE}{2a} & 0 & 0 \\
 & sy. & & & \frac{3EI_z}{2a^3} & -\frac{3EI_z}{2a^2} \\
 & & & & & \frac{2EI_z}{a}
\end{bmatrix} \qquad (6.4)$$

The element mass matrix of the frame element can also be obtained in the same way as the stiffness matrix. The element mass matrices for the truss element and the beam element, Eqs. (4.17) and (5.23), respectively, are extended into 6 × 6 matrices and added together to give the element mass matrix for the frame element:

$$\mathbf{m}_e = \frac{\rho A a}{105} \begin{bmatrix}
70 & 0 & 0 & 35 & 0 & 0 \\
 & 78 & 22a & 0 & 27 & -13a \\
 & & 8a^2 & 0 & 13a & -6a^2 \\
 & & & 70 & 0 & 0 \\
 & sy. & & & 78 & -22a \\
 & & & & & 8a^2
\end{bmatrix} \qquad (6.5)$$

The same simple procedure can be applied to the force vector as well. The element force vectors for the truss and beam elements, Eqs. (4.18) and (5.24), respectively, are extended into 6 × 1 vectors corresponding to their respective DOFs and added together. If the element is loaded by external distributed forces f_x and f_y along the x-axis; concentrated forces f_{sx1}, f_{sx2}, f_{sy1} and f_{sy2}; and concentrated moments m_{s1} and m_{s2}, respectively, at nodes 1 and 2, the total nodal force vector becomes

$$\mathbf{f}_e = \begin{Bmatrix} f_x a + f_{sx1} \\ f_y a + f_{sy1} \\ f_y a^2/3 + m_{s1} \\ f_x a + f_{sx2} \\ f_y a + f_{sy2} \\ -f_y a^2/3 + m_{s1} \end{Bmatrix} \tag{6.6}$$

The final FEM equation will thus have the form of Eq. (3.89) with the element matrices having the forms in Eqs. (6.4) to (6.6).

6.2.2 Equations in Global Coordinate System

The matrices formulated in the previous section are for a particular frame element in a specific orientation. A full frame structure usually comprises numerous frame elements of different orientations joined together. As such, their local coordinate system would vary from one orientation to another. To assemble the element matrices together, all the matrices must first be expressed in a common coordinate system, which is the global coordinate system. The coordinate transformation process is the same as that discussed in Chapter 4 for truss structures.

Assume that local nodes 1 and 2 correspond to the global nodes i and j, respectively. The displacement at a local node should have two translational components in the x and y directions and one rotational deformation. They are numbered sequentially by u, v and θ_z at each of the two nodes, as shown in Figure 6.3. The displacement at a global node should also have two translational components in the X and Y directions and one rotational deformation. They are numbered sequentially by D_{3i-2}, D_{3i-1} and D_{3i} for the ith node, as shown in Figure 6.3. The same sign convention also applies to node j. The coordinate transformation gives the relationship between the displacement vector $\mathbf{d}_e$ based on the local coordinate system and the displacement vector $\mathbf{D}_e$ for the same element, but based on the global coordinate system:

$$\mathbf{d}_e = \mathbf{T}\mathbf{D}_e \tag{6.7}$$

where

$$\mathbf{D}_e = \begin{Bmatrix} D_{3i-2} \\ D_{3i-1} \\ D_{3i} \\ D_{3j-2} \\ D_{3j-1} \\ D_{3j} \end{Bmatrix} \tag{6.8}$$

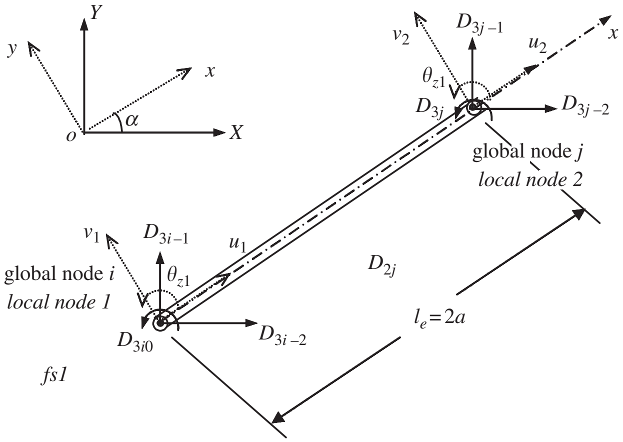


Figure 6.3. Coordinate transformation for 2D frame elements.

and **T** is the transformation matrix for the frame element given by

$$\mathbf{T} = \begin{bmatrix} l_x & m_x & 0 & 0 & 0 & 0 \\ l_y & m_y & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & l_x & m_x & 0 \\ 0 & 0 & 0 & l_y & m_y & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{6.9}$$

in which

$$\begin{aligned} l_x &= \cos(x, X) = \cos\alpha = \frac{X_j - X_i}{l_e} \\ m_x &= \cos(x, Y) = \sin\alpha = \frac{Y_j - Y_i}{l_e} \end{aligned} \tag{6.10}$$

where α is the angle between the x-axis and the X-axis, as shown in Figure 6.3, and

$$\begin{aligned} l_y &= \cos(y, X) = \cos(90^\circ + \alpha) = -\sin\alpha = -\frac{Y_j - Y_i}{l_e} \\ m_y &= \cos(y, Y) = \cos\alpha = \frac{X_j - X_i}{l_e} \end{aligned} \tag{6.11}$$

Note that the coordinate transformation in the X–Y plane does not affect the rotational DOF, as its direction is in the z direction (normal to the x–y plane), which still remains the same as the Z direction in the global coordinate system. The length of the element, l_e, can be calculated by

$$l_e = \sqrt{(X_j - X_i)^2 + (Y_j - Y_i)^2} \tag{6.12}$$

Equation (6.7) can be easily verified, as it simply says that at node i, u_1 equals the summation of all the projections of D_{3i-2} and D_{3i-1} onto the local x axis, and v_1 equals the summation of all the projections of D_{3i-2} and D_{3i-1} onto the local y axis. The same can be said at node j. The matrix **T** for a frame element transforms a 6 × 6 matrix into another 6 × 6 matrix. Using the transformation matrix, **T**, the matrices for the frame element in the global coordinate system become

$$\mathbf{K}_e = \mathbf{T}^T \mathbf{k}_e \mathbf{T} \tag{6.13}$$

$$\mathbf{M}_e = \mathbf{T}^T \mathbf{m}_e \mathbf{T} \tag{6.14}$$

$$\mathbf{F}_e = \mathbf{T}^T \mathbf{f}_e \tag{6.15}$$

Note that there is no change in dimension between the matrices and vectors in the local and global coordinate systems.

6.3 FEM EQUATIONS FOR SPACE FRAMES

6.3.1 Equations in Local Coordinate System

The approach used to develop the two-dimensional frame elements can be used to develop the three-dimensional frame elements as well. The only difference is that there are more DOFs at a node in a 3D frame element than there are in a 2D frame element. There are altogether six DOFs at a node in a 3D frame element: three translational displacements in the x, y and z directions, and three rotations with respect to the x, y and z axes. Therefore, for an element with two nodes, there are altogether twelve DOFs, as shown in Figure 6.4.

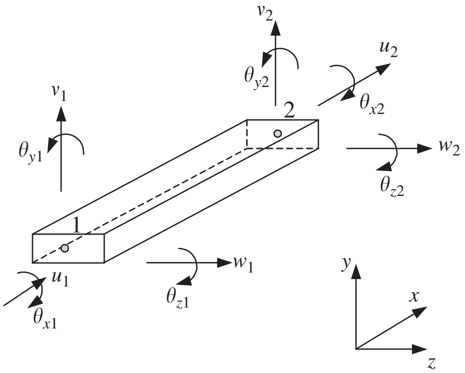


Figure 6.4. Frame element in space with twelve DOFs.

The element displacement vector for a frame element in space can be written as

$$\mathbf{d}_e = \begin{Bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \\ d_9 \\ d_{10} \\ d_{11} \\ d_{12} \end{Bmatrix} = \begin{Bmatrix} u_1 \\ v_1 \\ w_1 \\ \theta_{x1} \\ \theta_{y1} \\ \theta_{z1} \\ u_2 \\ v_2 \\ w_2 \\ \theta_{x2} \\ \theta_{y2} \\ \theta_{z2} \end{Bmatrix} \begin{matrix} \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \end{matrix}\right\} \text{displacement components at node 1} \\ \left.\begin{matrix} \\ \\ \\ \\ \\ \\ \end{matrix}\right\} \text{displacement components at node 2} \end{matrix} \qquad (6.16)$$

The element matrices can be obtained by a similar process of obtaining the matrices of the truss element in space and that of beam elements, and adding them together. Because of the huge matrices involved, the details will not be shown in this book, but the stiffness matrix is listed here as follows, and can be easily confirmed simply by inspection:

$$\mathbf{k}_e = \begin{array}{c} \begin{array}{cccccccccccc} u_1 & v_1 & w_1 & \theta_{x1} & \theta_{y1} & \theta_{z1} & u_2 & v_2 & w_2 & \theta_{x2} & \theta_{y2} & \theta_{z2} \\ \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \end{array} \\ \left[\begin{array}{cccccccccccc} \frac{AE}{2a} & 0 & 0 & 0 & 0 & 0 & \frac{-AE}{2a} & 0 & 0 & 0 & 0 & 0 \\ & \frac{3EI_z}{2a^3} & 0 & 0 & 0 & \frac{3EI_z}{2a^2} & 0 & \frac{-3EI_z}{2a^3} & 0 & 0 & 0 & \frac{3EI_z}{2a^2} \\ & & \frac{3EI_y}{2a^3} & 0 & \frac{-3EI_y}{2a^2} & 0 & 0 & 0 & \frac{-3EI_y}{2a^3} & 0 & \frac{-3EI_y}{2a^2} & 0 \\ & & & \frac{GJ}{2a} & 0 & 0 & 0 & 0 & 0 & \frac{-GJ}{2a} & 0 & 0 \\ & & & & \frac{2EI_y}{a} & 0 & 0 & 0 & \frac{3EI_y}{2a^2} & 0 & \frac{EI_y}{a} & 0 \\ & & & & & \frac{2EI_z}{a} & 0 & \frac{-3EI_z}{2a^2} & 0 & 0 & 0 & \frac{EI_z}{a} \\ & & & & & & \frac{AE}{2a} & 0 & 0 & 0 & 0 & 0 \\ & & & & & & & \frac{3EI_z}{2a^3} & 0 & 0 & 0 & \frac{-3EI_z}{2a^2} \\ & & sy. & & & & & & \frac{3EI_y}{2a^3} & 0 & \frac{3EI_y}{2a^2} & 0 \\ & & & & & & & & & \frac{GJ}{2a} & 0 & 0 \\ & & & & & & & & & & \frac{2EI_y}{a} & 0 \\ & & & & & & & & & & & \frac{2EI_z}{a} \end{array}\right] \end{array} \qquad (6.17)$$

where I_y and I_z are the second moment of area (or moment of inertia) of the cross-section of the beam with respect to the y and z axes, respectively. Note that the fourth DOF is related to the *torsional* deformation. The development of a *torsional element* of a bar is very much the same as that for a truss element. The only difference is that the axial deformation is replaced by the torsional angular deformation, and axial force is replaced by torque. Therefore, in the resultant stiffness matrix, the element tensile stiffness AE/l_e is replaced by the element torsional stiffness GJ/l_e, where G is the shear modules and J is the polar moment of inertia of the cross-section of the bar.

The mass matrix is also shown as follows:

$$\mathbf{m}_e = \frac{\rho A a}{105}\begin{bmatrix}
70 & 0 & 0 & 0 & 0 & 0 & 35 & 0 & 0 & 0 & 0 & 0 \\
 & 78 & 0 & 0 & 0 & 22a & 0 & 27 & 0 & 0 & 0 & -13a \\
 & & 78 & 0 & -22a & 0 & 0 & 0 & 27 & 0 & 13a & 0 \\
 & & & 70r_x^2 & 0 & 0 & 0 & 0 & 0 & -35r_x^2 & 0 & 0 \\
 & & & & 8a^2 & 0 & 0 & 0 & -13a & 0 & -6a^2 & 0 \\
 & & & & & 8a^2 & 0 & 13a & 0 & 0 & 0 & -6a^2 \\
 & & & & & & 70 & 0 & 0 & 0 & 0 & 0 \\
 & & & & & & & 78 & 0 & 0 & 0 & -22a \\
 & & & & & & & & 78 & 0 & 22a & 0 \\
 & & & sy. & & & & & & 70r_x^2 & 0 & 0 \\
 & & & & & & & & & & 8a^2 & 0 \\
 & & & & & & & & & & & 8a^2
\end{bmatrix} \tag{6.18}$$

where

$$r_x^2 = \frac{I_x}{A} \tag{6.19}$$

in which I_x is the second moment of area (or moment of inertia) of the cross-section of the beam with respect to the x axis.

6.3.2 Equations in Global Coordinate System

Having known the element matrices in the local coordinate system, the next thing to do is to transform the element matrices into the global coordinate system to account for the differences in orientation of all the local coordinate systems that are attached on individual frame members.

Assume that the local nodes 1 and 2 of the element correspond to global nodes i and j, respectively. The displacement at a local node should have three translational components in the x, y and z directions, and three rotational components with respect to the x, y and z-axes. They are numbered sequentially by d_1–d_{12} corresponding to the physical deformations as defined by Eq. (6.16). The displacement at a global node should also have three translational components in the X, Y and Z directions, and three rotational components with respect to the X, Y and Z axes. They are numbered sequentially by D_{6i-5}, D_{6i-4}, ..., and D_{6i} for the ith node, as shown in Figure 6.5. The same sign convention applies to node j. The coordinate transformation gives the relationship between the displacement vector $\mathbf{d}_e$ based on the local coordinate system and the displacement vector $\mathbf{D}_e$ for the same element but

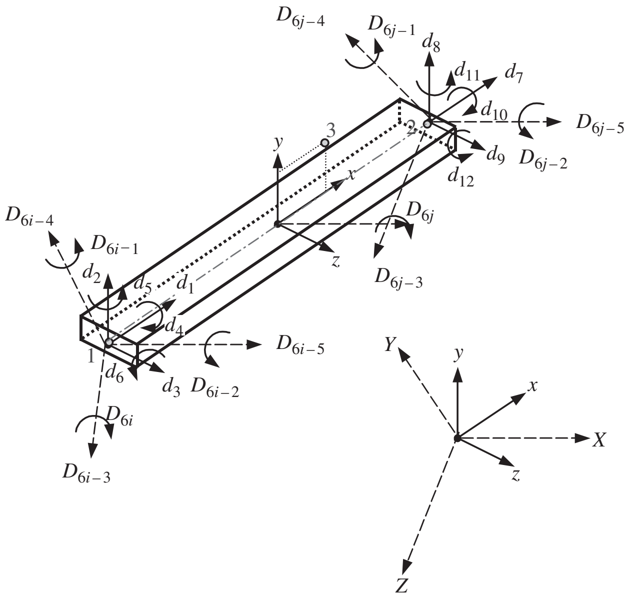


Figure 6.5. Coordinate transformation for a frame element in space.

based on the global coordinate system:

$$\mathbf{d}_e = \mathbf{T}\mathbf{D}_e \tag{6.20}$$

where

$$\mathbf{D}_e = \begin{Bmatrix} D_{6i-5} \\ D_{6i-4} \\ D_{6i-3} \\ D_{6i-2} \\ D_{6i-1} \\ D_{6i} \\ D_{6j-5} \\ D_{6j-4} \\ D_{6j-3} \\ D_{6j-2} \\ D_{6j-1} \\ D_{6j} \end{Bmatrix} \tag{6.21}$$

and **T** is the transformation matrix for the truss element given by

$$\mathbf{T} = \begin{bmatrix} \mathbf{T}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{T}_3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{T}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{T}_3 \end{bmatrix} \tag{6.22}$$

in which

$$\mathbf{T}_3 = \begin{bmatrix} l_x & m_x & n_x \\ l_y & m_y & n_y \\ l_z & m_z & n_z \end{bmatrix} \tag{6.23}$$

where l_k, m_k and n_k $(k = x, y, z)$ are direction cosines defined by

$$\begin{aligned} l_x &= \cos(x, X), \quad m_x = \cos(x, Y), \quad n_x = \cos(x, Z) \\ l_y &= \cos(y, X), \quad m_y = \cos(y, Y), \quad n_y = \cos(y, Z) \\ l_z &= \cos(z, X), \quad m_z = \cos(z, Y), \quad n_z = \cos(z, Z) \end{aligned} \tag{6.24}$$

To define these direction cosines, the position and the three-dimensional orientation of the frame element have to be defined first. With nodes 1 and 2, the location of the element is fixed on the local coordinate frame, and the orientation of the element has also been fixed in the x direction. However, the local coordinate frame can still rotate about the axis of the beam. One more additional point in the local coordinate has to be defined. This point can be chosen anywhere in the local x–y plane, but not on the x-axis. Therefore, node 3 is chosen, as shown in Figure 6.6.

The position vectors $\vec{V}_1$, $\vec{V}_2$ and $\vec{V}_3$ can be expressed as

$$\vec{V}_1 = X_1\vec{X} + Y_1\vec{Y} + Z_1\vec{Z} \tag{6.25}$$

$$\vec{V}_2 = X_2\vec{X} + Y_2\vec{Y} + Z_2\vec{Z} \tag{6.26}$$

$$\vec{V}_3 = X_3\vec{X} + Y_3\vec{Y} + Z_3\vec{Z} \tag{6.27}$$

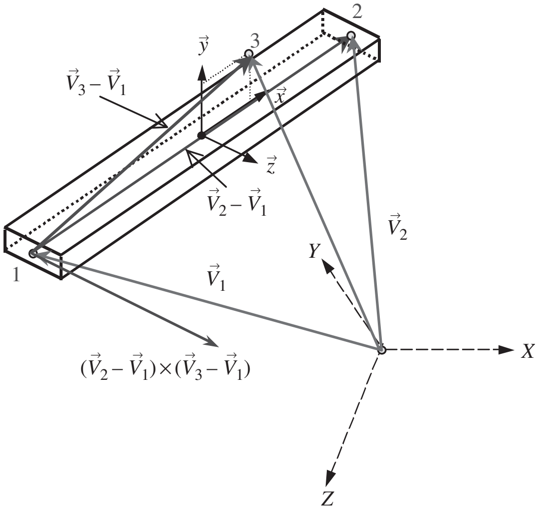


Figure 6.6. Vectors for defining the location and three-dimensional orientation of the frame element in space.

where X_k, Y_k and Z_k ($k = 1, 2, 3$) are the coordinates for node k, and $\vec{X}$, $\vec{Y}$ and $\vec{Z}$ are unit vectors along the X, Y and Z axes. We now define

$$\left.\begin{aligned} X_{kl} &= X_k - X_l \\ Y_{kl} &= Y_k - Y_l \\ Z_{kl} &= Z_k - Z_l \end{aligned}\right\} \quad k, l = 1, 2, 3 \tag{6.28}$$

Vectors $(\vec{V}_2 - \vec{V}_1)$ and $(\vec{V}_3 - \vec{V}_1)$ can thus be obtained using Eqs. (6.25) to (6.28) as follows:

$$\vec{V}_2 - \vec{V}_1 = X_{21}\vec{X} + Y_{21}\vec{Y} + Z_{21}\vec{Z} \tag{6.29}$$

$$\vec{V}_3 - \vec{V}_1 = X_{31}\vec{X} + Y_{31}\vec{Y} + Z_{31}\vec{Z} \tag{6.30}$$

The length of the frame element can be obtained by

$$l_e = 2a = \left|\vec{V}_2 - \vec{V}_1\right| = \sqrt{X_{21}^2 + Y_{21}^2 + Z_{21}^2} \tag{6.31}$$

The unit vector along the x-axis can thus be expressed as

$$\vec{x} = \frac{(\vec{V}_2 - \vec{V}_1)}{\left|\vec{V}_2 - \vec{V}_1\right|} = \frac{X_{21}}{2a}\vec{X} + \frac{Y_{21}}{2a}\vec{Y} + \frac{Z_{21}}{2a}\vec{Z} \tag{6.32}$$

Therefore, the direction cosines in the x direction are given as

$$\begin{aligned} l_x &= \cos(x, X) = \vec{x} \cdot \vec{X} = \frac{X_{21}}{2a} \\ m_x &= \cos(x, Y) = \vec{x} \cdot \vec{Y} = \frac{Y_{21}}{2a} \\ n_x &= \cos(x, Z) = \vec{x} \cdot \vec{Z} = \frac{Z_{21}}{2a} \end{aligned} \tag{6.33}$$

From Figure 6.6, it can be seen that the direction of the z-axis can be defined by the cross product of vectors $(\vec{V}_2 - \vec{V}_1)$ and $(\vec{V}_3 - \vec{V}_1)$. Hence, the unit vector along the z-axis can be expressed as

$$\vec{z} = \frac{(\vec{V}_2 - \vec{V}_1) \times (\vec{V}_3 - \vec{V}_1)}{\left|(\vec{V}_2 - \vec{V}_1) \times (\vec{V}_3 - \vec{V}_1)\right|} \tag{6.34}$$

Substituting Eqs. (6.29) and (6.30) into the above equation,

$$\vec{z} = \frac{1}{2A_{123}}\{(Y_{21}Z_{31} - Y_{31}Z_{21})\vec{X} + (Z_{21}X_{31} - Z_{31}X_{21})\vec{Y} + (X_{21}Y_{31} - X_{31}Y_{21})\vec{Z}\} \tag{6.35}$$

where

$$A_{123} = \sqrt{(Y_{21}Z_{31} - Y_{31}Z_{21})^2 + (Z_{21}X_{31} - Z_{31}X_{21})^2 + (X_{21}Y_{31} - X_{31}Y_{21})^2} \tag{6.36}$$

Using Eq. (6.35), it is found that

$$\begin{aligned} l_z &= \vec{z} \cdot \vec{X} = \frac{1}{2A_{123}}(Y_{21}Z_{31} - Y_{31}Z_{21}) \\ m_z &= \vec{z} \cdot \vec{Y} = \frac{1}{2A_{123}}(Z_{21}X_{31} - Z_{31}X_{21}) \\ n_z &= \vec{z} \cdot \vec{Z} = \frac{1}{2A_{123}} + (X_{21}Y_{31} - X_{31}Y_{21}) \end{aligned} \tag{6.37}$$

Since the y-axis is perpendicular to both the x-axis and the z-axis, the unit vector along the y-axis can be obtained by cross product,

$$\vec{y} = \vec{z} \times \vec{x} \tag{6.38}$$

which gives

$$\begin{aligned} l_y &= m_z n_x - n_z m_x \\ m_y &= n_z l_x - l_z n_x \\ n_y &= l_z m_x - m_z l_x \end{aligned} \tag{6.39}$$

in which l_x, m_x, n_x, l_z, m_z and n_z have been obtained using Eqs. (6.33) and (6.37).

Using the transformation matrix, **T**, the matrices for space frame elements in the global coordinate system can be obtained as

$$\mathbf{K}_e = \mathbf{T}^T \mathbf{k}_e \mathbf{T} \tag{6.40}$$

$$\mathbf{M}_e = \mathbf{T}^T \mathbf{m}_e \mathbf{T} \tag{6.41}$$

$$\mathbf{F}_e = \mathbf{T}^T \mathbf{f}_e \tag{6.42}$$

6.4 REMARKS

In the formulation of the matrices for the frame element in this chapter, the superposition of the truss element and the beam element has been used. This technique assumes that the axial effects are not coupled with the bending effects in the element. What this means simply is that the axial forces applied on the element will not result in any bending deformation, and the bending forces will not result in any axial deformation. Frame elements can also be used for frame structures with curved members. In such cases, the coupling effects can exist even in the elemental level. Therefore, depending on the curvature of the member, the meshing of the structure can be very important. For example, if the curvature is very large resulting in a significant coupling effect, a finer mesh is required to provide the necessary accuracy.

In practical structures, it is very rare to have beam structures subjected to purely transverse loading. Most skeletal structures are either trusses or frames that carry both axial and transverse loads. It can now be seen that the beam element, developed in Chapter 5, as well as the truss element, developed in Chapter 4, are simply specific cases of the frame element. Therefore, in most commercial software packages, including ABAQUS, the frame element is just known generally as the beam element.

The beam element formulated in Chapter 5, or general beam element formulated in this chapter, is based on so-called Euler–Bernoulli beam theory that is suitable for thin beams with a small thickness to pan ratio ($<1/20$). For *thick* or *deep* beams of a large thickness to pan ratio, corresponding beam theories should be used to develop thick beam elements. The procedure of developing thick beams is very similar to that of developing thick plates, to be discussed in Chapter 8. Most commercial software packages also offer thick beam elements, and the use of these elements is much the same as the thin beam elements.

6.5 CASE STUDY: FINITE ELEMENT ANALYSIS OF A BICYCLE FRAME

In the design of many modern devices and equipment, the finite element method has become an indispensable tool for the many successful products that we have come to use daily. In this case study, the analysis of a bicycle frame is carried out. Historically, intuition and trial-and-error in physical prototyping testing processes have played important roles in coming out with the evolution of today's diamond-shaped bicycle frame, as shown in Figure 6.7. However, using such a physical trial-and-error design procedure is costly and time consuming, and has its limitations, especially when new materials are introduced and when new applications or demands are placed on the structure. Hence, there is the need to

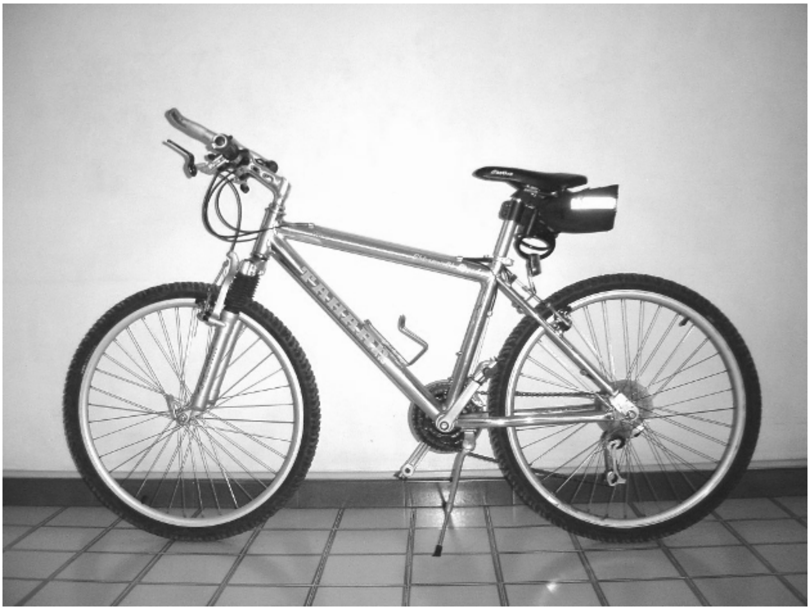


Figure 6.7. Diamond-shaped bicycle frame.

use the finite element tool to help the designer come up with reliable properties in the design to meet the demands expected by the consumers. Using the finite element method to perform a virtual prototyping instead of a physical prototyping can save lots of cost, time and effort.

There can be numerous factors to consider when it comes to designing a bicycle frame. For example, factors like the weight of the frame, the maximum load the frame can carry; the impact toughness of the frame, and so on. In a finite element analysis, one would require information on the material properties, the boundary conditions and the loading conditions. The loading conditions on a bicycle can be extremely complex, especially when a rider is riding a bicycle and going through different terrain. In this case study we consider a loading condition of a horizontal impact applied to the bicycle, simulating the effect of a low speed, head-on collision into a wall or curb. This case is one of the physical tests that manufacturers have to comply with before a bicycle design is approved.

6.5.1 Modelling

The bicycle frame to be modelled is made of aluminium, whose properties are shown in Table 6.1. The bicycle frame is meshed by two-nodal 'beam elements' in ABAQUS. Note that in ABAQUS, as well as many other software, a general beam element in space is basically the same as the frame element developed in this chapter. The 'beam element' in space offered by ABAQUS has the same DOFs as the frame element in this chapter, that is, three translational DOFs and three rotational DOFs.

Altogether, 74 elements (71 nodes) are used in the skeletal model of the bicycle frame, as shown in Figure 6.8. Note that like all finite element meshes, connectivity at the nodes is very important. It is important to make sure that at the joints, there is connectivity between the elements. No node should be left unattached to another element unless the node happens to be at the end of a structure.

To model the horizontal impact, the two nodes at the rear dropouts are constrained from any displacements and a horizontal force of 1000 N is applied to the front dropout. The forces and constraints are shown in Figure 6.9.

6.5.2 Abaqus Input File

The ABAQUS input file for the above-described finite element model is shown below. The text boxes to the right of the input file are not part of the file, but explain what the sections of the file mean.

Table 6.1. Material properties of aluminium

Young's modulus, E GPa	Poisson's ratio, ν
69.0	0.33

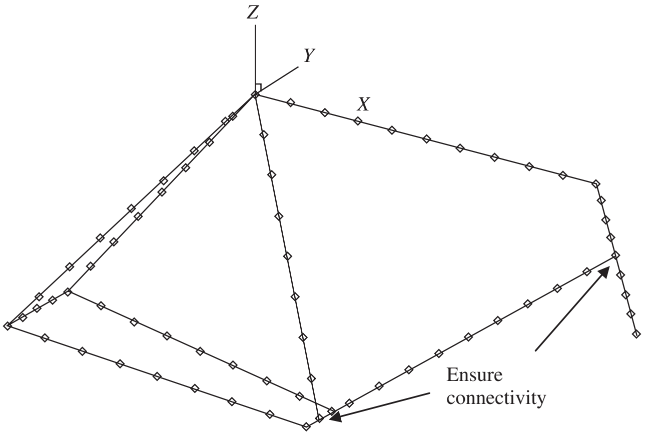


Figure 6.8. Finite element mesh of a bicycle frame.

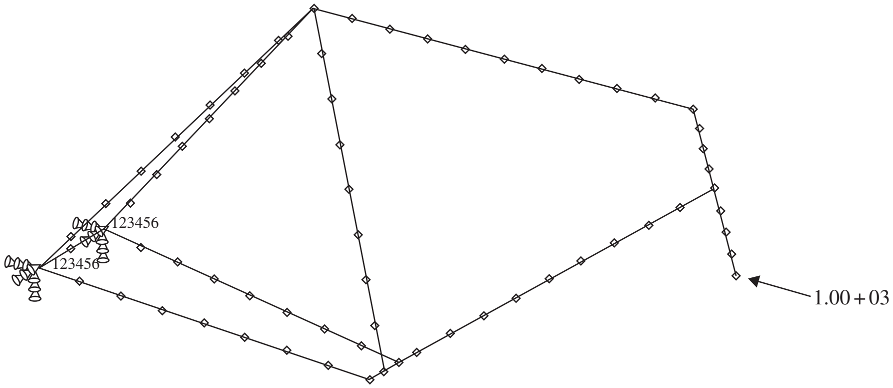


Figure 6.9. Loadings and boundary conditions on the bicycle frame.

```
*HEADING, SPARSE
ABAQUS job to calculate low speed impact on bicycle frame
**
*NODE
1, -0.347, 0., -0.39
2, -0.303625, -0.06125, -0.34125
3, -0.347, -0.035, -0.39
⋮
68, 0.57829, 0., -0.12125
69, 0.586579, 0., -0.1475
70, 0.594868, 0., -0.17375
71, 0.603158, 0., -0.2
**
**
```

Nodal cards

Define the coordinates of the nodes in the model. The first entry being the node ID, while the second, third and fourth entries are the *x*, *y* and *z* coordinates of the position of the node, respectively.

```
*ELEMENT, TYPE=B33 , ELSET=FRAME
1, 34, 38
2, 38, 42
3, 42, 46
4, 46, 48
5, 48, 50
6, 50, 52
⋮
72, 68, 69
73, 69, 70
74, 70, 71
**
```

Element (connectivity) cards

Define the element type and what nodes make up the element. B33 represents spatial, 2-nodal, beam element following the Euler–Bernoulli theory with cubic interpolation. Note that this element has six DOFs per node – three translational and three rotational. There are many other element types in the ABAQUS element library. The “ELSET = FRAME” statement is simply for naming this set of elements so that it can be referenced when defining the material properties. In the subsequent data entry, the first entry is the element ID, and the following two entries are the nodes making up the element.

```
*BEAM SECTION, ELSET=FRAME, MATERIAL=ALU, SECTION=CIRC
0.012
**
*MATERIAL, NAME=ALU
**
*DENSITY
2710.,
**
*ELASTIC, TYPE=ISO
6.9E+10, 0.33
**
```

Material cards

Define material properties under the name “ALU”. Density and elastic properties are defined. TYPE = ISO represents isotropic properties.

Property cards

Define properties to the elements of set “FRAME”. “SECT = CIRC” describes the cross-section as a circle. ABAQUS provides a choice of other cross-sections. The first data line under “BEAM SECTION” defines the geometry of the cross-section. The material is defined by “ALU”.

```
*BOUNDARY, OP=NEW
FIXED, 1,, 0.
FIXED, 2,, 0.
FIXED, 3,, 0.
FIXED, 4,, 0.
FIXED, 5,, 0.
FIXED, 6,, 0.
**
```

Boundary Conditions cards

Define boundary conditions. Nodes belonging to the node set, “FIXED” have all their DOFs constrained (=0).

```
** Step 1, Default Static Step
** LoadCase, Default
**
*STEP, AMPLITUDE=RAMP, PERTURBATION
Linear Static Analysis
**
*STATIC
**
**
**
**
```

Control cards

Indicates the analysis step. In this case it is a “STATIC” analysis.

```
*NSET, NSET=FIXED
8, 9
*NSET, NSET=FORCE
71,
**
**
```

Node sets

Group nodes into node sets “FIXED” and “FORCE”.

```
*CLOAD, OP=NEW
FORCE, 1, -1000.
**
**
*NODE PRINT, FREQ=1
U,
*NODE FILE, FREQ=1
U,
**
*EL PRINT, POS=INTEG, FREQ=1
1
S,
E,
*EL FILE, POS=INTEG, FREQ=1
1
S,
E,
**
*END STEP
```

Loading cards

Define the loads on the nodes according to the respective groups the nodes belong to. For example, in this case, node 71 belonging to "NSET = FORCE" is given a force of −1000 N in the 1 (x) direction.

Output control cards

Define the output required. For example, in this case, we require the nodal output, displacement "U", the stress, "S" and the strain, "E".

The above input file actually looks similar to that in Chapter 5. In fact, most ABAQUS input files have similar formats, only the information provided may be different depending on the problem required to solve.

6.5.3 Solution Processes

The nodal and element cards provide information on the dimensions, position of nodes and the connectivity of the elements. As discussed, these parameters play important roles in the formation of the element stiffness and mass matrices. Note that in the element cards, the element type specified is B33 which represents a general 'beam element' in space with two nodes following the Euler–Bernoulli beam theory for its transverse displacement field. In ABAQUS, this would be exactly the same as the frame element developed in this chapter, since the degrees of freedom also include the axial displacement component. ABAQUS, like most other software, does not have a pure beam element consisting of only the transverse displacement field. This is because not many real structures are a true beam structure without the axial displacement components. Hence, readers should not be confused by the use of beam elements in this case.

Next, the material properties and the cross-sectional geometry and dimensions are provided in the material cards and properties cards, respectively. As can be seen in Eqs. (6.4) and (6.5), the material properties as well as the moment of area is required to compute the stiffness and mass matrices. It is interesting to note that most of the information provided goes into forming the finite element matrices. In fact, that is the basic idea when it comes to applying the finite element method: to form the finite element matrix equation and solve the equation to obtain the required field variables.

The next card would be the definition of the boundary conditions. In this case, the rear dropouts that are represented by nodes 8 and 9 and grouped as a node set, 'FIXED', are constrained in all DOFs. Recall that in this case, since each node has six DOFs, we can actually reduce the size of the matrix here by eliminating 12 rows and columns each, just as we have done in worked examples in Chapters 4 and 5. For the loads, it is given as a concentrated force of −1000 N at node 71 in the X direction. This would be reflected in the force vector, Eq. (6.42).

The control cards control the type of analysis to be performed, which in this case is a static analysis. In static analyses, the static equation $\mathbf{KD} = \mathbf{F}$ is solved for $\mathbf{D}$, which is a vector of the displacements and rotations of the nodes in the model. The method used is usually algorithms like the Gauss elimination method, which is mentioned in Section 3.5.

The last part of the input file would consist of the output control cards, which specify the type of output requested. In this case, the nodal displacement and rotation components (U) and the elemental stress (S) and strain (E) components are specified. With the input file written, one can then invoke ABAQUS to execute the analysis.

6.5.4 Results and Discussion

Using the above input file, the finite element equation is solved and a deformation plot showing how the frame actually deforms under the specified loading is shown in Figure 6.10. The magnitude of the deformation is actually magnified 20× as the true deformation is much too small for viewing purposes. Such deformation plots, or the results containing the displacements of the nodes, are useful to a designer since he or she will then be able to visualize the way in which the frame will deform under specific conditions. Furthermore, knowing the magnitude of deformation also helps to gauge aspects of the frame in the design,

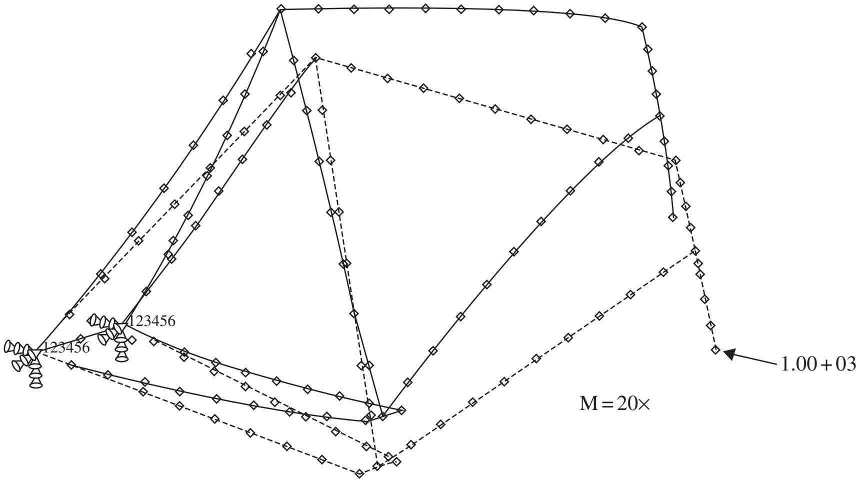


Figure 6.10. Deformation plot of bicycle frame.

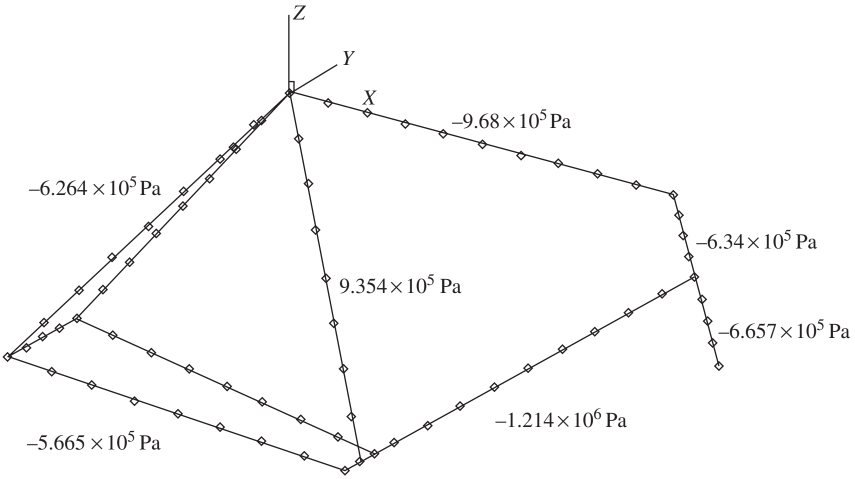


Figure 6.11. Stresses in the bicycle frame.

as consumers would not want a bicycle which could not be ridden once one accidentally hits a wall or curb at slow speed.

The other more important result that could be obtained from this case study would be the stresses that incur with this particular loading condition. Figure 6.11 shows the average axial stresses along the centroid of the aluminum beam members. It should be noted that it is possible to obtain stresses on different sections of the cross-section, for example, the upper or lower surface along the circumference of a circular cross-section. These stresses are useful for testing whether the bicycle frame will fail under the applied loads. If these stresses are smaller than the yield stress or the design stress of the material, then it can be safely concluded that there is a high chance that the material will not fail or undergo plastic deformation. It is important that the deformations remain elastic, that is, it will return to the original shape upon removal of the applied load.

Hence, it can be seen how the finite element method can aid the design engineer when it comes to designing a product. It would be disastrous if a new design of an engineering system is mass-produced without going through any sort of analysis to check its reliability.

6.6 REVIEW QUESTIONS

1. Explain why the superposition technique can be used to formulate the frame elements simply using the formulations of the truss and beam elements. On what conditions will this superimposition technique fail?
2. In the transformation from the local coordinate system to the global coordinate system in a planar frame, does the rotation degree of freedom undergo any transformation? Why?

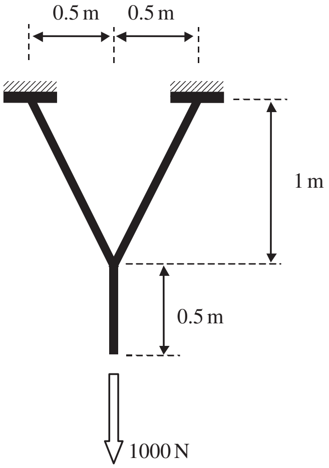


Figure 6.12. Three member planar frame structure.

3. Work out the displacements of the planar frame structure shown in Figure 6.12. All the members are of the same material ($E = 69.0\,\text{GPa}$, $\nu = 0.33$) and with circular cross-sections. The areas of the cross-sections are $0.01\,\text{m}^2$. Compare the results with those obtained for review question 6 in Chapter 4.

7

FEM FOR TWO-DIMENSIONAL SOLIDS

7.1 INTRODUCTION

In this chapter, we develop, in an easy to understand manner, finite element equations for the stress analysis of two-dimensional (2D) solids subjected to external loads. The basic concepts, procedures and formulations can also be found in many existing textbooks (see, e.g. Zienkiewicz and Taylor, 2000). The element developed is called a 2D solid element that is used for structural problems where the loading–and hence the deformation–occur within a plane. Though no real life structure can be truly 2D, experienced analysts can often idealize many practical problems to 2D problems to obtain satisfactory results by carrying out analyses using 2D models, which can be very much more efficient and cost-effective compared to conducting full 3D analyses. In engineering applications, there are ample practical problems that can be modelled as 2D problems. As discussed in Chapter 2, there are plane stress and plane strain problems, whereby correspondingly, plane stress and plane strain elements need to be used to solve them. For example, if we have a plate structure with loading acting in the plane of the plate as in Figure 7.1, we need to use 2D plane stress elements. When we want to model the effects of water pressure on a dam, as shown in Figure 7.2, we have to use 2D plane strain elements.

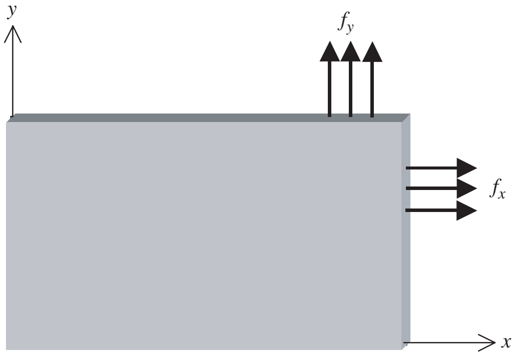


Figure 7.1. A typical 2D plane stress problem.

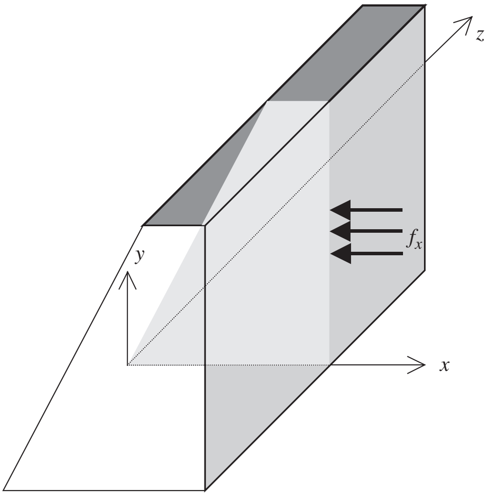


Figure 7.2. A typical 2D plane strain problem.

Note that in Figure 7.1, plane stress conditions are usually applied to structures that have a relatively small thickness as compared to its other dimensions. Due to the absence of any off-plane external force, the normal stresses are negligible, which leads to a plane stress situation. In cases where plane strain conditions are applied, as in Figure 7.2, the thickness of the structure (in the z direction) is relatively large as compared to its other dimensions, and the loading (pressure) is uniform along the elongated direction. The deformation is, therefore, approximated to be the same throughout its thickness. In this case, the off-plane strain (strain components in the z direction) is negligible, which leads to a plane strain situation. In either a plane stress or plane strain situation, the governing system equation can be drastically simplified, as shown in Chapter 2. The formulations for plane stress and plane strain problems are very much the same, except for the difference in the material constant matrix.

A 2D solid element, be it plane strain or plane stress, can be triangular, rectangular or quadrilateral in shape with straight or curved edges. The most often used elements in engineering practice are linear. Quadratic elements are also used for situations that require high accuracy in stress, but they are less often used for practical problems. Higher order elements have also been developed, but they are less often used except for certain specific problems. The order of the 2D element is determined by the order of the shape functions used. A linear element uses linear shape functions, and therefore the edges of the element are straight. A quadratic element uses quadratic shape functions, and their edges can be curved. The same can be said for elements of third order or higher.

In a 2D model, the elements can only deform in the plane where the model is defined, and in most situations, this is the x–y plane. At any point, the variable, that is the displacement, has two components in the x and y directions, and so do the external forces. For plane strain problems, the thickness of the true structure is usually not important, and is normally treated as a unit quantity uniformly throughout the 2D model. However, for plane stress problems, the thickness is an important parameter for computing the stiffness matrix and stresses. Throughout this chapter, it is assumed that the elements have a uniform thickness of h. If the structure to be modelled has a varying thickness, the structure needs to be divided into small elements, where in each element a uniform thickness can be used. On the other hand, formulation of 2D elements with varying thicknesses can also be done easily, as the procedure is similar to that of a uniform element. Very few commercially available software packages provide elements of varying thickness.

The equation system for a 2D element will be more complex as compared with the 1D element because of the higher dimension. The procedure for developing these equations is, however, very similar to that for the 1D truss elements, which is detailed in Chapter 4. These steps can be summarized in the following three-step procedure:

1. Construction of *shape functions* matrix **N** that satisfies Eqs. (3.34) and (3.41).
2. Formulation of the *strain matrix* **B**.
3. Calculation of $\mathbf{k}_e$, $\mathbf{m}_e$, and $\mathbf{f}_e$ using **N** and **B** and Eqs. (3.71), (3.75) and (3.81).

We shall be focusing on the formulation of three types of simple but very important elements: linear triangular, bilinear rectangular, and isoparametric linear quadrilateral elements. Once the formulation of these three types of element is understood, the development of other types of elements of higher orders is straightforward, because the same techniques can be utilized. Development of higher order elements will be discussed at the end of this chapter.

7.2 LINEAR TRIANGULAR ELEMENTS

The linear triangular element was the first type of element developed for 2D solids. The formulation is also the simplest among all the 2D solid elements. It has been found that the linear triangular element is less accurate compared to linear quadrilateral elements. For this reason, it is often thought to be ideal to use quadrilateral elements, but the reality is that the triangular element is still a very useful element for its adaptivity to complex geometry. Triangular elements are normally used when we want to mesh a 2D model involving complex geometry with acute corners. In addition, the triangular configuration with the simplest topological feature makes it easier to develop meshing processors. Nowadays, analysts are hoping to use a fully automated mesh generator to perform the complex task of analysis that needs repeated or even adaptive re-meshing. Most automated mesh generators can only create triangular elements. There are automated mesh generators that can generate a quadrilateral mesh, but they still use triangular elements as patches for difficult situations, and end up with a mesh of mixed elements. Hence, whether we like it or not, we still have to use triangular elements for many practical engineering problems.

Consider a 2D model in the x–y plane, shown schematically in Figure 7.3. The 2D domain is divided in a *proper* manner into a number of *triangular elements*. The 'proper' meshing of a domain will be outlined in Chapter 11, where a list of guidelines is provided. In a mesh of linear triangular elements, each triangular element has three nodes and three straight edges.

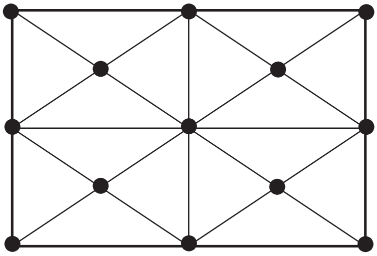


Figure 7.3. Rectangular domain meshed with triangular elements.

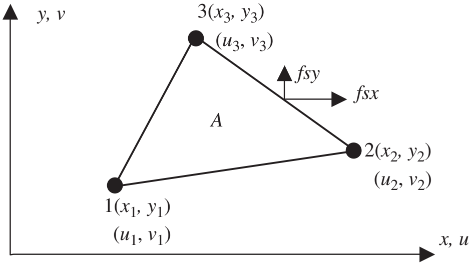


Figure 7.4. Linear triangular element.

7.2.1 Field Variable Interpolation

Consider now a triangular element of thickness h. The nodes of the element are numbered 1, 2 and 3 counter-clockwise, as shown in Figure 7.4. For 2D solid elements, the field variable is the displacement, which has two components (u and v), and hence each node has two Degrees Of Freedom (DOFs). Since a linear triangular element has three nodes, the total number of DOFs of a linear triangular element is six. For the triangular element, the local coordinate of each element can be taken as the same as the global coordinate, since there are no advantages in specifying a different local coordinate system for each element.

Now, let us examine how a triangular element can be formulated. The displacement **U** is generally a function of the coordinates x and y, and we express the displacement at any point in the element using the displacements at the nodes and shape functions. It is therefore assumed that (see Section 3.4.2)

$$\mathbf{U}^h(x, y) = \mathbf{N}(x, y)\mathbf{d}_e \tag{7.1}$$

where the superscript h indicates that the displacement is approximated, and $\mathbf{d}_e$ is a vector of the *nodal displacements* arranged in the order of

$$\mathbf{d}_e = \begin{Bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \end{Bmatrix} \begin{matrix} \left.\begin{matrix} \\ \\ \end{matrix}\right\} \text{displacements at node 1} \\ \left.\begin{matrix} \\ \\ \end{matrix}\right\} \text{displacements at node 2} \\ \left.\begin{matrix} \\ \\ \end{matrix}\right\} \text{displacements at node 3} \end{matrix} \tag{7.2}$$

and the matrix of shape functions **N** is arranged as

$$\mathbf{N} = \begin{bmatrix} \underbrace{\begin{matrix} N_1 & 0 \\ 0 & N_1 \end{matrix}}_{\text{Node 1}} & \underbrace{\begin{matrix} N_2 & 0 \\ 0 & N_2 \end{matrix}}_{\text{Node 2}} & \underbrace{\begin{matrix} N_3 & 0 \\ 0 & N_3 \end{matrix}}_{\text{Node 3}} \end{bmatrix} \tag{7.3}$$

in which $N_i (i = 1, 2, 3)$ are three shape functions corresponding to the three nodes of the triangular element. Equation (7.1) can be explicitly expressed as

$$\begin{aligned} u^h(x, y) &= N_1(x, y)u_1 + N_2(x, y)u_2 + N_3(x, y)u_3 \\ v^h(x, y) &= N_1(x, y)v_1 + N_2(x, y)v_2 + N_3(x, y)v_3 \end{aligned} \tag{7.4}$$

which implies that each of the displacement components at any point in the element is approximated by an interpolation from the nodal displacements using the shape functions. This is because the two displacement components are basically independent from each other. The question now is how can we construct these shape functions for our triangular element that satisfies the sufficient requirements: delta function property; partitions of unity; and linear field reproduction.

7.2.2 Shape Function Construction

Development of the shape functions is normally the first, and most important, step in developing finite element equations for any type of element. In determining the shape functions N_i $(i = 1, 2, 3)$ for the triangular element, we can of course follow exactly the standard procedure described in Sections 3.4.3 and 4.2.1, by starting with an assumption of the displacements using polynomial basis functions with unknown constants. These unknown constants are then determined using the nodal displacements at the nodes of the element. This standard procedure works in principle for the development of any type of element, but may not be the most convenient method. We demonstrate here another slightly different approach for constructing shape functions. We start with an assumption of shape functions directly using polynomial basis functions with unknown constants. These unknown constants are then determined using the property of the shape functions. The only difference here is that we assume directly the shape function instead of the displacements. For a linear triangular element, we assume that the shape functions are linear functions of x and y. They should, therefore, have the form of

$$N_1 = a_1 + b_1x + c_1y \tag{7.5}$$

$$N_2 = a_2 + b_2x + c_2y \tag{7.6}$$

$$N_3 = a_3 + b_3x + c_3y \tag{7.7}$$

where a_i, b_i and c_i $(i = 1, 2, 3)$ are constants to be determined. Equation (7.5) can be written in a concise form,

$$N_i = a_i + b_ix + c_iy, \quad i = 1, 2, 3 \tag{7.8}$$

We write the shape functions in the following matrix form:

$$N_i = \underbrace{\{1 \quad x \quad y\}}_{\mathbf{p}^T} \underbrace{\begin{Bmatrix} a_i \\ b_i \\ c_i \end{Bmatrix}}_{\alpha} = \mathbf{p}^T \alpha \tag{7.9}$$

where α is the vector of the three unknown constants, and $\mathbf{p}$ is the vector of polynomial basis functions (or monomials). Using Eq. (3.21), the moment matrix $\mathbf{P}$ corresponding to basis $\mathbf{p}$ can be given by

$$\mathbf{P} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix} \tag{7.10}$$

Note that the above equation is written for the shape functions, and not for the displacements. For this particular problem, we use up to the first order of polynomial basis. Depending upon the problem, we could use higher order of polynomial basis functions. The *complete order* of polynomial basis functions in two-dimensional space up to the nth order can be given by using the so-called Pascal triangle, shown in Figure 3.2. The number of terms used in $\mathbf{p}$ depends upon the number of nodes the 2D element has. We usually try to use terms of lowest orders to make the basis as complete as possible in order. It is also possible to choose specific terms of higher orders for different types of elements. For our triangular element there are three nodes, and therefore the lowest terms with complete first order are used, as shown in Eq. (7.9). The assumption of Eq. (7.5) implies that the displacement is assumed to vary linearly in the element. In Eq. (7.8) there is a total of nine constants to be determined. Our task now is to determine these constants.

If the shape functions constructed possess the delta function property, based on Lemmas 2 and 3 given in Chapter 3, the shape functions constructed will possess the partition of unity and linear field reproduction, as long as the moment matrix given in Eq. (7.10) is of full rank. Therefore, we can expect that the complete linear basis functions used in Eq. (7.9) guarantee that the shape functions to be constructed satisfy the sufficient requirements for FEM shape functions. What we need to do now is simply impose the delta function property on the assumed shape functions to determine the unknown constants a_i, b_i and c_i.

The delta functions property states that the shape function must be a unit at its home node, and zero at its remote nodes. For a two-dimensional problem, it can be expressed as

$$N_i(x_j, y_j) = \begin{cases} 1 & \text{for } i = j \\ 0 & \text{for } i \neq j \end{cases} \tag{7.11}$$

For a triangular element, this condition can be expressed explicitly for all three shape functions in the following equations. For shape function N_1, we have

$$\begin{aligned} N_1(x_1, y_1) &= 1 \\ N_1(x_2, y_2) &= 0 \\ N_1(x_3, y_3) &= 0 \end{aligned} \tag{7.12}$$

This is because node 1 at (x_1, y_1) is the home node of N_1, and nodes 2 at (x_2, y_2) and 3 at (x_3, y_3) are the remote nodes of N_1. Using Eqs. (7.5) and (7.12), we have

$$\begin{aligned} N_1(x_1, y_1) &= a_1 + b_1x_1 + c_1y_1 = 1 \\ N_1(x_2, y_2) &= a_1 + b_1x_2 + c_1y_2 = 0 \\ N_1(x_3, y_3) &= a_1 + b_1x_3 + c_1y_3 = 0 \end{aligned} \tag{7.13}$$

Solving Eq. (7.13) for a_1, b_1 and c_1, we obtain

$$a_1 = \frac{x_2y_3 - x_3y_2}{2A_e}, \quad b_1 = \frac{y_2 - y_3}{2A_e}, \quad c_1 = \frac{x_3 - x_2}{2A_e} \tag{7.14}$$

where A_e is the area of the triangular element that can be calculated using the determinant of the moment matrix:

$$A_e = \frac{1}{2}|\mathbf{P}| = \frac{1}{2}\begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} = \frac{1}{2}[(x_2y_3 - x_3y_2) + (y_2 - y_3)x_1 + (x_3 - x_2)y_1] \tag{7.15}$$

Note here that as long as the area of the triangular element is nonzero, or as long as the three nodes are not on the same line, the moment matrix **P** will be of full rank.

Substituting Eq. (7.14) into Eq. (7.5), we obtain

$$N_1 = \frac{1}{2A_e}[(x_2y_3 - x_3y_2) + (y_2 - y_3)x + (x_3 - x_2)y] \tag{7.16}$$

which can be re-written as

$$N_1 = \frac{1}{2A_e}[(y_2 - y_3)(x - x_2) + (x_3 - x_2)(y - y_2)] \tag{7.17}$$

This equation clearly shows that N_1 is a plane in the space of (x, y, N) that passes through the line of 2–3, and vanishes at nodes 2 at (x_2, y_2) and 3 at (x_3, y_3). This plane also passes the point of $(x_1, y_1, 1)$ in space that guarantees the unity of the shape function at the home node. Since the shape function varies linearly within the element, N_1 can then be easily plotted as in Figure 7.5(a). Making use of these features of N_1, we can immediately write out the other two shape functions for nodes 2 and 3. For node 2, the conditions are

$$\begin{aligned} N_2(x_1, y_1) &= 0 \\ N_2(x_2, y_2) &= 1 \\ N_2(x_3, y_3) &= 0 \end{aligned} \tag{7.18}$$

and the shape function N_2 should pass through the line 3–1, which gives

$$\begin{aligned} N_2 &= \frac{1}{2A_e}[(x_3y_1 - x_1y_3) + (y_3 - y_1)x + (x_1 - x_3)y] \\ &= \frac{1}{2A_e}[(y_3 - y_1)(x - x_3) + (x_1 - x_3)(y - y_3)] \end{aligned} \tag{7.19}$$

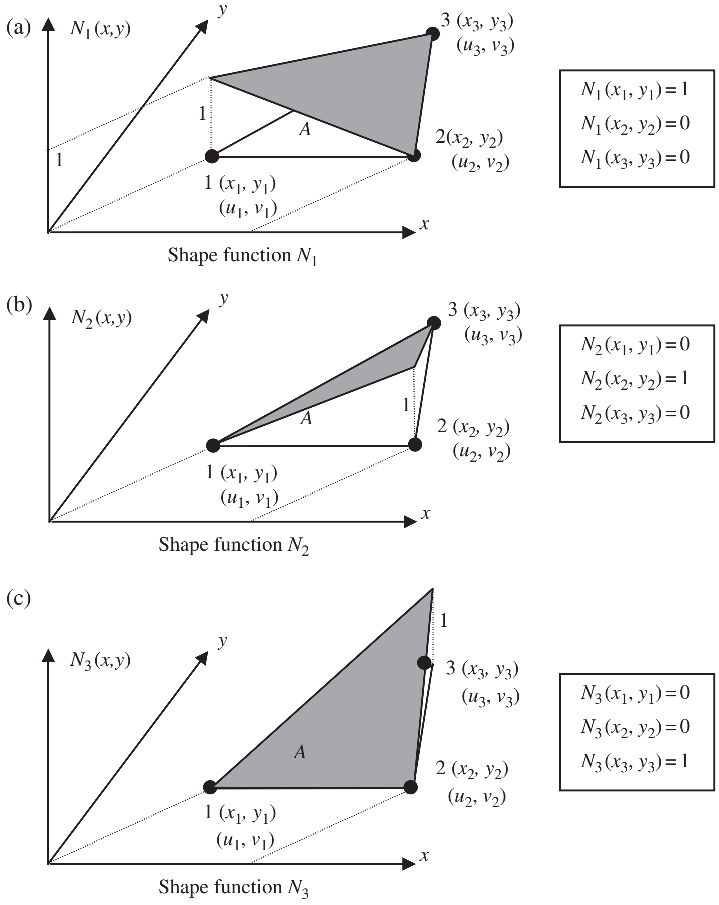


Figure 7.5. Linear triangular element and its shape functions.

which is plotted in Figure 7.5(b). For node 3, the conditions are

$$\begin{aligned} N_3(x_1, y_1) &= 0 \\ N_3(x_2, y_2) &= 0 \\ N_3(x_3, y_3) &= 1 \end{aligned} \tag{7.20}$$

and the shape function N_3 should pass through the line 1–2, and given by

$$\begin{aligned} N_3 &= \frac{1}{2A_e}[(x_1y_2 - x_1y_1) + (y_1 - y_2)x + (x_2 - x_1)y] \\ &= \frac{1}{2A_e}[(y_1 - y_2)(x - x_1) + (x_2 - x_1)(y - y_1)] \end{aligned} \tag{7.21}$$

which is plotted in Figure 7.5(c). The process of determining these constants is basically simple, algebraic manipulation. The shape functions are summarized in the following concise form:

$$N_i = a_i + b_i x + c_i y \tag{7.22}$$

$$\begin{aligned} a_i &= \frac{1}{2A_e}(x_j y_k - x_k y_j) \\ b_i &= \frac{1}{2A_e}(y_j - y_k) \qquad (7.23) \\ c_i &= \frac{1}{2A_e}(x_k - x_j) \end{aligned}$$

where the subscript i varies from 1 to 3, and j and k are determined by the cyclic permutation in the order of i, j, k. For example, if $i = 1$, then $j = 2, k = 3$. When $i = 2$, then $j = 3, k = 1$.

7.2.3 Area Coordinates

Another alternative and effective method for creating shape functions for triangular elements is to use so-called area coordinates L_1, L_2 and L_3. The use of the area coordinates will immediately lead to the shape functions for triangular elements. However, we first need to define the area coordinates.

In defining L_1, we consider a point P at (x, y) inside the triangle, as shown in Figure 7.6, and form a sub-triangle of 2–3–P. The area of this sub-triangle is noted as A_1, and can be calculated using the formula

$$A_1 = \frac{1}{2}\begin{vmatrix} 1 & x & y \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} = \frac{1}{2}[(x_2 y_3 - x_3 y_2) + (y_2 - y_3)x + (x_3 - x_2)y] \tag{7.24}$$

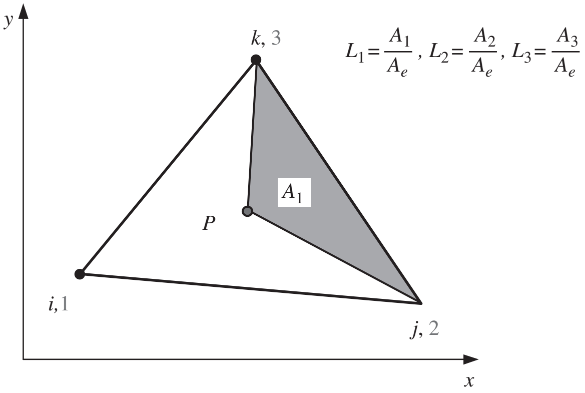

y
k, 3
$L_1 = \frac{A_1}{A_e}$, $L_2 = \frac{A_2}{A_e}$, $L_3 = \frac{A_3}{A_e}$
A_1
P
i,1
j, 2
x

Figure 7.6. Definition of area coordinates.

The area coordinate L_1 is then defined as

$$L_1 = \frac{A_1}{A_e} \tag{7.25}$$

Similarly, for L_2 we form sub-triangle 3–1–P with an area of A_2 given by

$$A_2 = \frac{1}{2}\begin{vmatrix} 1 & x & y \\ 1 & x_3 & y_3 \\ 1 & x_1 & y_1 \end{vmatrix} = \frac{1}{2}[(x_3y_1 - x_1y_3) + (y_3 - y_1)x + (x_1 - x_3)y] \tag{7.26}$$

The area coordinate L_2 is then defined as

$$L_2 = \frac{A_2}{A_e} \tag{7.27}$$

For L_3, we naturally write

$$L_3 = \frac{A_3}{A_e} \tag{7.28}$$

where A_3 is the area of the sub-triangle 1–2–P, calculated using

$$A_3 = \frac{1}{2}\begin{vmatrix} 1 & x & y \\ 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \end{vmatrix} = \frac{1}{2}[(x_1y_2 - x_2y_1) + (y_1 - y_2)x + (x_2 - x_1)y] \tag{7.29}$$

It is very easy to confirm the unity property of the area coordinates L_1, L_2 and L_3. First, they are partitions of unity, i.e.

$$L_1 + L_2 + L_3 = 1 \tag{7.30}$$

that can be proven using the definition of the area coordinates:

$$L_1 + L_2 + L_3 = \frac{A_2}{A_e} + \frac{A_2}{A_e} + \frac{A_3}{A_e} = \frac{A_2 + A_2 + A_3}{A_e} = 1 \tag{7.31}$$

Secondly, these area coordinates are of delta function properties. For example, L_1 will definitely be zero if P is at the remote nodes 2 and 3, and it will be a unit if P is at its home node 1. The same arguments are also valid for L_2 and L_3.

These two properties are exactly those defined for shape functions. Therefore, we immediately have

$$N_1 = L_1, \quad N_2 = L_2, \quad N_3 = L_3 \tag{7.32}$$

The previous equation can also be easily confirmed by comparing Eqs. (7.16) with (7.25), (7.19) with (7.27), and (7.21) with (7.28). The area coordinates are very convenient for constructing higher order shape functions for triangular elements.

Once the shape function matrix has been developed, one can write the displacement at any point in the element in terms of nodal displacements in the form of Eq. (7.1). The next step is to develop the strain matrix so that we can write the strain, and hence the stress, at any point in the element in terms of the nodal displacements. This will further lead to the element matrices.

7.2.4 Strain Matrix

Let us now move to the second step, which is to derive the strain matrix required for computing the stiffness matrix of the element. According to the discussion in Chapter 2, there are only three major stress components, $\sigma^T = \{\sigma_{xx} \quad \sigma_{yy} \quad \sigma_{xy}\}$ in a 2D solid, and the corresponding strains, $\varepsilon^T = \{\varepsilon_{xx} \quad \varepsilon_{yy} \quad \varepsilon_{xy}\}$ can be expressed as

$$\begin{aligned} \varepsilon_{xx} &= \frac{\partial u}{\partial x} \\ \varepsilon_{yy} &= \frac{\partial v}{\partial y} \\ \varepsilon_{xy} &= \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \end{aligned} \tag{7.33}$$

or in a concise matrix form,

$$\varepsilon = \mathbf{LU} \tag{7.34}$$

where **L** is called a differential operation matrix, and can be obtained simply by inspection of Eq. (7.33):

$$\mathbf{L} = \begin{bmatrix} \partial/\partial x & 0 \\ 0 & \partial/\partial y \\ \partial/\partial y & \partial/\partial x \end{bmatrix} \tag{7.35}$$

Substituting Eq. (7.1) into Eq. (7.34), we have

$$\varepsilon = \mathbf{LU} = \mathbf{LNd}_e = \mathbf{Bd}_e \tag{7.36}$$

where **B** is termed the *strain matrix*, which can be obtained by the following equation once the shape function is known:

$$\mathbf{B} = \mathbf{LN} = \begin{bmatrix} \partial/\partial x & 0 \\ 0 & \partial/\partial y \\ \partial/\partial y & \partial/\partial x \end{bmatrix} \mathbf{N} \tag{7.37}$$

Equation (7.36) implies that the strain is now expressed by the nodal displacement of the element using the strain matrix. Equations (7.36) and (7.37) are applicable for all types of 2D elements.

Using Eqs. (7.3), (7.22), (7.23) and Eq. (7.37), the strain matrix **B** for the linear triangular element can be easily obtained, to have the following simple form:

$$\mathbf{B} = \begin{bmatrix} a_1 & 0 & a_2 & 0 & a_3 & 0 \\ 0 & b_1 & 0 & b_2 & 0 & b_3 \\ b_1 & a_1 & b_2 & a_2 & b_3 & a_3 \end{bmatrix} \tag{7.38}$$

It can be clearly seen that the strain matrix **B** for a linear triangular element is a constant matrix. This implies that the strain within a linear triangular element is constant, and thus so is the stress. Therefore, the linear triangular elements are also referred to as *constant strain* *elements* or *constant stress elements*. In reality, stress or strain varies across the structure, hence using linear triangular elements with a coarse mesh will result in a rather inaccurate stress or strain distribution. We would need to have a fine mesh of linear triangular elements in order to show an appropriate variation of stress or strain across the structure.

7.2.5 Element Matrices

Having obtained the shape function and the strain matrix, the displacement and strain (hence the stress) can all be expressed in terms of the nodal displacements of the element. The element matrices, like the stiffness matrix $\mathbf{k}_e$, mass matrix $\mathbf{m}_e$, and the nodal force vector $\mathbf{f}_e$, can then be found using the equations developed in Chapter 3.

The element stiffness matrix $\mathbf{k}_e$ for 2D solid elements can be obtained using Eq. (3.71):

$$\mathbf{k}_e = \int_{V_e} \mathbf{B}^T\mathbf{c}\mathbf{B}\,\mathrm{d}V = \int_{A_e}\left(\int_0^h \mathrm{d}z\right)\mathbf{B}^T\mathbf{c}\mathbf{B}\,\mathrm{d}A = \int_{A_e} h\mathbf{B}^T\mathbf{c}\mathbf{B}\,\mathrm{d}A \tag{7.39}$$

Note that the material constant matrix $\mathbf{c}$ has been given by Eqs. (2.31) and (2.32), respectively, for the plane stress and plane strain problems. Since the strain matrix $\mathbf{B}$ is a constant matrix, as shown in Eq. (7.38), and the thickness of the element is assumed to be uniform, the integration in Eq. (7.39) can be carried out very easily, which leads to

$$\mathbf{k}_e = hA_e\mathbf{B}^T\mathbf{c}\mathbf{B} \tag{7.40}$$

The element mass matrix $\mathbf{m}_e$ can also be easily obtained by substituting the shape function matrix into Eq. (3.75):

$$\mathbf{m}_e = \int_{V_e} \rho\mathbf{N}^T\mathbf{N}\,\mathrm{d}V = \int_{A_e}\int_0^h \mathrm{d}x\rho\mathbf{N}^T\mathbf{N}\,\mathrm{d}A = \int_{A_e} h\rho\mathbf{N}^T\mathbf{N}\,\mathrm{d}A \tag{7.41}$$

For elements with uniform thickness and density, we can rewrite Eq. (7.41) as

$$\mathbf{m}_e = h\rho\int_{A_e}\begin{bmatrix} N_1N_1 & 0 & N_1N_2 & 0 & N_1N_3 & 0 \\ 0 & N_1N_1 & 0 & N_1N_2 & 0 & N_1N_3 \\ N_2N_1 & 0 & N_2N_2 & 0 & N_2N_3 & 0 \\ 0 & N_2N_1 & 0 & N_2N_2 & 0 & N_2N_3 \\ N_3N_1 & 0 & N_3N_2 & 0 & N_3N_3 & 0 \\ 0 & N_3N_1 & 0 & N_3N_2 & 0 & N_3N_3 \end{bmatrix}\mathrm{d}A \tag{7.42}$$

The integration of all the terms in the mass matrix can be carried out by simply using a mathematical formula developed by Eisenberg and Malvern (1973):

$$\int_A L_1^m L_2^n L_3^p\,\mathrm{d}A = \frac{m!n!p!}{(m+n+p+2)!}2A \tag{7.43}$$

where $L_i = N_i$ is the *area coordinates* for triangular elements that is the same as the shape function, as we have seen in Section 7.2.2. The element mass matrix $\mathbf{m}_e$ is found to be

$$\mathbf{m}_e = \frac{\rho h A}{12}\begin{bmatrix} 2 & 0 & 1 & 0 & 1 & 0 \\ & 2 & 0 & 1 & 0 & 1 \\ & & 2 & 0 & 1 & 0 \\ & & & 2 & 0 & 1 \\ & sy. & & & 2 & 0 \\ & & & & & 2 \end{bmatrix} \tag{7.44}$$

The nodal force vector for 2D solid elements can be obtained using Eqs. (3.78), (3.79) and (3.81). Suppose the element is loaded by a distributed force $\mathbf{f}_s$ on the edge 2–3 of the element, as shown in Figure 7.4; the nodal force vector becomes

$$\mathbf{f}_e = \int_l [\mathbf{N}]^{\mathrm{T}}\Big|_{2\text{–}3} \begin{Bmatrix} f_{sx} \\ f_{sy} \end{Bmatrix} \mathrm{d}l \tag{7.45}$$

If the load is uniformly distributed, f_{sx} and f_{sy} are constants within the element, so the above equation becomes

$$x\mathbf{f}_e = \frac{1}{2} l_{2\text{–}3} \begin{Bmatrix} 0 \\ 0 \\ f_x \\ f_y \\ f_x \\ f_y \end{Bmatrix} \tag{7.46}$$

where $l_{2\text{–}3}$ is the length of the edge 2–3 of the element.

Once the element stiffness matrix $\mathbf{k}_e$, mass matrix $\mathbf{m}_e$ and nodal force vector $\mathbf{f}_e$ have been obtained, the global finite element equation can be obtained by assembling the element matrices by summing up the contribution from all the adjacent elements at the shared nodes.

7.3 LINEAR RECTANGULAR ELEMENTS

Triangular elements are usually not preferred by many analysts nowadays, unless there are difficulties with the meshing and re-meshing of models of complex geometry. The main reason is that the triangular elements are usually less accurate than rectangular or quadrilateral elements. As shown in the previous section, the strain matrix of the linear triangular elements is constant, accounting for the inaccuracy. With advances in meshing algorithms, many models of complex geometry with sharp corners or curved edges can be modelled using quadrilateral elements. For the rectangular element, the strain matrix is not a constant, as will be shown in this section. This will provide a more realistic presentation in strain, and hence the stress distribution, across the structure. The formulation of the equations for the rectangular elements is simpler compared to the triangular elements, because the shape functions can form very easily due to the regularity in the shape of the rectangular element. The simple three-step procedure is applicable, and will be shown in the following sections.

7.3.1 Shape Function Construction

Consider a 2D domain. The domain is discretized into a number of *rectangular elements* with four nodes and four straight edges, as shown in Figure 7.7. As always, we number the nodes in each element 1, 2, 3 and 4 in a counter-clockwise direction. Note also that, since each node has two DOFs, the total DOFs for a linear rectangular element would be eight.

The dimension of the element is defined here as $2a \times 2b \times h$. A local *natural coordinate system* (ξ, η) with its origin located at the centre of the rectangular element is defined. The relationship between the physical coordinate (x, y) and the local natural coordinate system (ξ, η) is given by

$$\xi = x/a, \quad \eta = y/b \tag{7.47}$$

Equation (7.47) defines a very simple coordinate *mapping* between physical and natural coordinate systems for rectangular elements as shown in Figure 7.8. Our formulation can now be based on the natural coordinate system. The use of natural coordinates will make the construction of the shape functions and evaluation of the matrix integrations very much easier. This kind of coordinate mapping technique is one of the most often used techniques in the FEM. It is extremely powerful when used for developing elements of complex shapes.

We perform the field variable interpolation and express the displacement within the element as an interpolation of the nodal displacements using shape functions. The displacement vector **U** is assumed to have the form

$$\mathbf{U}^h(x, y) = \mathbf{N}(x, y)\mathbf{d}_e \tag{7.48}$$

where the nodal displacement vector $\mathbf{d}_e$ is arranged in the form

$$\mathbf{d}_e = \left\{ \begin{matrix} u_1 \\ v_1 \\ u_2 \\ u_2 \\ u_3 \\ u_3 \\ u_4 \\ u_4 \end{matrix} \right\} \begin{matrix} \left.\vphantom{\begin{matrix} u_1 \\ v_1 \end{matrix}}\right\} \text{displacements at node 1} \\ \left.\vphantom{\begin{matrix} u_1 \\ v_1 \end{matrix}}\right\} \text{displacements at node 2} \\ \left.\vphantom{\begin{matrix} u_1 \\ v_1 \end{matrix}}\right\} \text{displacements at node 3} \\ \left.\vphantom{\begin{matrix} u_1 \\ v_1 \end{matrix}}\right\} \text{displacements at node 4} \end{matrix} \tag{7.49}$$

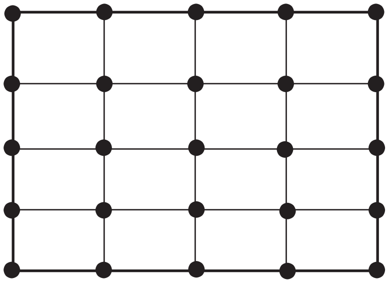


Figure 7.7. Rectangular domain meshed by rectangular elements.

and the matrix of shape functions has the form

$$\mathbf{N} = \begin{bmatrix} \underbrace{\begin{matrix} N_1 & 0 \\ 0 & N_1 \end{matrix}}_{\text{Node 1}} & \underbrace{\begin{matrix} N_2 & 0 \\ 0 & N_2 \end{matrix}}_{\text{Node 2}} & \underbrace{\begin{matrix} N_3 & 0 \\ 0 & N_3 \end{matrix}}_{\text{Node 3}} & \underbrace{\begin{matrix} N_4 & 0 \\ 0 & N_4 \end{matrix}}_{\text{Node 4}} \end{bmatrix} \tag{7.50}$$

where the shape functions N_i $(i = 1, 2, 3, 4)$ are the shape functions corresponding to the four nodes of the rectangular element.

In determining these shape functions N_i $(i = 1, 2, 3, 4)$, we can follow exactly the same steps used in Sections 4.2.1 or 7.2.2, by starting with an assumption of the displacement or shape functions using polynomial basis functions with unknown constants. These unknown constants are then determined using the displacements at the nodes of the element or the property of the shape functions. The only difference here is that we need to use four terms of monomials of basis functions. As we have seen in Section 7.2.2, the process is quite troublesome and lengthy. For many cases, one often constructs shape functions simply by some 'shortcut' methods. One of these is by inspection, and utilizing the properties of shape functions.

Due to the regularity of the square element in the natural coordinates, the shape functions in Eq. (7.50) can be written out directly as follows, without going through the detailed process that we described in the previous section for triangular elements:

$$\begin{aligned} N_1 &= \tfrac{1}{4}(1-\xi)(1-\eta) \\ N_2 &= \tfrac{1}{4}(1+\xi)(1-\eta) \\ N_3 &= \tfrac{1}{4}(1+\xi)(1+\eta) \\ N_4 &= \tfrac{1}{4}(1-\xi)(1+\eta) \end{aligned} \tag{7.51}$$

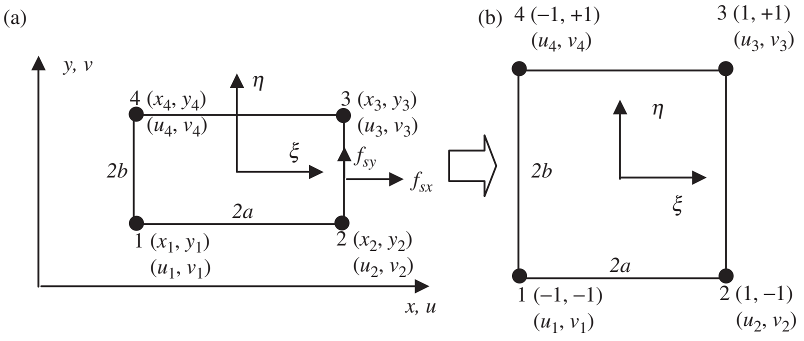


Figure 7.8. Rectangular element and the coordinate systems. (a) Rectangular element in physical system, (b) square element in natural coordinate system.

It is very easy to confirm that all the shape functions given in Eq. (7.51) satisfy the delta function property of Eq. (3.34). For example, for N_3 we have

$$\begin{aligned}
N_3|_{\text{at node 1}} &= \tfrac{1}{4}(1+\xi)(1+\eta)\Big|_{\substack{\xi=-1\\ \eta=-1}} = 0 \\
N_3|_{\text{at node 2}} &= \tfrac{1}{4}(1+\xi)(1+\eta)\Big|_{\substack{\xi=1\\ \eta=-1}} = 0 \\
N_3|_{\text{at node 3}} &= \tfrac{1}{4}(1+\xi)(1+\eta)\Big|_{\substack{\xi=1\\ \eta=1}} = 1 \\
N_3|_{\text{at node 4}} &= \tfrac{1}{4}(1+\xi)(1+\eta)\Big|_{\substack{\xi=-1\\ \eta=1}} = 0
\end{aligned} \tag{7.52}$$

The same examination of N_1, N_2 and N_4 will confirm the same property.

It is also very easy to confirm that all the shape functions given in Eq. (7.51) satisfy the partition of unity property of Eq. (3.41):

$$\begin{aligned}
\sum_{i=1}^{4} N_i &= N_1 + N_2 + N_3 + N_4 \\
&= \tfrac{1}{4}[(1-\xi)(1-\eta) + (1+\xi)(1-\eta) + (1+\xi)(1+\eta) + (1-\xi)(1+\eta)] \\
&= \tfrac{1}{4}[2(1-\xi) + 2(1+\xi)] = 1
\end{aligned} \tag{7.53}$$

The partitions of unity property can also be easily confirmed using Lemma 1 in Chapter 3.

Equation (7.53) should be called a bilinear shape function to be exact, as it varies linearly in both the ξ and η directions. It varies quadratically in any direction other than these two ξ and η directions. Denoting the natural coordinates of node j by (ξ_j, η_j), the bilinear shape function N_j can be re-written in a concise form:

$$N_j = \tfrac{1}{4}(1+\xi_j\xi)(1+\eta_j\eta) \tag{7.54}$$

7.3.2 Strain Matrix

Using the same procedure as for the case of the triangular element, the strain matrix $\mathbf{B}$ would have the same form as in Eq. (7.37), that is

$$\begin{aligned}
\mathbf{B} &= \mathbf{LN} \\
&= \begin{bmatrix}
-\dfrac{1-\eta}{a} & 0 & \dfrac{1-\eta}{a} & 0 & \dfrac{1+\eta}{a} & 0 & -\dfrac{1+\eta}{a} & 0 \\
0 & -\dfrac{1-\xi}{b} & 0 & -\dfrac{1+\xi}{b} & 0 & \dfrac{1+\xi}{b} & 0 & \dfrac{1-\xi}{b} \\
-\dfrac{1-\xi}{b} & -\dfrac{1-\eta}{a} & -\dfrac{1+\xi}{b} & \dfrac{1-\eta}{a} & \dfrac{1+\xi}{b} & \dfrac{1+\eta}{a} & \dfrac{1-\xi}{b} & -\dfrac{1+\eta}{a}
\end{bmatrix}
\end{aligned} \tag{7.55}$$

It is now clear that the strain matrix for a bilinear rectangular element is no longer a constant matrix. This implies that the strain, and hence the stress, within a linear rectangular element is not constant.

7.3.3 Element Matrices

Having obtained the shape function and the strain matrix **B**, the element stiffness matrix $\mathbf{k}_e$, mass matrix $\mathbf{m}_e$, and the nodal force vector $\mathbf{f}_e$ can be obtained using the equations presented in Chapter 3. Using first the relationship given in Eq. (7.47), we have

$$\mathrm{d}x\,\mathrm{d}y = ab\,\mathrm{d}\xi\,\mathrm{d}\eta \tag{7.56}$$

Substituting Eq. (7.56) into Eq. (7.39), we obtain

$$\mathbf{k}_e = \int_A h\mathbf{B}^{\mathrm{T}}\mathbf{c}\mathbf{B}\,\mathrm{d}A = \int_{-1}^{+1}\int_{-1}^{+1} abh\mathbf{B}^{\mathrm{T}}\mathbf{c}\mathbf{B}\,\mathrm{d}\xi\,\mathrm{d}\eta \tag{7.57}$$

The material constant matrix **c** has been given by Eqs. (2.31) and (2.32), respectively, for plane stress and plane strain problems. Evaluation of the integral in Eq. (7.57) would not be as straightforward, since the strain matrix **B** is a function of ξ and η. It is still possible to obtain the closed form for the stiffness matrix by carrying out the integrals in Eq. (7.57) analytically. In practice, we often use a numerical integration scheme to evaluate the integral, and the commonly used *Gauss integration scheme* will be introduced here. The Gauss integration scheme is a very simple and efficient procedure that performs numerical integral, and it is briefly outlined here.

7.3.4 Gauss Integration

Consider first a one-dimensional integral. Using the Gauss integration scheme, the integral is evaluated simply by a summation of the integrand evaluated at *m Gauss points* multiplied by corresponding *weight coefficients* as follows:

$$I = \int_{-1}^{+1} f(\xi)\,\mathrm{d}\xi = \sum_{j=1}^{m} w_j f(\xi_j) \tag{7.58}$$

The locations of the Gauss points and the weight coefficients have been found for different m, and are given in Table 7.1. In general, the use of more Gauss points will produce more accurate results for the integration. However, excessive use of Gauss points will increase the computational time and use up more computational resources, and it may not necessarily give better results. The appropriate number of Gauss points to be used depends upon the complexity of the integrand. It has been proven that the use of m Gauss points gives the exact results of a polynomial integrand of up to an order of $n = 2m - 1$. For example, if the integrand is a linear function (straight line), we have $2m - 1 = 1$, which gives $m = 1$. This means that for a linear integrand, one Gauss point will be sufficient to

Table 7.1. Gauss integration points and weight coefficients

m	ξ_j	w_j	Accuracy n
1	0	2	1
2	$-1/\sqrt{3}, 1/\sqrt{3}$	1,1	3
3	$-\sqrt{0.6}, 0, \sqrt{0.6}$	5/9, 8/9, 5/9	5
4	−0.861136, −0.339981, 0.339981, 0.861136	0.347855, 0.652145, 0.652145, 0.347855	7
5	−0.906180, −0.538469, 0, 0.538469, 0.906180	0.236927, 0.478629, 0.568889, 0.478629, 0.236927	9
6	−0.932470, −0.661209, −0.238619, 0.238619, 0.661209, 0.932470	0.171324, 0.360762, 0.467914, 0.467914, 0.360762, 0.171324	11

give the exact result of the integration. If the integrand is of a polynomial of a third order, we have $2m - 1 = 3$, which gives $m = 2$. This means that for an integrand of a third order polynomial, the use of two Gauss points will be sufficient to give the exact result. The use of more than two points will still give the same results, but takes more computation time. For two-dimensional integrations, the Gauss integration is sampled in two directions, as follows:

$$I = \int_{-1}^{+1} \int_{-1}^{+1} f(\xi, \eta)\, \mathrm{d}\xi\, \mathrm{d}\eta = \sum_{i=1}^{n_x} \sum_{j=1}^{n_y} w_i w_j f(\xi_i, \eta_j) \tag{7.59}$$

Figure 7.9(b) shows the locations of four Gauss points used for integration in a square region.

The element stiffness matrix $\mathbf{k}_e$ can be obtained by numerically carrying out the integrals in Eq. (7.57) using the Gauss integration scheme of Eq. (7.59). 2×2 Gauss points shown in Figure 7.9(b) are sufficient to obtain the exact solution for the stiffness matrix given by Eq. (7.57). This is because the entry in the strain matrix, $\mathbf{B}$, is a linear function of ξ or η. The integrand in Eq. (7.57) consists of $\mathbf{B}^T\mathbf{cB}$, which implies multiplications of two linear functions, and hence this becomes a quadratic function. In Table 7.1, having two Gauss points sampled in each direction is sufficient to obtain the exact results for a polynomial function in that direction of order up to 3. Figure 7.9(a) and (c) show some other different and possible number of integration points in a square region.

To obtain the element mass matrix $\mathbf{m}_e$, we substitute Eq. (7.56) into Eq. (3.75) to obtain

$$\begin{aligned}\mathbf{m}_e &= \int_V \rho \mathbf{N}^T\mathbf{N}\, \mathrm{d}V = \int_A \int_0^h \mathrm{d}x \rho \mathbf{N}^T\mathbf{N}\, \mathrm{d}A = \int_A h\rho \mathbf{N}^T\mathbf{N}\, \mathrm{d}A \\ &= \int_{-1}^{+1} \int_{-1}^{+1} abh\rho \mathbf{N}^T\mathbf{N}\, \mathrm{d}\xi\, \mathrm{d}\eta \end{aligned} \tag{7.60}$$

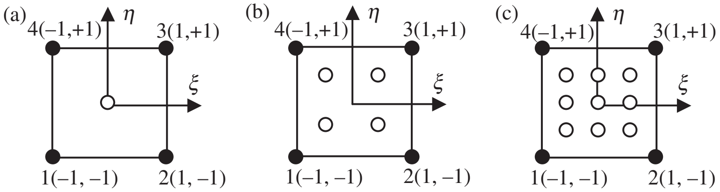


Figure 7.9. Integration points for $n_x = n_y = 1, 2$ and 3 in a square region.

Upon evaluation of the integral, after substitution of Eq. (7.50) into Eq. (7.60), the element mass matrix $\mathbf{m}_e$ is obtained explicitly as

$$\mathbf{m}_e = \frac{\rho hab}{9}\begin{bmatrix} 4 & 0 & 2 & 0 & 1 & 0 & 2 & 0 \\ & 4 & 0 & 2 & 0 & 1 & 0 & 2 \\ & & 4 & 0 & 2 & 0 & 1 & 0 \\ & & & 4 & 0 & 2 & 0 & 1 \\ & & & & 4 & 0 & 2 & 0 \\ & & & & & 4 & 0 & 2 \\ & sy. & & & & & 4 & 0 \\ & & & & & & & 4 \end{bmatrix} \tag{7.61}$$

In obtaining element m_{ij} in the mass matrix, the following integral has been carried out and repeatedly used:

$$\begin{aligned} m_{ij} &= \rho hab \int_{-1}^{+1} \int_{-1}^{+1} N_i N_j \,\mathrm{d}\xi \,\mathrm{d}\eta \\ &= \frac{\rho hab}{16} \int_{-1}^{+1} (1 + \xi_i \xi)(1 + \xi_j \xi) \,\mathrm{d}\xi \int_{-1}^{+1} (1 + \eta_i \eta)(1 + \eta_j \eta) \,\mathrm{d}\eta \\ &= \frac{\rho hab}{4} \left(1 + \frac{1}{3}\xi_i \xi_j\right)\left(1 + \frac{1}{3}\eta_i \eta_j\right) \end{aligned} \tag{7.62}$$

For example, in calculating m_{33}, we use the above equation to obtain

$$m_{33} = \frac{\rho hab}{4}\left(1 + \frac{1}{3} \times 1 \times 1\right)\left(1 + \frac{1}{3} \times 1 \times 1\right) = 4 \times \frac{\rho hab}{9} \tag{7.63}$$

In practice, the integrals in Eq. (7.60) are often calculated numerically using the Gauss integration scheme.

The nodal force vector for a rectangular element can be obtained by using Eqs. (3.78), (3.79) and (3.81). Suppose the element is loaded by a distributed force $\mathbf{f}_s$ on edge 2–3 of the element, as shown in Figure 7.8; the nodal force vector becomes

$$\mathbf{f}_e = \int_l [\mathbf{N}]^{\mathrm{T}}\Big|_{2\text{-}3} \begin{Bmatrix} f_{sx} \\ f_{sy} \end{Bmatrix} \mathrm{d}l \tag{7.64}$$

If the load is uniformly distributed within the element, and f_{sx} and f_{sy} are constant, the above equation becomes

$$\mathbf{f}_e = b \begin{Bmatrix} 0 \\ 0 \\ f_x \\ f_y \\ f_x \\ f_y \\ 0 \\ 0 \end{Bmatrix} \tag{7.65}$$

where b is the half length of the side 2–3. Equation (7.65) suggests that the evenly distributed load is divided equally onto nodes 2 and 3.

The stiffness matrix $\mathbf{k}_e$, mass matrix $\mathbf{m}_e$ and nodal force vector $\mathbf{f}_e$ can be used directly to assemble the global FE equation, Eq. (3.96). Coordinate transformation is needed if the orientation of the local natural coordinate does not coincide with that of the global coordinate system. In such a case, *quadrilateral elements* are often used, which is to be developed in the next section.

7.4 LINEAR QUADRILATERAL ELEMENTS

Though the rectangular element can be very useful, and is usually more accurate than the triangular element, it is difficult to use it for problems with any geometry rather than rectangles. Hence, its practical application is very limited. A much more practical and useful element would be the so-called quadrilateral element, that can have unparalleled edges. However, there can be a problem for the integration of the mass and stiffness matrices for a quadrilateral element, because of the irregular shape of the integration domain. The Gauss integration scheme cannot be implemented directly with quadrilateral elements. Therefore, what is required first is to *map* the quadrilateral element into the natural coordinates system to become a square element, so that the shape functions and the integration method used for the rectangular element can be utilized. Hence, key in the development of a quadrilateral element is the coordinate mapping. Once the mapping is established, the rest of the procedure is exactly the same as that used for formulating the rectangular element in the previous section.

7.4.1 Coordinate Mapping

Figure 7.10 shows a 2D domain with the shape of an airplane wing. As you can imagine, dividing such a domain into rectangular elements of parallel edges is impossible. The job can be easily accomplished by the use of *quadrilateral elements* with four straight but unparallel edges, as shown in Figure 7.10. In developing the quadrilateral elements, we use the same coordinate mapping that was used for the rectangular elements in the previous section. Due to the slightly increased complexity of the element shape, the mapping will become a little more involved, but the procedure is basically the same.

Consider now a quadrilateral element with four nodes numbered 1, 2, 3 and 4 in a counter-clockwise direction, as shown in Figure 7.11. The coordinates for the four nodes are indicated in Figure 7.11(a) in the *physical coordinate system*. The physical coordinate system can be the same as the global coordinate system for the entire structure. As there are two DOFs at a node, a linear quadrilateral element has a total of eight DOFs, like the rectangular element. A local *natural coordinate system* (ξ, η) with its origin at the centre of the squared element mapped from the global coordinate system is used to construct the shape functions, and the displacement is interpolated using the equation

$$\mathbf{U}^h(\xi, \eta) = \mathbf{N}(\xi, \eta)\mathbf{d}_e \tag{7.66}$$

Equation (7.66) represents a field variable interpolation using the nodal displacements. Using a similar concept, we can also interpolate the coordinates x and y themselves. In other words, we let coordinates x and y be interpolated from the nodal coordinates using the shape functions which are expressed as functions of the natural coordinates. This coordinate

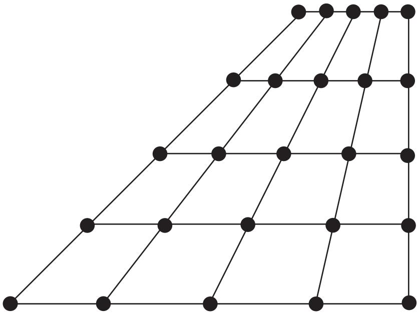


Figure 7.10. 2D domain meshed by quadrilateral elements.

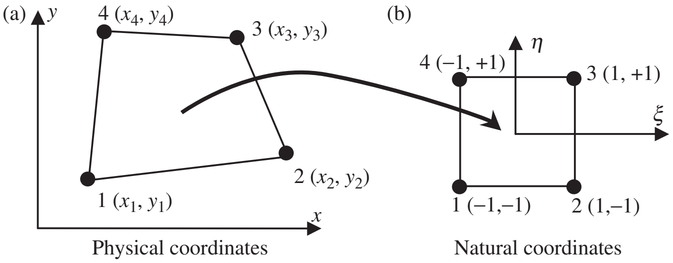


Figure 7.11. Coordinates mapping between coordinate systems.

interpolation is mathematically expressed as

$$\mathbf{X}(\xi,\eta)=\mathbf{N}(\xi,\eta)\mathbf{x}_e \tag{7.67}$$

where **X** is the vector of the physical coordinates,

$$\mathbf{X}=\begin{Bmatrix} x \\ y \end{Bmatrix} \tag{7.68}$$

and N is the matrix of shape functions given by Eq. (7.50). In Eq. (7.67), $\mathbf{x}_e$ is the physical coordinates at the nodes of the element, given by

$$\mathbf{x}_e=\begin{Bmatrix} x_1 \\ y_1 \\ x_2 \\ y_2 \\ x_3 \\ y_3 \\ x_4 \\ y_4 \end{Bmatrix} \begin{matrix} \left.\begin{matrix} \\ \\ \end{matrix}\right\} \text{coordinate at node 1} \\ \left.\begin{matrix} \\ \\ \end{matrix}\right\} \text{coordinate at node 2} \\ \left.\begin{matrix} \\ \\ \end{matrix}\right\} \text{coordinate at node 3} \\ \left.\begin{matrix} \\ \\ \end{matrix}\right\} \text{coordinate at node 4} \end{matrix} \tag{7.69}$$

Equation (7.67) can also be expressed explicitly as

$$\begin{aligned} x &= \sum_{i=1}^{4} N_i(\xi,\eta)x_i \\ y &= \sum_{i=1}^{4} N_i(\xi,\eta)y_i \end{aligned} \tag{7.70}$$

where N_i is the shape function defined for the rectangular element in Eqs. (7.53) or (7.54). Note that, due to the unique property of the shape function, the interpolation at these nodes will be exact. For example, substituting $\xi = 1$ and $\eta = -1$ in Eq. (7.70) gives $x = x_2$ and $y = y_2$, as shown in Figure 7.11. Physically, this means that point 2 in the natural coordinate system is mapped to point 2 in the physical coordinate system, and vice versa. The same can be easily observed also for points 1, 3 and 4.

Let us now analyse this mapping more closely. Substituting $\xi = 1$ into Eq. (7.70) gives

$$\begin{aligned} x &= \tfrac{1}{2}(1-\eta)x_2+\tfrac{1}{2}(1+\eta)x_3 \\ y &= \tfrac{1}{2}(1-\eta)y_2+\tfrac{1}{2}(1+\eta)y_3 \end{aligned} \tag{7.71}$$

or

$$\begin{aligned} x &= \tfrac{1}{2}(x_2+x_3)+\tfrac{1}{2}\eta(x_3-x_2) \\ y &= \tfrac{1}{2}(y_2+y_3)+\tfrac{1}{2}\eta(y_3-y_2) \end{aligned} \tag{7.72}$$

Eliminating η from the above two equations gives

$$y = \frac{(x_3 - x_2)}{(y_3 - y_2)} \left\{ x - \frac{1}{2}(x_2 + x_3) \right\} + \frac{1}{2}(y_2 + y_3) \tag{7.73}$$

which represents a straight line connecting the points (x_2, y_2) and (x_3, y_3). This means that edge 2–3 in the physical coordinate system is mapped onto edge 2–3 in the natural coordinate system. The same can be observed for the other three edges. Hence, we can see that the four straight edges of the quadrilateral in the physical coordinate system correspond to the four straight edges of the square in the natural coordinate system. Therefore, the full domain of the quadrilateral element is mapped onto a square one.

7.4.2 Strain Matrix

After mapping is performed for the coordinates, we can now evaluate the strain matrix **B**. To do so in this case, it is necessary to express the differentials in terms of the natural coordinates, since the relationship between the x and y coordinates and the natural coordinates is no longer as straightforward as in the case for rectangular elements. Utilizing the chain rule in application to partial differentiation, we have

$$\begin{aligned} \frac{\partial N_i}{\partial \xi} &= \frac{\partial N_i}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial N_i}{\partial y}\frac{\partial y}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} &= \frac{\partial N_i}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial N_i}{\partial y}\frac{\partial y}{\partial \eta} \end{aligned} \tag{7.74}$$

The above equations can be written in the matrix form

$$\begin{bmatrix} \partial N_i/\partial \xi \\ \partial N_i/\partial \eta \end{bmatrix} = \mathbf{J} \begin{bmatrix} \partial N_i/\partial x \\ \partial N_i/\partial y \end{bmatrix} \tag{7.75}$$

where **J** is the *Jacobian matrix* defined by

$$\mathbf{J} = \begin{bmatrix} \partial x/\partial \xi & \partial y/\partial \xi \\ \partial x/\partial \eta & \partial y/\partial \eta \end{bmatrix} \tag{7.76}$$

We now substitute the interpolation of the coordinates defined by Eq. (7.70) into the above equation, and obtain

$$\mathbf{J} = \begin{bmatrix} \partial N_1/\partial \xi & \partial N_2/\partial \xi & \partial N_3/\partial \xi & \partial N_4/\partial \xi \\ \partial N_1/\partial \eta & \partial N_2/\partial \eta & \partial N_3/\partial \eta & \partial N_4/\partial \eta \end{bmatrix} \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \end{bmatrix} \tag{7.77}$$

Rewriting Eq. (7.75) to obtain

$$\begin{bmatrix} \partial N_i/\partial x \\ \partial N_i/\partial y \end{bmatrix} = \mathbf{J}^{-1} \begin{bmatrix} \partial N_i/\partial \xi \\ \partial N_i/\partial \eta \end{bmatrix} \tag{7.78}$$

which gives the relationship between the differentials of the shape functions with respect to x and y with those with respect to ξ and η. We can now use the equation $\mathbf{B} = \mathbf{LN}$ to compute the strain matrix **B**, by replacing all the differentials of the shape functions with respect to x and y with those with respect to ξ and η, using Eq. (7.78). This process needs to be performed numerically by a computer.

7.4.3 Element Matrices

Once the strain matrix **B** has been obtained, we can proceed to evaluate the element matrices. The stiffness matrix can be obtained by Eq. (7.39). To evaluate the integration, the following formula, which has been proven by Murnaghan (1951), is used:

$$\mathrm{d}A = \det|\mathbf{J}|\,\mathrm{d}\xi\,\mathrm{d}\eta \tag{7.79}$$

where det $|\mathbf{J}|$ is the determinate of the Jacobian matrix. Hence, the element stiffness matrix can be written as

$$\mathbf{k}_e = \int_{-1}^{+1}\int_{-1}^{+1} h\mathbf{B}^{\mathrm{T}}\mathbf{c}\mathbf{B}\det|\mathbf{J}|\;\mathrm{d}\xi\,\mathrm{d}\eta \tag{7.80}$$

The above integrals can then be evaluated using the Gauss integration scheme discussed in the previous section. Notice how the coordinate mapping enables us to use the Gauss integration scheme over a simple squared area.

The shape function defined by Eq. (7.53) is a bilinear function of ξ and η. The elements in the strain matrix **B** are obtained by differentiating these bilinear functions with respect to ξ and η, and by dividing by the Jacobian matrix whose elements are also bilinear functions. Therefore, the elements of $\mathbf{B}^{\mathrm{T}}\mathbf{c}\mathbf{B}\det|\mathbf{J}|$ are fraction functions, which cannot usually be expressed by polynomials. This means that the stiffness matrix may not be able to be evaluated exactly using the Gauss integration scheme, unlike the case for the rectangular element.

The element mass matrix $\mathbf{m}_e$ can also be evaluated in the same way as for the rectangular element using Eq. (7.60):

$$\begin{aligned}\mathbf{m}_e &= \int_V \rho\mathbf{N}^T\mathbf{N}\mathrm{d}V = \int_A\int_0^h \mathrm{d}x\rho\mathbf{N}^T\mathbf{N}\,\mathrm{d}A = \int_A h\rho\mathbf{N}^T\mathbf{N}\,\mathrm{d}A \\ &= \int_{-1}^{+1}\int_{-1}^{+1} h\rho\mathbf{N}^T\mathbf{N}\det|\mathbf{J}|\;\mathrm{d}\xi\,\mathrm{d}\eta\end{aligned} \tag{7.81}$$

The element force vector is obtained in the same way as described for the rectangular element, because the integration for calculating the force vectors is one-dimensional line integrations. Having obtained the element matrices, the usual method of assembling the element matrices is carried out to obtain the global matrices.

7.4.4 Remarks

The shape functions used to interpolate the coordinates in Eq. (7.70) are the same as those used for interpolation of the displacements. Such an element is called an *isoparametric* *element*. However, the shape functions for coordinate and displacement interpolations do not necessarily have to be the same. Using different shape functions for coordinate and displacement interpolations will lead to the development of what is known as *subparametric* or *superparametric* elements. These elements have been studied in academic research, but less often used in practical applications. Details of such elements will not be covered in this book.

7.5 HIGHER ORDER ELEMENTS

7.5.1 Triangular Element Family

General formulation of shape functions

In developing higher order elements, we make use of the area coordinate system. Figure 7.12 shows a general triangular element of order p that has n_d nodes calculated by

$$n_d = (p+1)(p+2)/2 \tag{7.82}$$

Node i (I, J, K) is located at the Ith node in the L_1 direction, at the Jth node in the L_2 direction, and at the Kth node in the L_3 direction. From Figure 7.12, we have at any node that

$$I + J + K = p \tag{7.83}$$

The shape function can be written in the form (Argyris, 1968)

$$N_i = l_I^I(L_1) l_J^J(L_2) l_K^K(L_3) \tag{7.84}$$

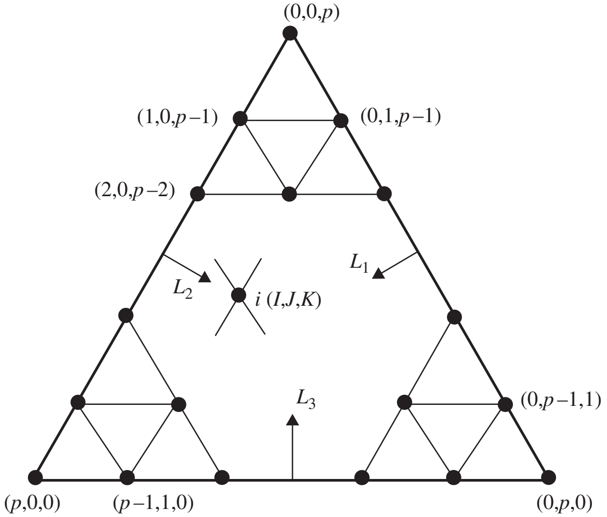


Figure 7.12. Triangular element of order p defined under the area coordinate system. Node i (I, J, K) is located at the Ith node in the L_1 direction, at the Jth node in the L_2 direction, and at the Kth node in the L_3 direction. At any node, we have $I + J + K = p$.

where l_I^I, l_J^J and l_K^K are defined by Eq. (4.82), but the coordinate ξ is replaced by the area coordinates, i.e.

$$l_\beta^\beta(L_\alpha) = \frac{(L_\alpha - L_{\alpha 0})(L_\alpha - L_{\alpha 1})\cdots(L_\alpha - L_{\alpha(\beta-1)})}{(L_{\alpha I} - L_{\alpha 0})(L_{\alpha I} - L_{\alpha 1})\cdots(L_{\alpha I} - L_{\alpha(\beta-1)})} \tag{7.85}$$

where $\alpha = 1, 2, 3$; $\beta = I, J, K$. For example, when $\alpha = 1$ and $\beta = I$, we have

$$l_I^I(L_1) = \frac{(L_1 - L_{10})(L_1 - L_{11})\cdots(L_1 - L_{1(I-1)})}{(L_{1I} - L_{10})(L_{1I} - L_{11})\cdots(L_{1I} - L_{1(I-1)})} \tag{7.86}$$

Since

$$l_\beta^\beta(L_\alpha) = \begin{cases} 1 & \text{when } L_\alpha = L_{\alpha I} \\ 0 & \text{otherwise} \end{cases} \tag{7.87}$$

it is easy to confirm the delta function property of

$$N_i = \begin{cases} 1 & \text{at nodes } L_1 = L_{1I},\ L_1 = L_{1J} \text{ and } L_1 = L_{1K} \\ 0 & \text{other nodes} \end{cases} \tag{7.88}$$

From Eqs. (7.84) and (7.85), the order of the shape function can be found to be the same as

$$(L_1)^I(L_2)^J(L_3)^K \tag{7.89}$$

Since L_α is linear function of x and y, the order of the shape function will be

$$I + J + K = p \tag{7.90}$$

Quadratic triangular elements

Consider a quadratic triangular element shown in Figure 7.13. The element has six nodes: three corner nodes and three mid-side nodes. Using Eqs (7.84) and (7.85), the shape functions can be obtained very easily. Here we demonstrate the calculation of N_1. Note that for

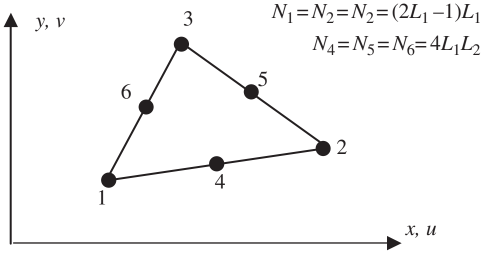


Figure 7.13. Quadratic triangular element.

the element shown in Figure 7.13, the area coordinate L_1 has three coordinate values:

$$\begin{aligned} L_{10} &= 0 && \text{at nodes 2, 3 and 5} \\ L_{11} &= 0.5 && \text{at nodes 4 and 6} \\ L_{12} &= 1.0 && \text{at node 1} \end{aligned} \tag{7.91}$$

Using Eq. (7.84), we have

$$\begin{aligned} N_1 &= l_2^2(L_1)l_0^0(L_2)l_0^0(L_2) = l_2^2(L_1) \times 1 \times 1 = l_2^2(L_1) \\ &= \frac{(L_1 - L_{10})(L_1 - L_{11})}{(L_{12} - L_{10})(L_{12} - L_{11})} = \frac{(L_1 - 0)(L_1 - 0.5)}{(1 - 0)(1 - 0.5)} \\ &= (2L_1 - 1)L_1 \end{aligned} \tag{7.92}$$

For the other two corner nodes 2, 3, we should have exactly the same equation:

$$N_1 = N_2 = N_2 = (2L_1 - 1)L_1 \tag{7.93}$$

For the mid-side node 4, we have

$$\begin{aligned} N_4 &= l_1^1(L_1)l_1^1(L_2)l_0^0(L_2) = l_1^1(L_1) \times l_1^1(L_2) \times 1 = l_1^1(L_1)l_1^1(L_2) \\ &= \frac{(L_1 - L_{10})}{(L_{11} - L_{10})}\frac{(L_2 - L_{20})}{(L_{21} - L_{20})} = \frac{(L_1 - 0)}{(0.5 - 0)}\frac{(L_2 - 0)}{(0.5 - 0)} \\ &= 4L_1L_2 \end{aligned} \tag{7.94}$$

This equation is also valid for the other two mid-nodes, and therefore we have

$$N_4 = N_5 = N_6 = 4L_1L_2 \tag{7.95}$$

Cubic triangular elements

For the cubic triangular element shown in Figure 7.14 that has nine nodes, the shape function can also be obtained using Eq. (7.84), as well as four area coordinate values of (taking L_1 as an example)

$$\begin{aligned} L_{10} &= 0 && \text{at nodes 2, 6, 7 and 3} \\ L_{11} &= \tfrac{1}{3} && \text{at nodes 5, 10 and 8} \\ L_{12} &= \tfrac{2}{3} && \text{at nodes 4 and 9} \\ L_{13} &= 1 && \text{at node 1} \end{aligned} \tag{7.96}$$

We omit the process and list the results below. The reader is encouraged to confirm the results. For corner nodes (1, 2, and 3):

$$N_1 = N_2 = N_3 = \tfrac{1}{2}(3L_1 - 1)(3L_1 - 2)L_1 \tag{7.97}$$

For side nodes (4–9):

$$N_4 \sim N_9 = \tfrac{9}{2}L_1L_2(3L_1 - 1) \tag{7.98}$$

For the interior node (10):

$$N_{10} = 27L_1L_2L_3 \tag{7.99}$$

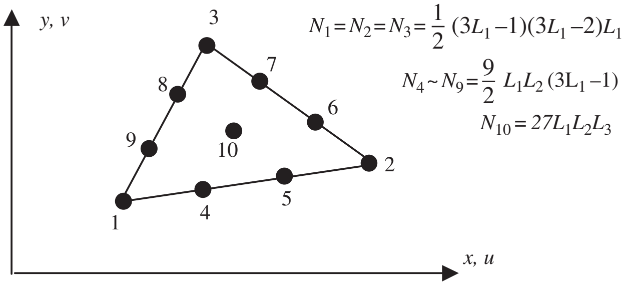

y, v
x, u
1
2
3
4
5
6
7
8
9
10
$N_1=N_2=N_3=\frac{1}{2}(3L_1-1)(3L_1-2)L_1$
$N_4\sim N_9=\frac{9}{2}L_1L_2(3L_1-1)$
$N_{10}=27L_1L_2L_3$

Figure 7.14. Cubic triangular element.

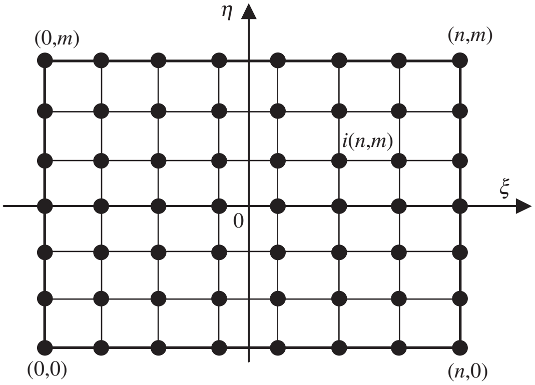


Figure 7.15. Rectangular element of arbitrary high orders.

7.5.2 Rectangular Elements

Lagrange type elements

Considering a rectangular element with $n_d = (n+1)(m+1)$ nodes, shown in Figure 7.15. The element is defined in the domain of $(-1 \leq \xi \geq 1, -1 \leq \eta \geq 1)$ in the natural coordinates ξ and η. Due to the regularity of the nodal distribution along both the ξ and η directions, the shape function of the element can be simply obtained by multiplying one-dimensional shape functions with respect to the ξ and η directions using the Lagrange interpolants defined in Eq. (4.82) (Zienkiewicz *et al.*, 2000):

$$N_i = N_I^{1D} N_J^{1D} = l_I^n(\xi) l_J^m(\eta) \tag{7.100}$$

Due to the delta function proper of the 1D shape functions given in Eq. (4.83), it is easy to confirm that the N_i given by Eq. (7.100) is also of the delta function property.

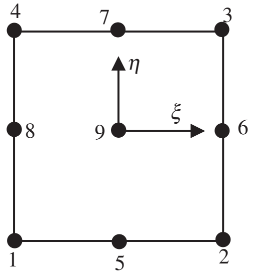


Figure 7.16. Nine-node rectangular element.

Using Eqs. (7.100) and (4.82), the nine-node quadratic element shown in Figure 7.16 can be given by

$$\begin{aligned}
N_1 &= N_1^{1D}(\xi)N_1^{1D}(\eta) = \tfrac{1}{4}\xi(1-\xi)\eta(1-\eta)\\
N_2 &= N_2^{1D}(\xi)N_1^{1D}(\eta) = -\tfrac{1}{4}\xi(1+\xi)\eta(1-\eta)\\
N_3 &= N_2^{1D}(\xi)N_2^{1D}(\eta) = \tfrac{1}{4}\xi(1+\xi)(1+\eta)\eta\\
N_4 &= N_1^{1D}(\xi)N_2^{1D}(\eta) = -\tfrac{1}{4}\xi(1-\xi)(1+\eta)\eta\\
N_5 &= N_3^{1D}(\xi)N_1^{1D}(\eta) = -\tfrac{1}{2}(1+\xi)(1-\xi)(1-\eta)\eta\\
N_6 &= N_2^{1D}(\xi)N_3^{1D}(\eta) = \tfrac{1}{2}\xi(1+\xi)(1+\eta)(1-\eta)\\
N_7 &= N_3^{1D}(\xi)N_2^{1D}(\eta) = \tfrac{1}{2}(1+\xi)(1-\xi)(1+\eta)\eta\\
N_8 &= N_1^{1D}(\xi)N_1^{1D}(\eta) = -\tfrac{1}{2}\xi(1-\xi)(1-\eta)\eta\\
N_9 &= N_3^{1D}(\xi)N_3^{1D}(\eta) = (1-\xi^2)(1-\eta^2)
\end{aligned} \tag{7.101}$$

From Eq. (7.101), it can easily be seen that all the shape functions are formed using the same set of nine basis functions:

$$1, \xi, \eta, \xi\eta, \xi^2, \eta^2, \xi^2\eta, \eta^2\xi, \xi^2\eta^2 \tag{7.102}$$

which are linearly-independent. From Lemma 2 in Chapter 3, we can expect that the shape functions given in Eq. (7.101) are partitions of unity. In addition, because the basis functions also contain the linear basis functions, these shape functions can also be expected to have linear field reproduction (Lemma 3), at least. Hence, they satisfy the sufficient requirements for FEM shape functions. Any other high order Lagrange type of rectangular element can be created in exactly the same way as for the nine-node element.

Serendipity type elements

The method used in constructing the Lagrange type of elements is very systematic. However, the Lagrange type of elements is not very widely used, due to the presence of the

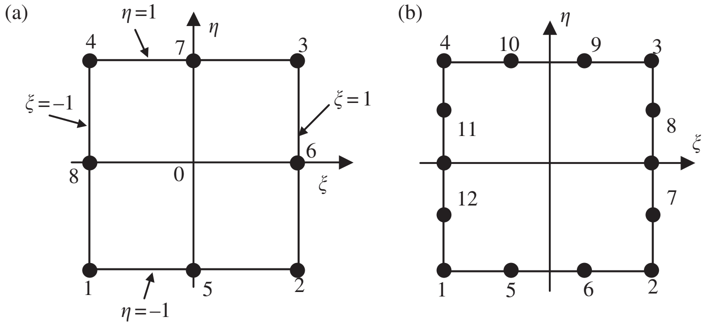


Figure 7.17. High order serendipity element. (a) Eight-node element. (b) 12-node element.

interior nodes. Serendipity type elements are created by inspective construction methods. We intentionally construct high order elements without interior nodes.

Consider the eight-node element shown in Figure 7.17a. The element has four corner nodes and four mid-side nodes. The shape functions in the natural coordinates for the quadratic rectangular element are given as

$$\begin{aligned} N_j &= \tfrac{1}{4}(1+\xi_j\xi)(1+\eta_j\eta)(\xi_j\xi+\eta_j\eta-1) \quad \text{for corner nodes } j=1,2,3,4 \\ N_j &= \tfrac{1}{2}(1-\xi^2)(1+\eta_j\eta) \quad \text{for mid-side nodes } j=5,7 \\ N_j &= \tfrac{1}{2}(1+\xi_j\xi)(1-\eta^2) \quad \text{for mid-side nodes } j=6,8 \end{aligned} \tag{7.103}$$

where (ξ_j, η_j) are the natural coordinates of node j. It is very easy to observe that the shape functions possess the delta function property. The shape function is constructed by simple inspections making use of the shape function properties. For example, for the corner node 1 (where $\xi_1 = -1, \eta_1 = -1$), the shape function N_1 has to pass the following three lines as shown in Figure 7.18 to ensure its vanishing at remote nodes:

$$\begin{aligned} 1-\xi &= 0 \Rightarrow \text{vanishes at nodes } 2, 6, 3 \\ 1-\eta &= 0 \Rightarrow \text{vanishes at nodes } 3, 4, 7 \\ -\xi-\eta-1 &= 0 \Rightarrow \text{vanishes at nodes } 5, 8 \end{aligned} \tag{7.104}$$

The shape N_1 can then immediately be written as

$$N_1 = C(1-\xi)(1-\eta)(-\xi-\eta-1) \tag{7.105}$$

where C is a constant to be determined using the condition that it has to be unity at node 1 at $(\xi_1 = -1, \eta_1 = -1)$, which gives

$$C = \frac{1}{(1-(-1))(1-(-1))(-(-1)-(-1)-1)} = \frac{1}{4} \tag{7.106}$$

We finally have

$$N_1 = \tfrac{1}{4}(1+\xi_1\xi)(1+\eta_1\eta)(\xi_1\xi+\eta_1\eta-1) \tag{7.107}$$

which is the first equation in Eq. (7.103) for $j = 1$.

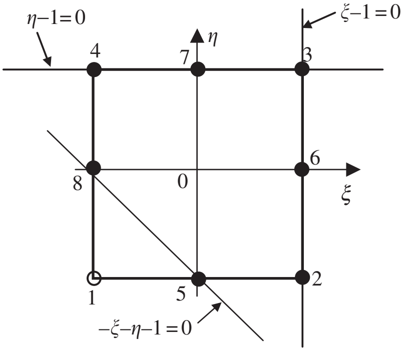


Figure 7.18. Construction of eight-node serendipity element. Three straight lines passing through the remote nodes of node 1 are used.

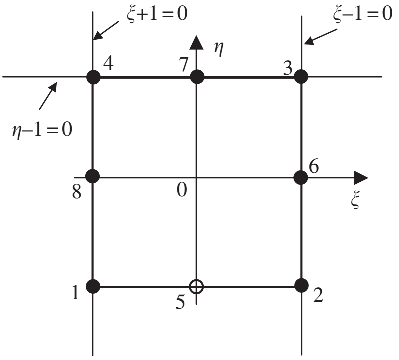


Figure 7.19. Construction of eight-node serendipity element. Three straight lines passing through the remote nodes of node 5 are used.

Shape functions at all the other corner nodes can be constructed in exactly the same manner. As for the mid-side nodes, say node 5, we enforce the shape function passing through the following three lines as shown in Figure 7.19:

$$\begin{aligned} 1-\xi &= 0 \Rightarrow \text{vanishes at nodes 2, 6, 3} \\ 1+\xi &= 0 \Rightarrow \text{vanishes at nodes 1, 8, 4} \\ 1-\eta &= 0 \Rightarrow \text{vanishes at nodes 3, 4, 7} \end{aligned} \tag{7.108}$$

The shape N_5 can then immediately be written as

$$N_5 = C(1+\xi)(1-\xi)(1-\eta) \tag{7.109}$$

where C is a constant to be determined using the condition that it has to be unity at node 5 at ($\xi_5 = 0$, $\eta_5 = -1$), which gives

$$C = \frac{1}{(1+\xi)(1-\xi)(1-\eta)} = \frac{1}{(1+0)(1-0)(1-(-1))} = \frac{1}{2} \tag{7.110}$$

We finally have

$$N_5 = \tfrac{1}{2}(1-\xi^2)(1+\eta_5\eta) \tag{7.111}$$

which is the second equation in Eq. (7.103) for $j = 5$.

Because the delta functions property is used for the shape functions given in Eq. (7.103), they of course possess the delta function property. It can be easily seen that all the shape functions can be formed using the same set of basis functions

$$1, \xi, \eta, \xi\eta, \xi^2, \eta^2, \xi^2\eta, \eta^2\xi \tag{7.112}$$

that are linearly-independent. From Lemmas 2 and 3, we confirm that the shape functions are partitions of unity, and at least linear field reproduction. Hence, they satisfy the sufficient requirements for FEM shape functions.

Following a similar procedure, the shape functions for the 12-node cubic element shown in Figure 7.17b can be written as

$$\left.\begin{aligned}
&N_j = \tfrac{1}{32}(1+\xi_j\xi)(1+\eta_j\eta)(9\xi^2+9\eta^2-10)\\
&\quad \text{for corner nodes } j = 1, 2, 3, 4\\
&N_j = \tfrac{9}{32}(1+\xi_j\xi)(1-\eta^2)(1+9\eta_j\eta)\\
&\quad \text{for side nodes } j = 7, 8, 11, 12 \text{ where } \xi_j = \pm 1 \text{ and } \eta_j = \pm\tfrac{1}{3}\\
&N_j = \tfrac{9}{32}(1+\eta_j\eta)(1-\xi^2)(1+9\xi_j\xi)\\
&\quad \text{for side nodes } j = 5, 6, 9, 10 \text{ where } \xi_j = \pm\tfrac{1}{3} \text{ and } \eta_j = \pm 1
\end{aligned}\right. \tag{7.113}$$

The reader is encouraged to figure out what lines should be used to form the shape functions listed in Eq. (7.113). When $\eta = \eta_i = 1$, the above equations reduce to one-dimensional cases of quadratic and cubic elements defined by Eqs. (4.84) and (4.85).

7.6 ELEMENTS WITH CURVED EDGES

Using high order elements, elements with curved edges can be used in the modelling. Two relatively frequently used higher order elements of curved edges are shown in Figure 7.20(a). In formulating these types of elements, the same mapping technique used for the linear quadrilateral elements (Section 7.4) can be used. In the physical coordinate system, elements with curved edges, as shown in Figure 7.20(a), are first formed in the problem domain. These elements are then mapped into the natural coordinate system using Eq. (7.67). The elements mapped in the natural coordinate system will have straight edges, as shown in Figure 7.20(b).

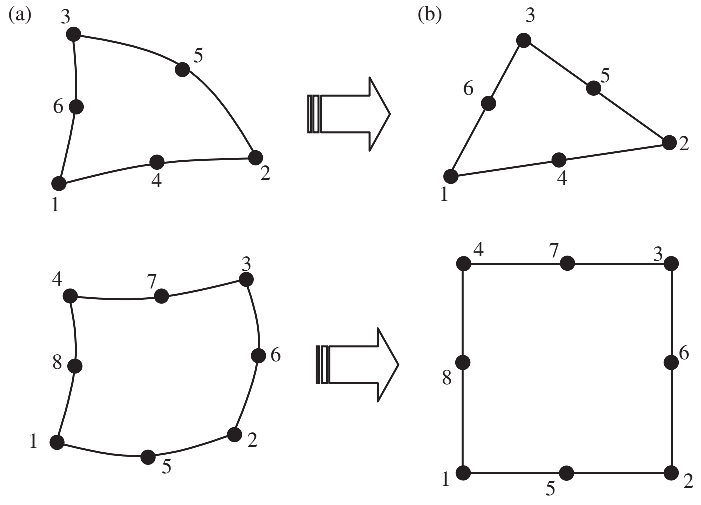


Figure 7.20. 2D solid elements with curved edges. (a) Curved elements in the physical coordinate system. (b) elements with straight edges obtained by mapping.

Higher order elements of curved edges are often used for modelling curved boundaries. Note that elements with excessively curved edges may cause problems in the numerical integration. Therefore, more elements should be used where the curvature of the boundary is large. In addition, it is recommended that in the internal portion of the domain, elements with straight edges should be used whenever possible. More details on modelling issues will be discussed intensively in Chapter 11.

7.7 COMMENTS ON GAUSS INTEGRATION

When the Gauss integration scheme is used, one has to decide how many Gauss points should be used. Theoretically, for a one-dimensional integral, using m points can give the exact solution for the integral of a polynomial integrand of up to an order of $(2m - 1)$. As a general rule of thumb, more points should be used for a higher order of elements. It is also noted that using a smaller number of Gauss points tends to counteract the *over-stiff behaviour* associated with the displacement-based finite element method.

This over-stiff behaviour of the displacement-based finite element method comes about primarily because of the use of the shape function. As discussed, the displacement in an element is assumed using shape functions interpolated from the nodal displacements. This implies that the deformation of the element is actually prescribed in the fashion of the shape function. This gives a constraint to the element, and thus the element behaves more stiffly than it should. It is often observed that higher order elements are usually softer than lower order ones. This is because the use of more nodes decreases the constraint on the element.

Coming back to the Gauss integration issue, two Gauss points for linear elements, and about two or three Gauss points in each direction for quadratic elements, should be sufficient for most cases. Many of the explicit FEM codes based on explicit formulation tend to use one-point integration to achieve the best performance in saving CPU time.

7.8 CASE STUDY: SIDE DRIVE MICRO-MOTOR

In this case study, we analyse another MEMs device: a common micro-actuator in the form of a side drive electrostatic micro-motor, as shown in Figure 7.21. Such micro-motors are usually made from polysilicon using lithographic techniques. Their diameters vary depending on the design, with the first designs having diameters of 60–120 μm. Of course, the actual working dynamics of the micro-motor will be rather complex to model, though it can still be readily done if required. Therefore, to illustrate certain points pertaining to the use of basic 2D solid elements, we basically use the geometrical and material information for this micro-motor and apply arbitrary loading and boundary conditions to it.

Isotropic material properties will be employed here to makes things less complicated. The material properties of polysilicon are shown in Table 7.2. We shall do a stress analysis on the rotor with some loading condition on the rotor blades. Examining the rotor in Figure 7.21, we can see that it is symmetrical, i.e. we need not model the full rotor, but rather we can

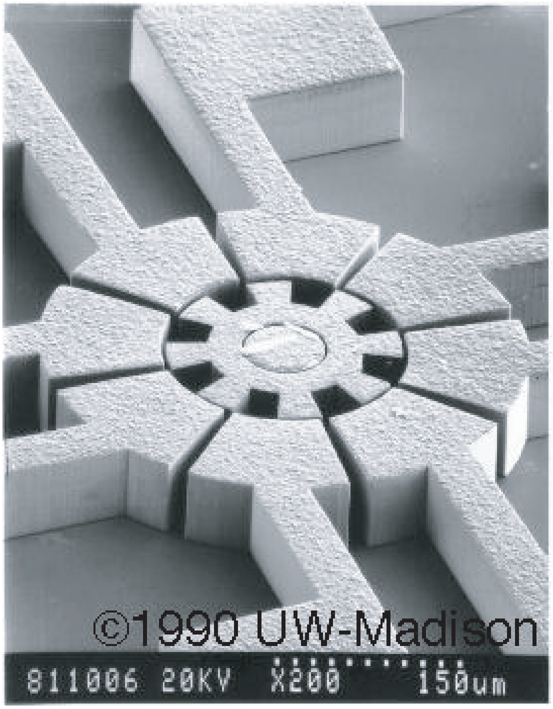


Figure 7.21. SEM image of an electrostatic micro-motor with eight rotor and 12 stator poles. (Courtesy of Professor Henry Guckel and the University of Wisconsin-Madison.)

Table 7.2. Elastic properties of polysilicon

Young's Modulus, E	169 GPa
Poisson's ratio, ν	0.262
Density, ρ	2300 kgm^{-3}

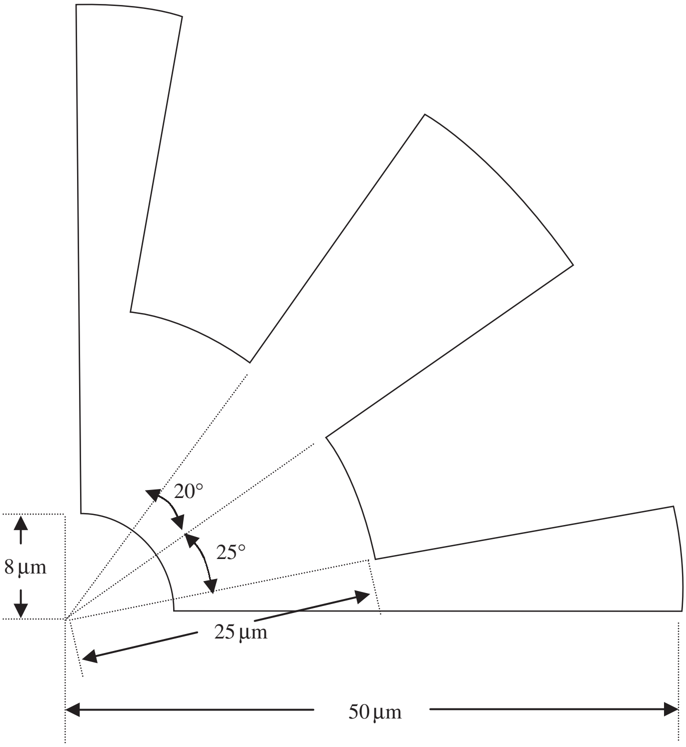


Figure 7.22. Plan view (2D) of a quarter of micro-motor rotor.

just model say one quarter of the rotor and apply the necessary boundary conditions. We can do this since this one-quarter model will be repeated geometrically anyway. Of course, we can even model one eighth of the model and the results will be the same if the condition of repetition is properly applied. Hence, this becomes a neat and efficient way of modelling repetitive or symmetrical geometry.

7.8.1 Modelling

Figure 7.22 shows the one-quarter model of the micro-motor rotor. We take the diameter of the whole rotor to be 100 µm and the depth or thickness to be 13 µm, to correspond with realistic values of micro-motor designs. The geometry can be easily drawn using

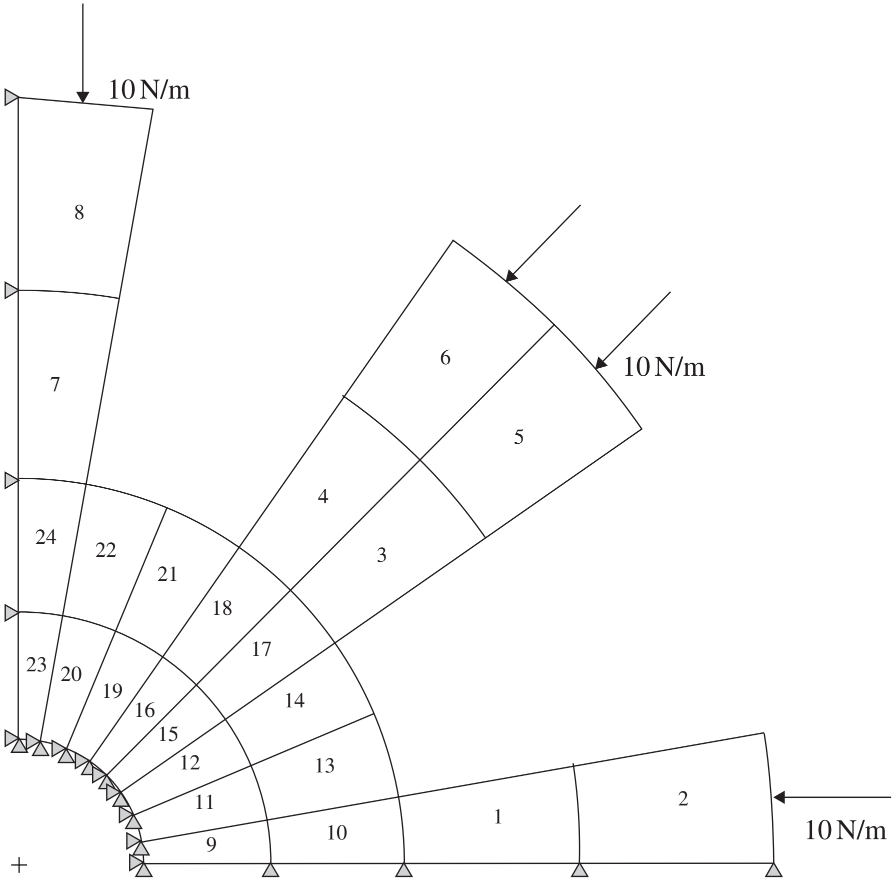


Figure 7.23. Finite element mesh with 24 2D quadrilateral, four nodal elements.

preprocessors like PATRAN or using basic CAD software, after which it can be imported into preprocessors for meshing. Note that preprocessors are software used to aid us in visualizing the geometry, and to mesh up the geometry using finite elements, especially for complicated geometries. To illustrate the formulation of the finite element equations clearly, we would initially mesh up the geometry in Figure 7.22 with a very sparse mesh, as shown in Figure 7.23. Four nodal, quadrilateral elements are used with a total of 24 elements and 41 nodes in the model. We shall increase the number of elements (and nodes) in later analyses to compare the results. Since the depth or thickness of the motor is much smaller than the other dimensions, and the external forces are assumed to be within the plane of the rotor, we can assume plane stress conditions.

In the above figure, it can also be seen that a distributed force of 10 N/m is applied compressively to the rotor blades. The centre hole in the rotor which is supposed to be the location for a 'hub' to keep the rotor in place is assumed to be constrained. The nodes along the edge $y = 0$ are constrained in the y direction and the nodes along the edge $x = 0$ are constrained in the x direction. These are to simulate the symmetrical boundary conditions of the model, since those nodes are not supposed to move in the direction normal to the plane of symmetry. Similarly, if we are to model a one-eighth model, such rules for symmetry apply except that one of the planes of symmetry will be the line $y = x$.

7.8.2 ABAQUS Input File

The ABAQUS input file for the above described finite element model is shown below. Note that some parts of the input file containing the data values are left out to limit the length of the file in this book. The text boxes to the right of the input file are not part of the file, but rather explain what the sections of the file mean.

```
*HEADING, SPARSE
Static analysis of micro-motor
**
*NODE
1, 8., 0.
2, 7.87846, 1.38919
3, 16.5, 0.
4, 16.2493, 2.8652
5, 25., 0.
⋮
38, 6.5118, 36.9303
39, -2.73294E-7, 37.5
40, 8.68241, 49.2404
41, 5.46197E-7, 50.
**
**
```

Nodal cards

Defines the coordinates of the nodes in the model. The first entry being the node ID while the second and third entries are the x and y coordinates of the position of the node, respectively.

```
*ELEMENT, TYPE=CPS4, ELSET=PLSTRESS
1, 1, 3, 4, 2
2, 3, 5, 6, 4
3, 2, 4, 11, 8
4, 8, 11, 12, 9

⋮

20, 23, 50, 51, 24
21, 49, 52, 53, 50
22, 50, 53, 54, 51
23, 33, 57, 58, 39
24, 57, 59, 60, 58
**
** plstress
**
```

Element (connectivity) cards

Defines the element type and what nodes make up the element. CPS4 represents that it is a plane stress, four-nodal, quadrilateral element. There are many other element types in the ABAQUS element library. For example, if we were to use a plane strain element, the element type would be CPE4. The “ELSET = PLSTRESS” statement is simply for naming this set of elements so that it can be referenced when defining the material properties. In the subsequent data entry, the first entry is the element ID, and the following four entries are the nodes making up the element. The order of the nodes for all elements must be consistent and counter-clockwise.

```
*SOLID SECTION, ELSET=PLSTRESS, MATERIAL=POLYSILI
13.,
**
** polysilicon
**
*MATERIAL, NAME=POLYSILI
**
*DENSITY
2.3E-15,
**
*ELASTIC, TYPE=ISO
169000., 0.262
**
```

Property cards

Defines properties to the elements of set "PLSTRESS". It will have the material properties defined under "POLYSILI".

Material cards

Defines material properties under the name "POLYSILI". Density and elastic properties are defined. TYPE = ISO represents isotropic properties.

```
** Fixed
**
*BOUNDARY, OP=NEW
FIXED, 1,, 0.
FIXED, 2,, 0.
**
** Fixed-x
**
*BOUNDARY, OP=NEW
FIXED-X, 1,, 0.
**
** Fixed-y
**
*BOUNDARY, OP=NEW
FIXED-Y, 2,, 0.
**
**
```

BC cards

Defines boundary conditions. For example, the first one labelled "FIXED" represents that nodes belonging to the set "FIXED" have its "1" and "2" directions constrained.

```
*STEP, AMPLITUDE=RAMP, PERTURBATION
Linear Static Analysis
**
**
*STATIC
**
```

Control card

Indicates the analysis step. In this case it is a "STATIC" analysis.

```
*NSET, NSET=FIXED
1, 2, 8, 9, 17, 18, 26, 27,
35,
*NSET, NSET=FIXED-X
37, 39, 58, 60
*NSET, NSET=FIXED-Y
3, 5, 42, 44
*ELSET, ELSET=PRESS
18, 21, 22, 24
**
** press
**
```

Node sets

Sets of nodes defined to be used for referencing when defining boundary and loading conditions. For example, the nodes 37, 39, 58 and 60 are grouped up as a set labelled "FIXED-X".

Load cards

"DLOAD" defines distributed loading on the element set "PRESS" defined earlier.

```
*DLOAD, OP=NEW
PRESS, P2, 10.
**
**
*NODE PRINT, FREQ=1
U,
*NODE FILE, FREQ=1
U,
**
*EL PRINT, POS=INTEG, FREQ=1
S,
E,
*EL FILE, POS=INTEG, FREQ=1
S,
E,
**
*END STEP
```

> *Output control cards*
>
> Defines the output required. For example for nodal output, we require the displacement "U", while for element output, we require the stress, "S" and strain "E".

The input file above shows how a basic ABAQUS input file can be set up. It should be noted that the units used in this case study are micrometres, and all the conversions of the necessary inputs is done for consistency, as before.

7.8.3 Solution Process

Let us now try to relate the information we provided in the input file with what is covered in this chapter. As before, the first sets of data usually defined are the nodes and their coordinates. Then, there are the element cards containing the connectivity information. The importance of this information has already been mentioned in previous case studies. Looking at Figure 7.23, it is not difficult to guess that the element used is an isoparametric quadrilateral element (CPS4–2D, quadrilateral, bilinear, plane stress elements), rather than that of a rectangular element. Obviously, it can be visualized that using rectangular elements would pose a problem in meshing the geometry here. In fact, the use of purely rectangular elements is so rare that most software (including ABAQUS) only provides the more versatile quadrilateral element. This information from the nodal and element cards will be used for constructing the element matrices (Eqs. (7.80) and (7.81)).

Next, the property cards define the properties of the elements, and also specify the material the elements should possess. For the plane stress elements, the thickness of the elements must be specified (13 μm in this case), since it is required in the stiffness and mass matrices (the mass matrix is actually not required in this case study, since this is a static analysis). Similarly, the elastic properties of the polysilicon material defined in the material card are also required in the element matrices. It should be noted that in ABAQUS, the integral in Eq. (7.80) is evaluated using the Gauss integration scheme, and the default number of Gauss points for the bilinear element is 4.

The boundary cards (BC cards) define the boundary conditions for the model. To model the symmetrical boundary conditions, at the lines of symmetry ($x = 0$ and $y = 0$), the nodal displacement component normal to the line is constrained to zero. The nodes (node set, FIXED) along the centre hole where the hub should be is also fully clamped in. The load cards defined specify the distributed loading on the motor, as shown in Figure 7.23. These will be used to form the force vector, which is similar in form to that of Eq. (7.64).

The control cards are used to control the analysis, which in this case defines that this is a static analysis. Finally, the output cards define the necessary output requested, which here are the displacement components, the stress components and the strain components.

Once the input file has been created, one can then invoke ABAQUS to execute the analysis, and the results will be written into an output file that can be read by the post-processor.

7.8.4 Results and Discussion

Using the above ABAQUS input file that describes the problem, a static analysis is carried out. Figure 7.24 shows the Von Mises stress distribution obtained with 24 bilinear quadrilateral elements. It should be noted here that 24 elements (41 nodes) for such a problem may not be sufficient for accurate results. Analyses with a denser mesh (129 nodes and 185 nodes) using the same element type are also carried out. Their input files will be similar to that shown, but with more nodes and elements.

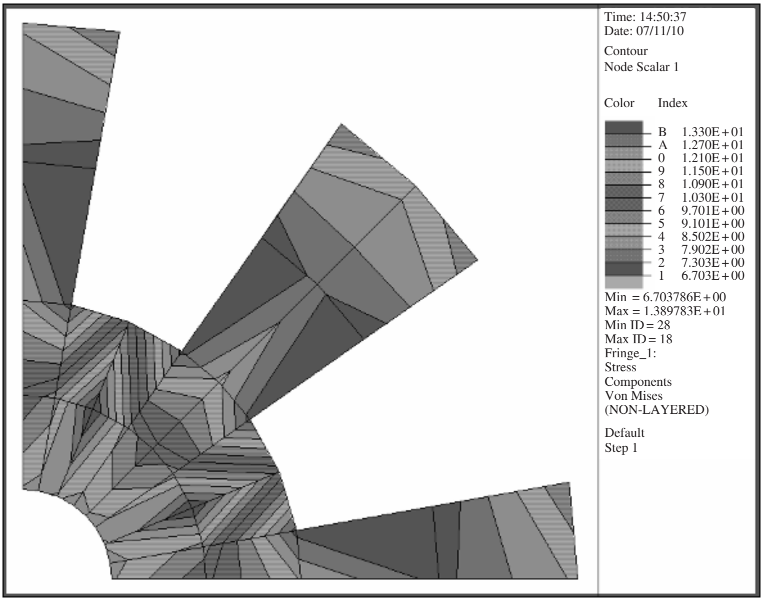


Figure 7.24. Analysis no. 1: Von Mises stress distribution using 24 bilinear quadrilateral elements.

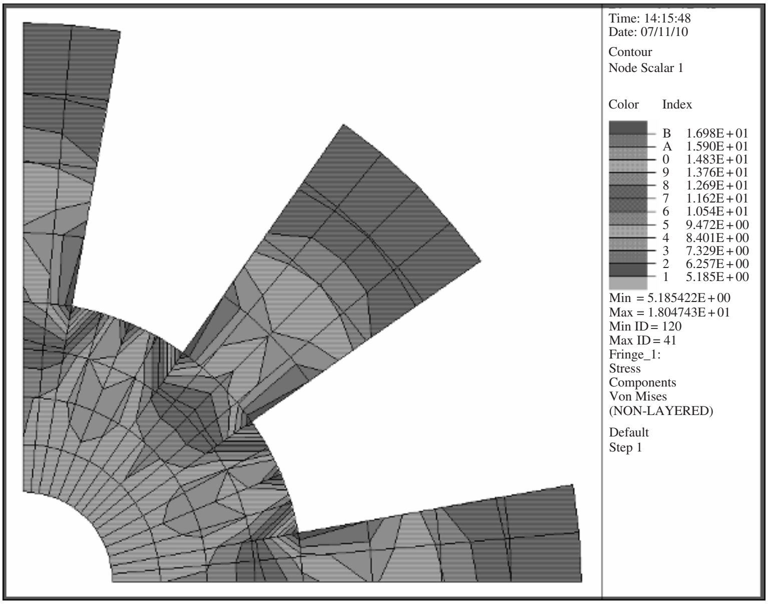


Figure 7.25. Analysis no. 2: Von Mises stress distribution using 96 bilinear quadrilateral elements.

Figures 7.25 and 7.26 show the Von Mises stress distribution obtained using 96 (129 nodes) and 144 elements (185 nodes), respectively. Figure 7.27 also shows the results obtained when 24 eight-nodal elements (105 nodes in total) are used instead of four-nodal elements. The element type in ABAQUS for eight nodal, plane stress, quadratic element is 'CPS8'. Finally, linear, triangular elements (CPS3) are also used for comparison, and the stress distribution obtained is shown in Figure 7.28.

From the results obtained, it can be noted that analysis 1, which uses 24 bilinear elements, does not seem as accurate as the other three. Table 7.3 shows the maximum Von Mises stress for the five analyses. It can be seen that the maximum Von Mises stress using just 24 bilinear, quadrilateral elements (41 nodes) is just about 0.0139 GPa, which is a bit low when compared with the other analyses. The other analyses, especially from analyses 2 to 4 using quadrilateral elements, obtained results that are quite close to one another when we compare the maximum Von Mises stress. We can conclude that using just 24 bilinear, quadrilateral elements is definitely not sufficient in this case. The comparison also shows that using quadratic elements (eight-nodal) with a total of 105 nodes, yielded results that are close to analysis 3 with the bilinear elements and 105 nodes. In this case, the quadratic elements also have curved edges, instead of straight edges and this would define the curved geometry better. Looking at the maximum Von Mises stress obtained using triangular elements in analysis 5, we can see that, despite having the same number of nodes as in analysis 2, the results obtained showed some deviation. This clearly shows that quadrilateral elements in general provide better accuracy than triangular elements. However, it is still convenient to use triangular elements to mesh complex geometry containing sharp corners.

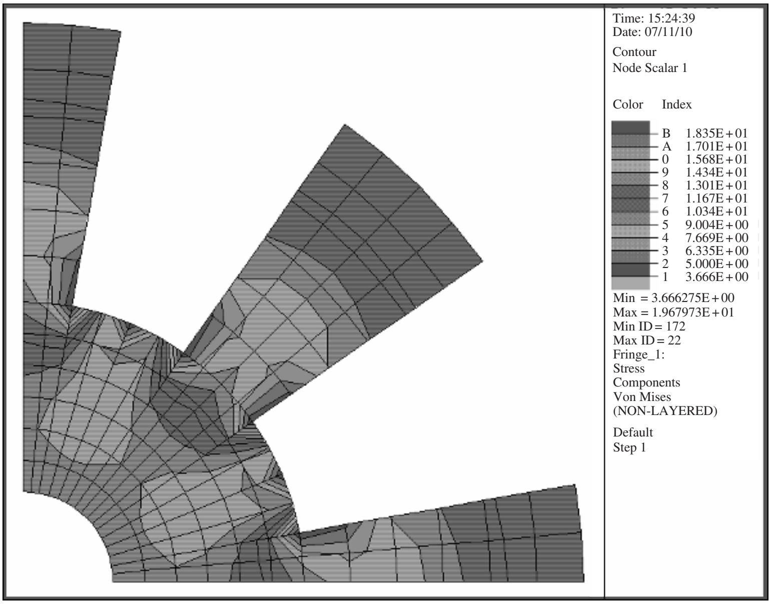


Figure 7.26. Analysis no. 3: Von Mises stress distribution using 144 bilinear quadrilateral elements.

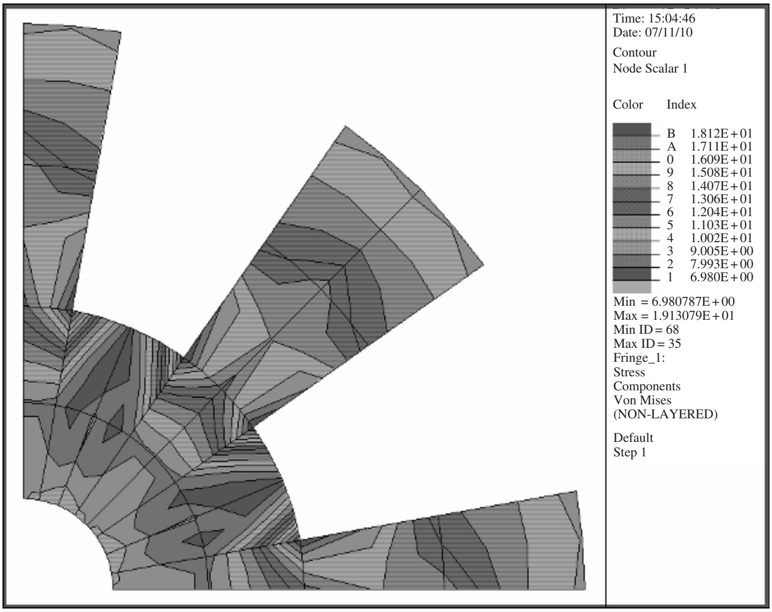


Figure 7.27. Analysis no. 4: Von Mises stress distribution using 24 eight-nodal, quadratic elements.

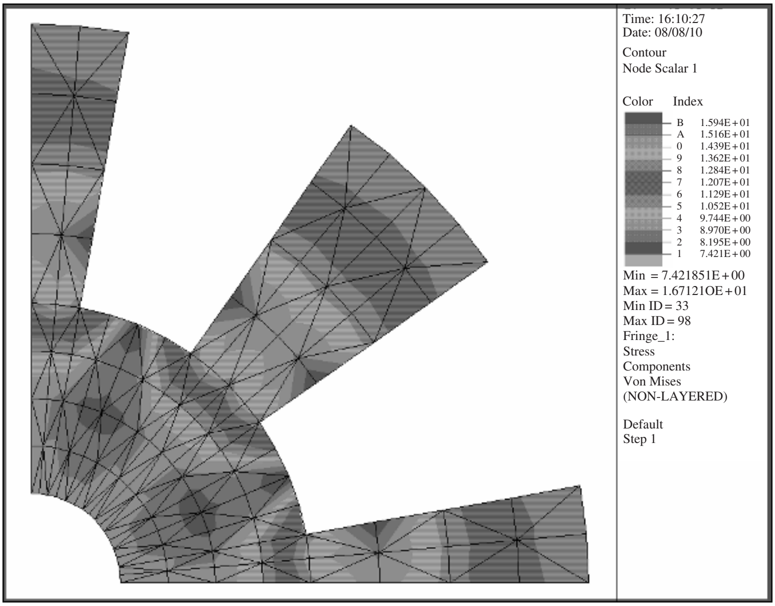


Figure 7.28. Analysis no. 5: Von Mises stress distribution using 192 three-nodal, triangular elements.

Table 7.3. Maximum Von Mises stress

Analysis no.	Number/type of elements	Total number of nodes in model	Maximum Von Mises stress (GPa)
1	24 bilinear, quadrilateral	41	0.0139
2	96 bilinear, quadrilateral	129	0.0180
3	144 bilinear, quadrilateral	185	0.0197
4	24 quadratic, quadrilateral	105	0.0191
5	192 linear, triangular	129	0.0167

From the stress distribution, it can generally be seen that there is stress concentration at the corners of the rotor structure, as expected. Therefore, if structural failure is to occur, it would be at these areas of stress concentration.

7.9 REVIEW QUESTIONS (PETYT, 1990)

1. Show that the stiffness matrix of an isotropic linear triangular element whose thickness varies linearly in the element is

$$[\mathbf{k}]_e = \overline{h} A_e [\mathbf{B}]^T [\mathbf{D}][\mathbf{B}]$$

where [**B**] is the strain matrix, [**D**] is matrix of material constants, A_e is the area of the triangle and $\bar{h}$ the average thickness $(h_1 + h_2 + h_3)/3$, where h_1, h_2 and h_3 are the nodal thickness at the node.

2. Show that the mass matrix of a linear triangular element whose thickness varies linearly within the plane of the element is

$$[\mathbf{m}]_e = \frac{\rho\bar{h}A_e}{60}\begin{bmatrix} 6+4\alpha_1 & 0 & 6-\alpha_3 & 0 & 6-\alpha_2 & 0 \\ & 6+4\alpha_1 & 0 & 6-\alpha_3 & 0 & 6-\alpha_2 \\ & & 6+4\alpha_2 & 0 & 6-\alpha_1 & 0 \\ & & & 6+4\alpha_2 & 0 & 6-\alpha_1 \\ & sy. & & & 6+4\alpha_3 & 0 \\ & & & & & 6+4\alpha_3 \end{bmatrix}$$

where ρ is the density, A_e is the area, $\bar{h}$ is the mean thickness and $\alpha_i = h_i/\bar{h}$ with $i = 1, 2, 3$ for the three nodes.

3. The thickness variation of a linear rectangular element is given by

$$h(\xi, \eta) = \sum_{i=1}^{4} N_i(\xi, \eta)h_i$$

where N_i is the bilinear shape function, and h_i is the thickness value at node i. How many Gauss points are required to evaluate exactly the mass and stiffness matrices?

4. If the thickness variation of a linear quadrilateral is the same as the rectangular element in Problem (3), how many Gauss points are required to evaluate the mass matrix exactly? How many Gauss points are required to integrate the volume of the element exactly? How many Gauss points are required to integrate the stiffness matrix exactly?

5. Construct the shape functions for the 12-node rectangular element for one corner node and one side node.

8

FEM FOR PLATES AND SHELLS

8.1 INTRODUCTION

In this chapter, finite element equations for plates and shells are developed. The procedure is to first develop FE matrices for plate elements, and the FE matrices for shell elements are then obtained by superimposing the matrices for plate elements and those for 2D solid plane stress elements developed in Chapter 7. Unlike the 2D solid elements in the previous chapter, plate and shell elements are computationally more tedious as they involve more Degrees Of Freedom (DOFs). The constitutive equations may seem daunting to one who may not have a strong background in the mechanics theory of plates and shells, or the integration may be quite involved if it is to be carried out analytically. However, the basic concept of formulating the finite element equation always remains the same. Readers are advised to pay more attention to the finite element concepts and the procedures outlined in developing plate and shell elements. After all, the computer can handle many of the tedious calculations/integrations that are required in the process of forming the elements. The basic concepts, procedures and formulations can also be found in many existing textbooks (see, e.g. Petyt,1990; Rao, 1999; Zienkiewicz and Taylor, 2000; etc.).

8.2 PLATE ELEMENTS

As discussed in Chapter 2, a plate structure is geometrically similar to the structure of the 2D plane stress problem, but it usually carries only transversal loads that lead to bending deformation in the plate. For example, consider the horizontal boards on a bookshelf that support the books. Those boards can be approximated as a plate structure, and the transversal loads are of course the weight of the books. Higher floors of a building are a typical plate structure that carries most of us every day, as are the wings of aircraft, which usually carry loads like the engines, as shown in Figure 2.13. The plate structure can be schematically represented by its middle plane laying on the x–y plane, as shown in Figure 8.1. The deformation caused by the transverse loading on a plate is represented by the deflection and rotation of the normals of the middle plane of the plate, and they will be independent of z and a function of only x and y. The element to be developed to model such plate structures is aptly known as the *plate element*. The formulation of a plate element is very much the same as for the 2D solid element, except for the process for deriving the strain matrix in which the theory of plates is used.

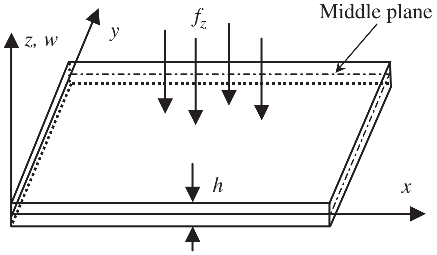


Figure 8.1. A plate and its coordinate system.

Plate elements are normally used to analyse the bending deformation of plate structures and the resulting forces such as shear forces and moments. In this aspect, it is similar to the beam element developed in Chapter 5, except that the plate element is two-dimensional whereas the beam element is one-dimensional. Like the 2D solid element, a plate element can also be triangular, rectangular or quadrilateral in shape. In this book, we cover the development of the rectangular element only, as it is often used. Matrices for the triangular element can also be developed easily using similar procedures, and those for the quadrilateral element can be developed using the idea of an isoparametric element discussed for 2D solid elements. In fact, the development of a quadrilateral element is much the same as the rectangular element, except for an additional procedure of coordinate mapping, as shown for the case of 2D solid elements.

There are a number of theories that govern the deformation of plates. In this chapter, rectangular elements based on the Mindlin plate theory that works for thick plates will be developed. Most books go into great detail to first cover plate elements based on the thin plate theory. However, most finite element packages do not use plate elements based on thin plate theory. In fact, most analysis packages like ABAQUS do not even offer the choice of plate elements. Instead, one has to use the more general shell elements, also discussed in this chapter. Furthermore, using the thin plate theory to develop the finite element equations has a problem, in that the elements developed are usually incompatible or ‘non-conforming’. This means that some components of the rotational displacements may not be continuous on the edges between elements. This is because the rotation depends only upon the deflection w in the thin plate theory, and hence the assumed function for w has to be used to calculate the rotation. Many texts go into even greater detail to explain the concept, and to prove the conformability of many kinds of thin plate elements. To our knowledge, there is really no need, practically, to understand such a concept and proof for readers who are interested in using the finite element method to solve real-life problems. In addition, many structures may not be considered as a ‘thin plate’, or rather their transverse shear strains cannot be ignored. Therefore, the Reissner–Mindlin plate theory is more suitable in general, and the elements developed based on the Reissner–Mindlin plate theory are more practical and useful. This book will only discuss the elements developed based on the Reissner–Mindlin plate theory.

There are a number of higher order plate theories that can be used for the development of finite elements. Since these higher order plate theories are extensions of the

Reissner–Mindlin plate theory, there should be no difficulty for readers who can formulate the Mindlin plate element to understand the formulation of higher order plate elements.

It is assumed that the element has a uniform thickness h. If the plate structure has a varying thickness, the structure has to be divided into small elements that can be treated as uniform elements. However, the formulation of plate elements with a varying thickness can also be done, as the procedure is similar to that of a uniform element; this would be good homework practice for readers after reading this chapter.

Consider now a plate that is represented by a two-dimensional domain in the x–y plane, as shown in Figure 8.1. The plate is divided in a proper manner into a number of *rectangular elements*, as shown in Figure 8.2. Each element will have four nodes and four straight edges. At a node, the degrees of freedom include the deflection w, the rotation about x axis θ_x, and the rotation about y axis θ_y, making the total DOF of each node three. Hence, for a rectangular element with four nodes, the total DOF of the element would be twelve.

Following the Reissner–Mindlin plate theory (see Chapter 2), its shear deformation will force the cross-section of the plate to rotate in the way shown in Figure 8.3. Any straight fibre that is perpendicular to the middle plane of the plate before the deformation rotates, but remains straight after the deformation. The two displacement components that are parallel

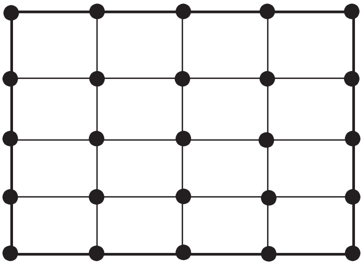


Figure 8.2. 2D domain of a plate meshed by rectangular elements.

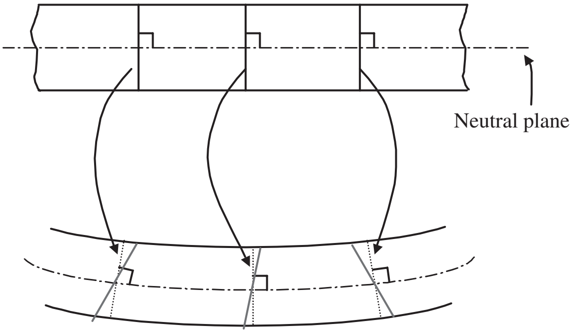


Figure 8.3. Shear deformation in a plate. A straight fibre that is perpendicular to the middle plane of the plate before deformation rotates but remains straight after deformation.

to the middle surface can then be expressed mathematically as

$$\begin{aligned} u(x, y, z) &= z\theta_y(x, y) \\ v(x, y, z) &= -z\theta_y(x, y) \end{aligned} \tag{8.1}$$

where θ_x and θ_y are, respectively, the rotations of the fibre of the plate with respect to the x and y axes. The in-plane strains can then be given as

$$\boldsymbol{\varepsilon} = -z\boldsymbol{\chi} \tag{8.2}$$

where $\boldsymbol{\chi}$ is the curvature, given as

$$\boldsymbol{\chi} = \mathbf{L}\boldsymbol{\theta} = \begin{Bmatrix} -\partial\theta_y/\partial x \\ \partial\theta_x/\partial y \\ \partial\theta_x/\partial x - \partial\theta_y/\partial y \end{Bmatrix} \tag{8.3}$$

in which $\mathbf{L}$ is the differential operator defined in Chapter 2, and is re-written as

$$\mathbf{L} = \begin{bmatrix} -\dfrac{\partial}{\partial x} & 0 \\ 0 & \dfrac{\partial}{\partial y} \\ \dfrac{\partial}{\partial x} & -\dfrac{\partial}{\partial y} \end{bmatrix} \tag{8.4}$$

The off-plane shear strain is then given as

$$\boldsymbol{\gamma} = \begin{Bmatrix} \xi_{xz} \\ \xi_{yz} \end{Bmatrix} = \begin{Bmatrix} \theta_y + \dfrac{\partial w}{\partial x} \\ -\theta_x + \dfrac{\partial w}{\partial y} \end{Bmatrix} \tag{8.5}$$

Note that Hamilton's principle uses energy functions for derivation of the equation of motion. The potential (strain) energy expression for a thick plate element is

$$U_e = \frac{1}{2}\int_{A_e}\int_0^h \boldsymbol{\varepsilon}^T\boldsymbol{\sigma}\,\mathrm{d}A\,\mathrm{d}z + \frac{1}{2}\int_{A_e}\int_0^h \boldsymbol{\tau}^T\boldsymbol{\gamma}\,\mathrm{d}A\,\mathrm{d}z \tag{8.6}$$

The first term on the right-hand side of Eq. (8.6) is for the in-plane stresses and strains, whereas the second term is for the transverse stresses and strains. $\boldsymbol{\tau}$ is the average shear stresses relating to the shear strain in the form

$$\boldsymbol{\tau} = \begin{Bmatrix} \tau_{xz} \\ \tau_{yz} \end{Bmatrix} = \kappa\begin{bmatrix} G & 0 \\ 0 & G \end{bmatrix}\boldsymbol{\gamma} = \kappa\mathbf{c}_s\boldsymbol{\gamma} \tag{8.7}$$

where G is the shear modulus, and κ is a constant that is usually taken to be $\pi^2/12$ or 5/6. Substituting Eqs. (8.2) and (8.7) into Eq. (8.6), the potential (strain) energy becomes

$$U_e = \frac{1}{2}\int_{A_e}\frac{h^3}{12}\boldsymbol{\chi}^T\mathbf{c}\boldsymbol{\chi}\,\mathrm{d}A + \frac{1}{2}\int_{A_e}\kappa h\boldsymbol{\gamma}^T\mathbf{c}_s\boldsymbol{\gamma}\,\mathrm{d}A \tag{8.8}$$

The kinetic energy of the thick plate is given by

$$T_e = \frac{1}{2}\int_{V_e} \rho(\dot{u}^2 + \dot{v}^2 + \dot{w}^2)\,\mathrm{d}V \tag{8.9}$$

which is basically a summation of the contributions of three velocity components in the x, y and z directions of all the particles in the entire domain of the plate. Substituting Eq. (8.1) into the above equation leads to

$$T_e = \frac{1}{2}\int_{A_e} \rho\left(h\dot{w}^2 + \frac{h^3}{12}\dot{\theta}_x^2 + \frac{h^3}{12}\dot{\theta}_y^2\right)\mathrm{d}A = \frac{1}{2}\int_{A_e}(\mathbf{d}^T\mathbf{I}\,\mathbf{d})\,\mathrm{d}A \tag{8.10}$$

where

$$\mathbf{d} = \begin{Bmatrix} w \\ \theta_x \\ \theta_y \end{Bmatrix} \tag{8.11}$$

and

$$\mathbf{I} = \begin{bmatrix} \rho h & 0 & 0 \\ 0 & \rho h^3/12 & 0 \\ 0 & 0 & \rho h^3/12 \end{bmatrix} \tag{8.12}$$

As we can see from Eq. (8.10), the terms related to in-plane displacements are less important for thin plates, since it is proportional to the cubic of the plate thickness.

8.2.1 Shape Functions

It can be seen from the above analysis of the constitutive equations that the rotations, θ_x and θ_y are independent of the deflection w. Therefore, when it comes to interpolating the generalized displacements, the deflection and rotations can actually be interpolated separately using independent shape functions. Therefore, the procedure of field variable interpolation is the same as that for 2D solid problems, except that there are three instead of two DOFs, for a node.

For four-node rectangular thick plate elements, the deflection and rotations can be summed as

$$w = \sum_{i=1}^{4} N_i w_i, \quad \theta_x = \sum_{i=1}^{4} N_i \theta_{x_i}, \quad \theta_y = \sum_{i=1}^{4} N_i \theta_{y_i} \tag{8.13}$$

where the shape function N_i is the same as the four-node 2D solid element in Chapter 7, i.e.

$$N_i = \tfrac{1}{4}(1 + \xi_i\xi)(1 + \eta_i\eta) \tag{8.14}$$

The element constructed will be a conforming element, meaning that w, θ_x and θ_y are continuous on the edges between elements. Rewriting Eq. (8.13) into a single matrix equation, we have

$$\begin{Bmatrix} w \\ \theta_x \\ \theta_y \end{Bmatrix}^h = \mathbf{N}\,\mathbf{d}_e \tag{8.15}$$

where $\mathbf{d}_e$ is the (generalized) displacement vector for all the nodes in the element, arranged in the order

$$\mathbf{d}_e = \begin{Bmatrix} w_1 \\ \theta_{x1} \\ \theta_{y1} \\ w_2 \\ \theta_{x2} \\ \theta_{y2} \\ w_3 \\ \theta_{x3} \\ \theta_{y3} \\ w_4 \\ \theta_{x4} \\ \theta_{y4} \end{Bmatrix}_e \begin{matrix} \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} \text{displacement at node 1} \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} \text{displacement at node 2} \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} \text{displacement at node 3} \\ \left.\begin{matrix} \\ \\ \\ \end{matrix}\right\} \text{displacement at node 4} \end{matrix} \tag{8.16}$$

and the shape function matrix is arranged in the order

$$\mathbf{N} = \left[\underbrace{\begin{matrix} N_1 & 0 & 0 \\ 0 & N_1 & 0 \\ 0 & 0 & N_1 \end{matrix}}_{\text{node 1}} \; \underbrace{\begin{matrix} N_2 & 0 & 0 \\ 0 & N_2 & 0 \\ 0 & 0 & N_2 \end{matrix}}_{\text{node 2}} \; \underbrace{\begin{matrix} N_3 & 0 & 0 \\ 0 & N_3 & 0 \\ 0 & 0 & N_3 \end{matrix}}_{\text{node 3}} \; \underbrace{\begin{matrix} N_4 & 0 & 0 \\ 0 & N_4 & 0 \\ 0 & 0 & N_4 \end{matrix}}_{\text{node 4}} \right] \tag{8.17}$$

8.2.2 Element Matrices

Once the shape function and nodal variables have been defined, element matrices can then be formulated following the standard procedure given in Chapter 7 for 2D solid elements. The only difference is that there are three DOFs at one node for plate elements.

To obtain the element mass matrix $\mathbf{m}_e$ and the element stiffness matrix $\mathbf{k}_e$, we have to use the energy functions given by Eqs. (8.8) and (8.9) and Hamilton's principle. Substituting Eq. (8.15) into the kinetic energy function, Eq. (8.9) gives

$$T_e = \tfrac{1}{2}\dot{\mathbf{d}}_e^T \mathbf{m}_e \, \dot{\mathbf{d}}_e \tag{8.18}$$

where the mass matrix $\mathbf{m}_e$ is given as

$$\mathbf{m}_e = \int_{A_e} \mathbf{N}^{\mathrm{T}} \mathbf{I} \mathbf{N} \, \mathrm{d}A \tag{8.19}$$

The above integration can be carried out analytically, but it will not be detailed in this book. Details can be obtained from Petyt [1990]. In practice, we often perform the integration numerically using the Gauss integration scheme, discussed in Chapter 7.

To obtain the stiffness matrix $\mathbf{k}_e$, we substitute Eq. (8.15) into Eq. (8.6), from which we obtain

$$\mathbf{k}_e = \int_{A_e} \frac{h^3}{12}[\mathbf{B}^{\mathrm{I}}]^{\mathrm{T}}\mathbf{c}\mathbf{B}^{\mathrm{I}}\,\mathrm{d}A + \int_{A_e} \kappa h[\mathbf{B}^{\mathrm{O}}]^{\mathrm{T}}\mathbf{c}_s\mathbf{B}^{\mathrm{O}}\,\mathrm{d}A \tag{8.20}$$

The first term in the above equation represents the strain energy associated with the in-plane stress and strains. The strain matrix $\mathbf{B}^{\mathrm{I}}$ has the form of

$$\mathbf{B}^{\mathrm{I}} = \begin{bmatrix}\mathbf{B}_1^{\mathrm{I}} & \mathbf{B}_2^{\mathrm{I}} & \mathbf{B}_3^{\mathrm{I}} & \mathbf{B}_4^{\mathrm{I}}\end{bmatrix} \tag{8.21}$$

where

$$\mathbf{B}_{\mathrm{j}}^{\mathrm{I}} = \begin{bmatrix} 0 & 0 & -\partial N_j/\partial x \\ 0 & \partial N_j/\partial y & 0 \\ 0 & \partial N_j/\partial x & -\partial N_j/\partial y \end{bmatrix} \tag{8.22}$$

Using the expression for the shape functions in Eq. (8.14), we obtain

$$\begin{aligned} \frac{\partial N_j}{\partial x} &= \frac{\partial N_j}{\partial \xi}\frac{\partial \xi}{\partial x} = \frac{1}{4a}\xi_i(1+\eta_i\eta) \\ \frac{\partial N_j}{\partial y} &= \frac{\partial N_j}{\partial \eta}\frac{\partial \eta}{\partial y} = \frac{1}{4b}(1+\xi_i\xi)\eta_i \end{aligned} \tag{8.23}$$

In deriving Eq. (8.23), the relationship $\xi = x/a$, $\eta = y/b$ has been employed.

The second term in Eq. (8.20) relates to the strain energy associated with the off-plane shear stress and strain. The strain matrix $\mathbf{B}^{\mathrm{O}}$ has the form

$$\mathbf{B}^{\mathrm{O}} = \begin{bmatrix}\mathbf{B}_1^{\mathrm{O}} & \mathbf{B}_2^{\mathrm{O}} & \mathbf{B}_3^{\mathrm{O}} & \mathbf{B}_4^{\mathrm{O}}\end{bmatrix} \tag{8.24}$$

where

$$\mathbf{B}_{\mathrm{j}}^{\mathrm{O}} = \begin{bmatrix} \partial N_j/\partial x & 0 & N_j \\ \partial N_j/\partial y & -N_j & 0 \end{bmatrix} \tag{8.25}$$

The integration in the stiffness matrix $\mathbf{k}_e$, Eq. (8.20) can be evaluated analytically as well. Practically, however, the Gauss integration scheme is used to evaluate the integrations numerically. Note that when the thickness of the plate is reduced, the element becomes over-stiff, a phenomenon that relates to so-called ‘shear locking’. The simplest and most practical means to solve this problem is to use 2 × 2 Gauss points for the integration of the first term, and use only one Gauss point for the second term in Eq. (8.20).

As for the force vector, we substitute the interpolation of the generalized displacements, given in Eq. (8.15), into the usual equation, as in Eq. (3.81):

$$\mathbf{f}_e = \int_{A_e} \mathbf{N}^T \begin{Bmatrix} f_z \\ 0 \\ 0 \end{Bmatrix} \mathrm{d}A \tag{8.26}$$

which gives the equivalent nodal force vector for the element. If the load is uniformly distributed in the element, f_z is constant, and the above equation becomes

$$\mathbf{f}_e^T = abf_z \begin{Bmatrix} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \end{Bmatrix} \tag{8.27}$$

Equation (8.27) implies that the distributed force is divided evenly into four concentrated forces of one quarter of the total force.

8.2.3 Higher Order Elements

For an eight-node rectangular thick plate element, the deflection and rotations can be summed as

$$w = \sum_{i=1}^{8} N_i w_i, \quad \theta_x = \sum_{i=1}^{8} N_i \theta_{x_i}, \quad \theta_y = \sum_{i=1}^{8} N_i \theta_{y_i} \tag{8.28}$$

where the shape function N_i is the same as the eight-node 2D solid element given by Eq. (7.52). The element constructed will be a conforming element, as w, θ_x and θ_y are continuous on the edges between elements. The formulation procedure is the same as for the rectangular plate elements.

8.3 SHELL ELEMENTS

A shell structure carries loads in all directions, and therefore undergoes bending and twisting, as well as in-plane deformation. Some common examples would be the dome-like design of the roof of a building with a large volume of space; or a building with special architectural requirements such as a church or mosque; or structures with a special functional requirement such as cylindrical and hemispherical water tanks; or lightweight structures like the fuselage of an aircraft, as shown in Figure 8.4. Shell elements have to be used for modelling such structures. The simplest but widely used shell element can be formulated easily by combining the 2D solid element formulated in Chapter 7 and the plate element formulated in the previous section. The 2D solid elements handle the *membrane* or in-plane effects, while the plate elements are used to handle *bending* or off-plane effects. The procedure for developing such an element is very similar to the short cut method used to formulate the frame elements using the truss and beam elements, as discussed in Chapter 6. Of course, the shell element can also be formulated using the usual method of defining shape functions, substituting into the constitutive equations, and thus obtaining the element matrices. However, as you might have guessed, it is going to be very tedious. Bear in mind, however, that the basic concept of deriving the finite element equation still holds, though we will be introducing a so-called short cut method. In this book, the derivation for four-nodal, rectangular shell elements will be outlined using the short cut method.

Figure 8.4. The fuselage of an aircraft can be considered to be a typical shell structure.

Since the plate structure can be treated as a special case of the shell structure, the shell element developed in this section is applicable for modelling plate structures. In fact, it is common practice to use a shell element offered in a commercial FE package to analyse plate structures.

8.3.1 Elements in Local Coordinate Systems

Shell structures are usually curved. We assume that the shell structure is divided into shell elements that are flat. The curvature of the shell is then followed by changing the orientation of the shell elements in space. Therefore, if the curvature of the shell is very large, a fine mesh of elements has to be used. This assumption sounds rough, but it is very practical and widely used in engineering practice. There are alternatives of more accurately formulated shell elements, but they are used only in academic research and have never been implemented in any commercially available software packages. Therefore, this book formulates only flat shell elements.

Similar to the frame structure, there are six DOFs at a node for a shell element: three translational displacements in the x, y and z directions, and three rotational deformations with respect to the x, y and z axes. Figure 8.5 shows the middle plane of a rectangular shell element and the DOFs at the nodes. The generalized displacement vector for the element can be written as

$$\mathbf{d}_e = \begin{Bmatrix} \mathbf{d}_{e1} \\ \mathbf{d}_{e2} \\ \mathbf{d}_{e3} \\ \mathbf{d}_{e4} \end{Bmatrix} \begin{array}{l} \text{node 1} \\ \text{node 2} \\ \text{node 3} \\ \text{node 4} \end{array} \tag{8.29}$$

where $\mathbf{d}_{ei}$ $(i = 1, 2, 3, 4)$ are the displacement vector at node i:

$$\mathbf{d}_{ei} = \begin{Bmatrix} u_i \\ v_i \\ w_i \\ \theta_{xi} \\ \theta_{yi} \\ \theta_{zi} \end{Bmatrix} \begin{array}{l} \text{displacement in } x \text{ direction} \\ \text{displacement in } y \text{ direction} \\ \text{displacement in } z \text{ direction} \\ \text{rotation about } x\text{-axis} \\ \text{rotation about } y\text{-axis} \\ \text{rotation about } z\text{-axis} \end{array} \tag{8.30}$$

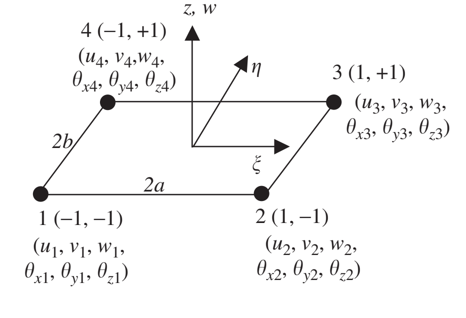

z, w
4 (−1, +1)
$(u_4, v_4, w_4, \theta_{x4}, \theta_{y4}, \theta_{z4})$
η
3 (1, +1)
$(u_3, v_3, w_3, \theta_{x3}, \theta_{y3}, \theta_{z3})$
$2b$
ξ
$2a$
1 (−1, −1)
$(u_1, v_1, w_1, \theta_{x1}, \theta_{y1}, \theta_{z1})$
2 (1, −1)
$(u_2, v_2, w_2, \theta_{x2}, \theta_{y2}, \theta_{z2})$

Figure 8.5. The middle plane of a rectangular shell element.

The stiffness matrix for a 2D solid, rectangular element is used for dealing with the membrane effects of the element, which corresponds to DOFs of u and v. The membrane stiffness matrix can thus be expressed in the following form using sub-matrices according to the nodes:

$$\mathbf{k}_e^m = \left[\begin{array}{cccc} \text{node1} & \text{node 2} & \text{node 3} & \text{node 4} \\ \mathbf{k}_{11}^m & \mathbf{k}_{12}^m & \mathbf{k}_{13}^m & \mathbf{k}_{14}^m \\ \mathbf{k}_{21}^m & \mathbf{k}_{22}^m & \mathbf{k}_{23}^m & \mathbf{k}_{24}^m \\ \mathbf{k}_{31}^m & \mathbf{k}_{32}^m & \mathbf{k}_{33}^m & \mathbf{k}_{34}^m \\ \mathbf{k}_{41}^m & \mathbf{k}_{42}^m & \mathbf{k}_{43}^m & \mathbf{k}_{44}^m \end{array}\right] \begin{array}{l} \\ \text{node 1} \\ \text{node 2} \\ \text{node 3} \\ \text{node 4} \end{array} \tag{8.31}$$

where the superscript m stands for the membrane matrix. Each sub-matrix will have a dimension of 2×2, since it corresponds to the two DOFs u and v at each node. Note again that the matrix above is actually the same as the stiffness matrix of the 2D rectangular, solid element, except it is written in terms of sub-matrices according to the nodes.

The stiffness matrix for a rectangular plate element is used for the bending effects, corresponding to DOFs of w, and θ_x, θ_y. The bending stiffness matrix can thus be expressed in the following form using sub-matrices according to the nodes:

$$\mathbf{k}_e^b = \left[\begin{array}{cccc} \text{node 1} & \text{node 2} & \text{node 3} & \text{node 4} \\ \mathbf{k}_{11}^b & \mathbf{k}_{12}^b & \mathbf{k}_{13}^b & \mathbf{k}_{14}^b \\ \mathbf{k}_{21}^b & \mathbf{k}_{22}^b & \mathbf{k}_{23}^b & \mathbf{k}_{24}^b \\ \mathbf{k}_{31}^b & \mathbf{k}_{32}^b & \mathbf{k}_{33}^b & \mathbf{k}_{34}^b \\ \mathbf{k}_{41}^b & \mathbf{k}_{42}^b & \mathbf{k}_{43}^b & \mathbf{k}_{44}^b \end{array}\right] \begin{array}{l} \\ \text{node 1} \\ \text{node 2} \\ \text{node 3} \\ \text{node 4} \end{array} \tag{8.32}$$

where the superscript b stands for the bending matrix. Each bending sub-matrix has a dimension of 3×3.

The stiffness matrix for the shell element in the local coordinate system is then formulated by combining Eqs. (8.31) and (8.32):

$$\mathbf{k}_e = \left[\begin{array}{cccccccccccc} \multicolumn{3}{c}{\overbrace{\hphantom{\mathbf{k}_{11}^m\ \mathbf{0}\ 0}}^{\text{node 1}}} & \multicolumn{3}{c}{\overbrace{\hphantom{\mathbf{k}_{11}^m\ \mathbf{0}\ 0}}^{\text{node 2}}} & \multicolumn{3}{c}{\overbrace{\hphantom{\mathbf{k}_{11}^m\ \mathbf{0}\ 0}}^{\text{node 3}}} & \multicolumn{3}{c}{\overbrace{\hphantom{\mathbf{k}_{11}^m\ \mathbf{0}\ 0}}^{\text{node 4}}} \\
\mathbf{k}_{11}^m & \mathbf{0} & 0 & \mathbf{k}_{12}^m & \mathbf{0} & 0 & \mathbf{k}_{13}^m & \mathbf{0} & 0 & \mathbf{k}_{14}^m & \mathbf{0} & 0 \\
\mathbf{0} & \mathbf{k}_{11}^b & 0 & \mathbf{0} & \mathbf{k}_{12}^b & 0 & \mathbf{0} & \mathbf{k}_{13}^b & 0 & \mathbf{0} & \mathbf{k}_{14}^b & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mathbf{k}_{21}^m & \mathbf{0} & 0 & \mathbf{k}_{22}^m & \mathbf{0} & 0 & \mathbf{k}_{23}^m & \mathbf{0} & 0 & \mathbf{k}_{24}^m & \mathbf{0} & 0 \\
\mathbf{0} & \mathbf{k}_{21}^b & 0 & \mathbf{0} & \mathbf{k}_{22}^b & 0 & \mathbf{0} & \mathbf{k}_{23}^b & 0 & \mathbf{0} & \mathbf{k}_{24}^b & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mathbf{k}_{31}^m & \mathbf{0} & 0 & \mathbf{k}_{32}^m & \mathbf{0} & 0 & \mathbf{k}_{33}^m & \mathbf{0} & 0 & \mathbf{k}_{34}^m & \mathbf{0} & 0 \\
\mathbf{0} & \mathbf{k}_{31}^b & 0 & \mathbf{0} & \mathbf{k}_{32}^b & 0 & \mathbf{0} & \mathbf{k}_{33}^b & 0 & \mathbf{0} & \mathbf{k}_{34}^b & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mathbf{k}_{41}^m & \mathbf{0} & 0 & \mathbf{k}_{42}^m & \mathbf{0} & 0 & \mathbf{k}_{43}^m & \mathbf{0} & 0 & \mathbf{k}_{44}^m & \mathbf{0} & 0 \\
\mathbf{0} & \mathbf{k}_{41}^b & 0 & \mathbf{0} & \mathbf{k}_{42}^b & 0 & \mathbf{0} & \mathbf{k}_{43}^b & 0 & \mathbf{0} & \mathbf{k}_{44}^b & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right] \begin{array}{l} \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \text{node 1} \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \text{node 2} \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \text{node 3} \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \text{node 4} \end{array} \tag{8.33}$$

The stiffness matrix for a rectangular shell matrix has a dimension of 24×24. Note that in Eq. (8.33), the components related to the DOF θ_z, are zeros. This is because there is no θ_z in the local coordinate system. If these zero terms are removed, the stiffness matrix would have a reduced dimension of 20 × 20. However, using the extended 24 × 24 stiffness matrix will make it more convenient for transforming the matrix from the local coordinate system into the global coordinate system.

Similarly, the mass matrix for a rectangular element can be obtained in the same way as the stiffness matrix. The mass matrix for the 2D solid element is used for the membrane effects, corresponding to DOFs of u and v. The membrane mass matrix can be expressed in the following form using sub-matrices according to the nodes:

$$\mathbf{m}_e^m = \begin{array}{c} \begin{array}{cccc} \text{node 1} & \text{node 2} & \text{node 3} & \text{node 4} \end{array} \\ \left[\begin{array}{cccc} \mathbf{m}_{11}^m & \mathbf{m}_{12}^m & \mathbf{m}_{13}^m & \mathbf{m}_{14}^m \\ \mathbf{m}_{21}^m & \mathbf{m}_{22}^m & \mathbf{m}_{23}^m & \mathbf{m}_{24}^m \\ \mathbf{m}_{31}^m & \mathbf{m}_{32}^m & \mathbf{m}_{33}^m & \mathbf{m}_{34}^m \\ \mathbf{m}_{41}^m & \mathbf{m}_{42}^m & \mathbf{m}_{43}^m & \mathbf{m}_{44}^m \end{array}\right] \end{array} \begin{array}{l} \\ \text{node 1} \\ \text{node 2} \\ \text{node 3} \\ \text{node 4} \end{array} \tag{8.34}$$

where the superscript m stands for the membrane matrix. Each membrane sub-matrix has a dimension of 2 × 2.

The mass matrix for a rectangular plate element is used for the bending effects, corresponding to DOFs of w, and θ_x, θ_y. The bending mass matrix can also be expressed in the following form using sub-matrices according to the nodes:

$$\mathbf{m}_e^b = \begin{array}{c} \begin{array}{cccc} \text{node 1} & \text{node 2} & \text{node 3} & \text{node 4} \end{array} \\ \left[\begin{array}{cccc} \mathbf{m}_{11}^b & \mathbf{m}_{12}^b & \mathbf{m}_{13}^b & \mathbf{m}_{14}^b \\ \mathbf{m}_{21}^b & \mathbf{m}_{22}^b & \mathbf{m}_{23}^b & \mathbf{m}_{24}^b \\ \mathbf{m}_{31}^b & \mathbf{m}_{32}^b & \mathbf{m}_{33}^b & \mathbf{m}_{34}^b \\ \mathbf{m}_{41}^b & \mathbf{m}_{42}^b & \mathbf{m}_{43}^b & \mathbf{m}_{44}^b \end{array}\right] \end{array} \begin{array}{l} \\ \text{node 1} \\ \text{node 2} \\ \text{node 3} \\ \text{node 4} \end{array} \tag{8.35}$$

where the superscript b stands for the bending matrix. Each bending sub-matrix has a dimension of 3 × 3.

The mass matrix for the shell element in the local coordinate system is then formulated by combining Eqs. (8.34) and (8.35):

$$\mathbf{m}_e = \begin{array}{c} \begin{array}{cccc} \overbrace{\qquad\qquad\qquad}^{\text{node 1}} & \overbrace{\qquad\qquad\qquad}^{\text{node 2}} & \overbrace{\qquad\qquad\qquad}^{\text{node 3}} & \overbrace{\qquad\qquad\qquad}^{\text{node 4}} \end{array} \\ \left[\begin{array}{cccccccccccc}
\mathbf{m}_{11}^m & \mathbf{0} & 0 & \mathbf{m}_{12}^m & \mathbf{0} & 0 & \mathbf{m}_{13}^m & \mathbf{0} & 0 & \mathbf{m}_{14}^m & \mathbf{0} & 0 \\
\mathbf{0} & \mathbf{m}_{11}^b & 0 & \mathbf{0} & \mathbf{m}_{12}^b & 0 & \mathbf{0} & \mathbf{m}_{13}^b & 0 & \mathbf{0} & \mathbf{m}_{14}^b & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mathbf{m}_{21}^m & \mathbf{0} & 0 & \mathbf{m}_{22}^m & \mathbf{0} & 0 & \mathbf{m}_{23}^m & \mathbf{0} & 0 & \mathbf{m}_{24}^m & \mathbf{0} & 0 \\
\mathbf{0} & \mathbf{m}_{21}^b & 0 & \mathbf{0} & \mathbf{m}_{22}^b & 0 & \mathbf{0} & \mathbf{m}_{23}^b & 0 & \mathbf{0} & \mathbf{m}_{24}^b & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mathbf{m}_{31}^m & \mathbf{0} & 0 & \mathbf{m}_{32}^m & \mathbf{0} & 0 & \mathbf{m}_{33}^m & \mathbf{0} & 0 & \mathbf{m}_{34}^m & \mathbf{0} & 0 \\
\mathbf{0} & \mathbf{m}_{31}^b & 0 & \mathbf{0} & \mathbf{m}_{32}^b & 0 & \mathbf{0} & \mathbf{m}_{33}^b & 0 & \mathbf{0} & \mathbf{m}_{34}^b & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\mathbf{m}_{41}^m & \mathbf{0} & 0 & \mathbf{m}_{42}^m & \mathbf{0} & 0 & \mathbf{m}_{43}^m & \mathbf{0} & 0 & \mathbf{m}_{44}^m & \mathbf{0} & 0 \\
\mathbf{0} & \mathbf{m}_{41}^b & 0 & \mathbf{0} & \mathbf{m}_{42}^b & 0 & \mathbf{0} & \mathbf{m}_{43}^b & 0 & \mathbf{0} & \mathbf{m}_{44}^b & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{array}\right] \end{array} \begin{array}{l} \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \text{node 1} \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \text{node 2} \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \text{node 3} \\ \left.\begin{array}{l} \\ \\ \\ \end{array}\right\} \text{node 4} \end{array} \tag{8.36}$$

Similarly, it is noted that the terms corresponding to the DOF θ_z are zero for the same reasons as explained for the stiffness matrix.

8.3.2 Elements in Global Coordinate System

The matrices for shell elements in the global coordinate system can be obtained by performing the transformations

$$\mathbf{K}_e = \mathbf{T}^T \mathbf{k}_e \mathbf{T} \tag{8.37}$$

$$\mathbf{M}_e = \mathbf{T}^T \mathbf{m}_e \mathbf{T} \tag{8.38}$$

$$\mathbf{F}_e = \mathbf{T}^T \mathbf{f}_e \tag{8.39}$$

where $\mathbf{T}$ is the transformation matrix, given by

$$\mathbf{T} = \begin{bmatrix} \mathbf{T}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{T}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{T}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{T}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{T}_3 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{T}_3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{T}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{T}_3 \end{bmatrix} \tag{8.40}$$

in which

$$\mathbf{T}_3 = \begin{bmatrix} l_x & m_x & n_x \\ l_y & m_y & n_y \\ l_z & m_z & n_z \end{bmatrix} \tag{8.41}$$

where l_k, m_k and n_k $(k = x, y, z)$ are direction cosines, which can be obtained in exactly the same way described in Section 6.3.2. The difference is that there is no need to define the additional point 3, as there are already four nodes for the shell element. The local coordinates x, y, z can be conveniently defined under the global coordinate system using the four nodes of the shell element.

The global matrices obtained will not have zero columns and rows if the elements joined at a node are not in the same plane. If all the elements joined at a node are in the same plane, then the global matrices will be singular. This kind of case is encountered when using shell elements to model a flat plate. In such situations, special techniques, such as a 'stabilizing matrix', have to be used to solve the global system equations.

8.4 REMARKS

The direct superposition of the matrices for 2D solid elements and plate elements are performed by assuming that the membrane effects are not coupled with the bending effects at the individual element level. This implies that the membrane forces will not result in any bending deformation, and bending forces will not cause any in-plane displacement in the element. For a shell structure in space, the membrane and bending effects are actually coupled globally, meaning that the membrane force at an element may result in bending deformations in the other elements, and the bending forces in an element may create in-plane displacements in other elements. The coupling effects are more significant for shell structures with a strong curvature. Therefore, for those structures, a finer element mesh should be used. Using the shell elements developed in this chapter implies that the curved shell structure has to be meshed by piecewise flat elements. This simplification in geometry needs to be taken into account when evaluating the results obtained.

8.5 CASE STUDY: NATURAL FREQUENCIES OF MICRO-MOTOR

In this case study, we examine the natural frequencies and mode shapes of the micro-motor described in Section 7.8. Natural frequencies are properties of a system, and it is important to study the natural frequencies and corresponding mode shapes of a system, because if a forcing frequency is applied to the system near to or at the natural frequency, resonance will occur. That is, there will be very large amplitude vibration that might be disastrous in some situations. In this case study, the flexural vibration modes of the rotor of the micro-motor will be analysed.

8.5.1 Modelling

The geometry of the micro-motor's rotor will be the same as that of Figure 7.22, and the elastic properties will remain unchanged using the properties in Table 7.2. To show the mode shapes more clearly, we model the rotor as a whole rather than as a symmetrical quarter model. However, using a quarter model is still possible, but one has to take note of symmetrical and anti-symmetrical modes (to be discussed in Chapter 11). Figure 8.6 shows the finite element model of the micro-motor containing 480 nodes and 384 elements. To

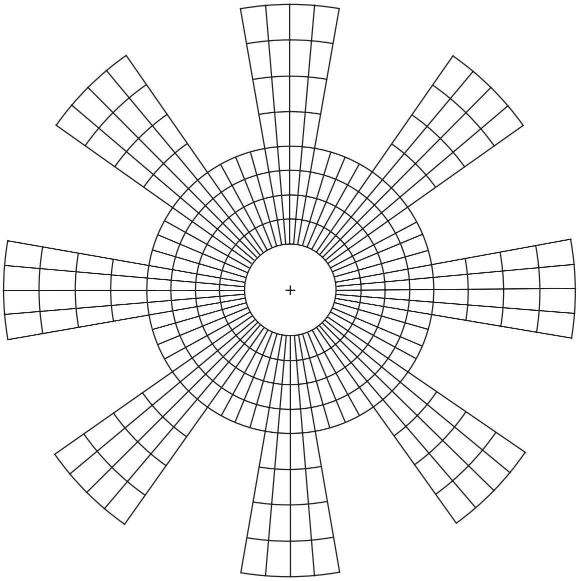


Figure 8.6. Finite element mesh using 2D, four nodal shell elements.

study the flexural vibration modes, plate elements discussed in this chapter ought to be used. However, as mentioned earlier in this chapter, most commercial finite element packages, including ABAQUS, do not allow the use of pure plate elements. Therefore, shell elements will be utilized here for meshing up the model of the micro-motor. 2D, four nodal shell elements (S4) are used. Recall that each shell element has three translational degrees of freedom and three rotational degrees of freedom, and it is actually a superposition of a plate element with a 2D solid element. Hence, to obtain just the flexural modes, we would need to constrain the degrees of freedom corresponding to the x translational displacement and the y translational displacement, as well as the rotation about the z axis. This would leave each shell element with the three degrees of freedom of a plate element. As before, the nodes along the edge of the centre hole will be constrained to be fixed. Since we are interested in the natural frequencies, there will be no external forces on the rotor.

8.5.2 ABAQUS Input File

The ABAQUS input file for the problem described is shown below. Note that some parts are not shown due to the space available in this text.

```
*HEADING, SPARSE
EIGENVALUE ANALYSIS OF MICRO MOTOR
**
*NODE
1, 8., 0.
2, 7.99238, 0.348994
3, 7.96955, 0.697324
4, 7.93155, 1.04427
5, 7.87846, 1.38919
⋮
997, -8.68241, -49.2404
998, -6.52629, -49.5722
999, -4.35774, -49.8097
1000, -2.1809, -49.9524
**
```

Nodal cards

Define the coordinates of the nodes in the model. The first entry being the node ID, while the second and third are the x and y coordinates of the position of the node, respectively.

```
**
*ELEMENT, TYPE=S4, ELSET=MOTOR
1, 1, 6, 7, 2
2, 2, 7, 8, 3
3, 3, 8, 9, 4
4, 4, 9, 10, 5
⋮
830, 994, 998, 999, 995
831, 995, 999, 1000, 996
832, 996, 1000, 760, 755
**
**
**
```

Element (connectivity) cards

Define the element type and what nodes make up the element. S4 represents that it is a four nodal, shell element. The “ELSET = MOTOR” statement is simply for naming this set of elements so that it can be referenced when defining the material properties. In the subsequent data entry, the first entry is the element ID, and the following four entries are the nodes making up the element. The order of the nodes for all elements must be consistent and counter-clockwise.

```
*SHELL SECTION, ELSET=MOTOR, MATERIAL=POLYSI
13.
**
** PolySi
**
*MATERIAL, NAME=POLYSI
**
*DENSITY
2.3E-15,
**
```

Property cards

Define properties to the elements of set "MOTOR". It will have the material properties defined under "POLYSI". The thickness of the elements is also defined in the data line.

```
*ELASTIC, TYPE=ISO
169000., 0.262
**
**
*BOUNDARY, OP=NEW
1, 1,, 0.
```

Material cards

Define material properties under the name "POLYSI". Density and elastic properties are defined. TYPE = ISO represents isotropic properties.

```
1, 2,, 0.
1, 3,, 0.
2, 1,, 0.
2, 2,, 0.
2, 3,, 0.
3, 1,, 0.
⋮
903, 4,, 0.
903, 5,, 0.
903, 6,, 0.
**
** fixedxy
**
*BOUNDARY, OP=NEW
6, 1,, 0.
6, 2,, 0.
7, 1,, 0.
7, 2,, 0.
⋮
997, 6,, 0.
998, 6,, 0.
999, 6,, 0.
1000, 6,, 0.
**
**
** Step 1, freq
** LoadCase, Default
**
*STEP, NLGEOM
This load case is the default load case that always appears
```

BC cards

Define boundary conditions. In this case, all the nodes along the centre circle are constrained to zero displacements. To simulate plate elements, DOFs 1, 2 and 6 are constrained for all the nodes in the model.

```
*FREQUENCY
8, , , , 30
**
**
**
*NODE   PRINT, FREQ=1
U,
*NODE   FILE, FREQ=1
U,
**
**
**
*END STEP
```

Output control cards

Define the output required. In this case, the nodal displacement components (U) are requested.

Control cards

Indicate the analysis step. In this case it is a "FREQUENCY" analysis, which extracts the eigenvalues for the problem.

8.5.3 Solution Process

Looking at the mesh in Figure 8.6, one can see that quadrilateral shell elements are used. Therefore, the equations for a linear, quadrilateral shell element must be formulated by ABAQUS. As before, the formulation of the element matrices would require information from the nodal cards and the element connectivity cards. The element type used here is S4, representing four nodal shell elements. There are other types of shell elements available in the ABAQUS element library.

After the nodal and element cards, next to be considered would be the property and material cards. The properties for the shell element used here must be defined, which in this case includes the material used and the thickness of the shell elements. The material cards are similar to those of the case study in Chapter 7 except that here the density of the material must be included, since we are not carrying out a static analysis as in Chapter 7.

The boundary (BC) cards then define the boundary conditions on the model. In this problem, we would like to obtain only the flexural vibration modes of the motor, hence the components of displacements in the plane of the motor are not actually required. As mentioned, this is very much the characteristic of the plate elements. Therefore, DOFs 1, 2 and 6 corresponding to the x and y displacements, and rotation about the z axis, is constrained. The other boundary condition would be the constraining of the displacements of the nodes at the centre of the motor.

Without the need to define any external loadings, the control cards then define the type of analysis ABAQUS would carry out. ABAQUS uses the sub-space iteration scheme by default to evaluate the eigenvalues of the equation of motion. This method is a very effective method of determining a number of lowest eigenvalues and corresponding eigenvectors for a very large system of several thousand DOFs. The procedure is outlined in the case study in Chapter 5. Finally, the output control cards define the necessary output required by the analyst.

8.5.4 Result and Discussion

Using the input file above, an eigenvalue extraction is carried out in ABAQUS. The output is extracted from the ABAQUS results file showing the first eight natural frequencies and

Table 8.1. Natural frequencies obtained from analyses

Mode	Natural frequencies (MHz)		
	768 triangular elements with 480 nodes	384 quadrilateral elements with 480 nodes	1280 quadrilateral elements with 1472 nodes
1	7.67	5.08	4.86
2	7.67	5.08	4.86
3	7.87	7.44	7.41
4	10.58	8.52	8.30
5	10.58	8.52	8.30
6	13.84	11.69	11.44
7	13.84	11.69	11.44
8	14.86	12.45	12.17

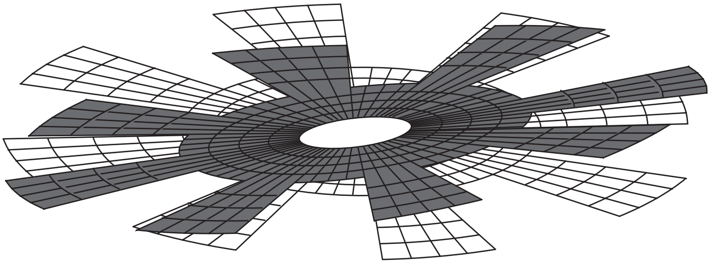


Figure 8.7. Mode 1.

tabulated in Table 8.1. The table also shows results obtained from using triangular elements as well as a finer mesh of quadrilateral elements. It is interesting to note that for certain modes, the eigenvalues and hence the frequencies are repetitive with the previous one. This is due to the symmetry of the circular rotor structure. For example, modes 1 and 2 have the same frequency, and looking at their corresponding mode shapes in Figures 8.7 and 8.8, respectively, one would notice that they are actually of the same shape but bending at a plane 90° from each other. As such, many consider this as one single mode. Therefore, though eight eigenmodes are extracted, it is effectively equivalent to only five eigenmodes. However, to be consistent with the result file from ABAQUS, all the modes extracted will be shown here. Figure 8.9 to 8.14 show the other mode shapes from this analysis. Remember that, since the in-plane displacements are already constrained, these modes are only the flexural modes of the rotor.

Comparing the natural frequencies obtained using 768 triangular elements with those obtained using the quadrilateral elements, one can see that the frequencies are generally higher using the triangular elements. Note that for the same number of nodes, using the quadrilateral elements requires half the number of elements. The results obtained using 384 quadrilateral elements do not differ much from those that use 1280 elements. This again

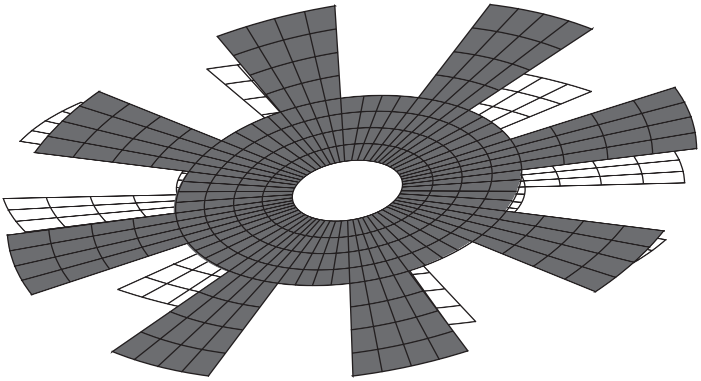


Figure 8.8. Mode 2.

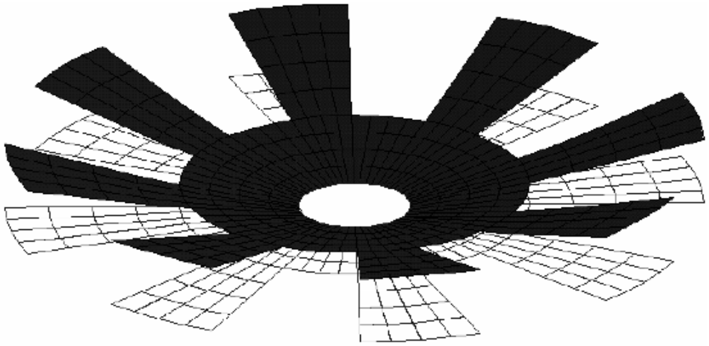


Figure 8.9. Mode 3.

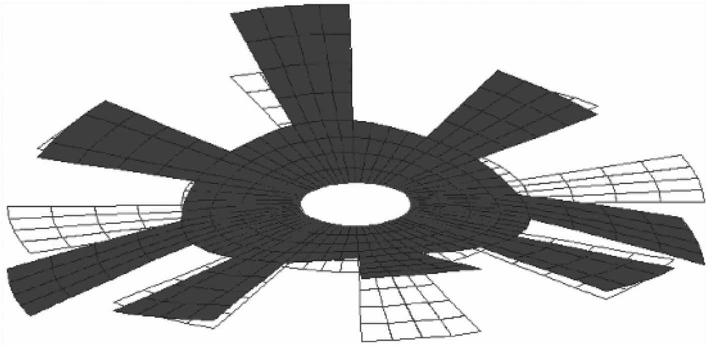


Figure 8.10. Mode 4.

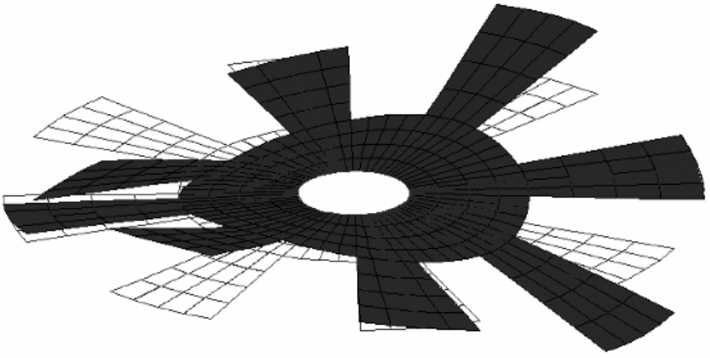


Figure 8.11. Mode 5.

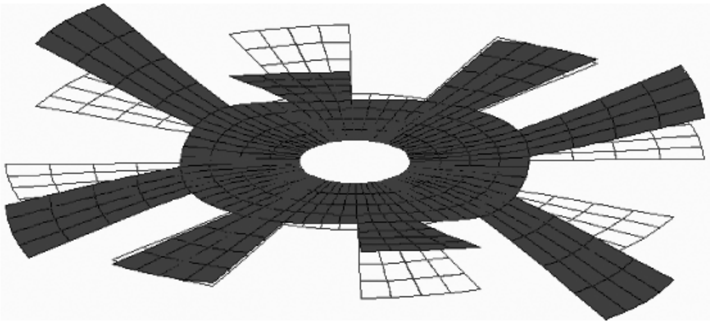


Figure 8.12. Mode 6.

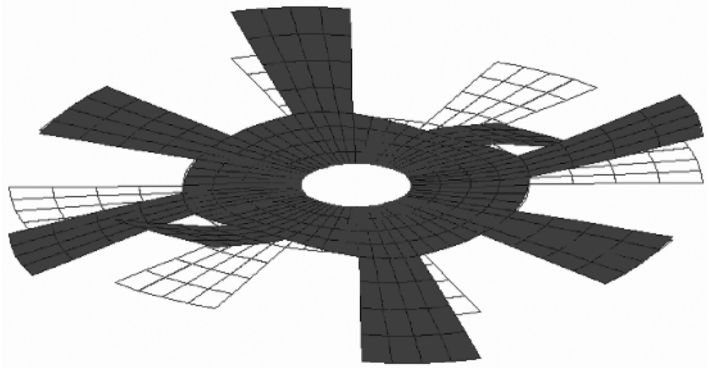


Figure 8.13. Mode 7.

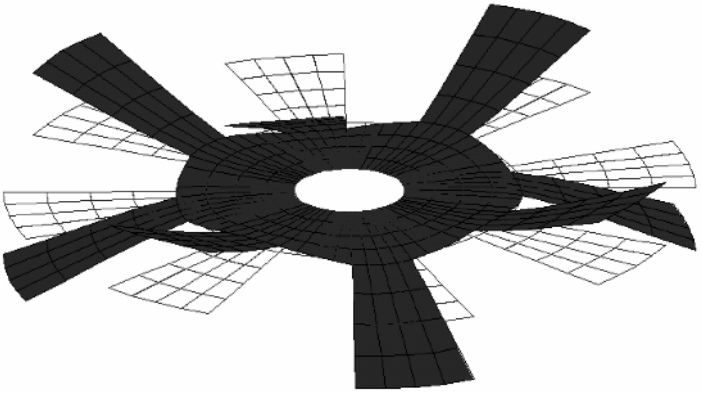


Figure 8.14. Mode 8.

shows that the triangular elements are less accurate than the quadrilateral elements. Note that the mode shapes obtained in the three analyses are the same.

8.6 CASE STUDY: TRANSIENT ANALYSIS OF A MICRO-MOTOR

While analysing the micro-motor, another case study is included here to illustrate an example of a transient analysis using ABAQUS. The same micro-motor shown in Chapter 7 will be analysed here.

The rotor of the micro-motor rotates due to the electrostatic force between the rotor and the stator poles of the motor. Let us assume a hypothetical case where there is a misalignment between the rotor and the stator poles in the motor. As such, there might be other force components acting on the rotor. The actual analysis of such a problem can be very complex, so in this case study we simply analyse a very simple case of the problem with loading conditions as shown in Figure 8.15. It can be seen that symmetrical conditions are used, resulting in a quarter model. The transient response of the transverse displacement components of the various parts of the rotor is to be calculated here.

8.6.1 Modelling

Since we are analysing the same structure as that in Chapter 7, the meshing aspects of the geometry will not be discussed again. It should be noted that an optimum number of elements (nodes) should be used for every finite element analysis. The same treatment of using the shell elements and constraining the necessary DOFs (1, 2 and 6) is carried out to simulate plate elements. The difference here is that there will be loadings in the form of a sinusoidal function with respect to time,

$$F = A \sin \varpi t \tag{8.42}$$

applied as concentrated loadings at the positions shown in Figure 8.15.

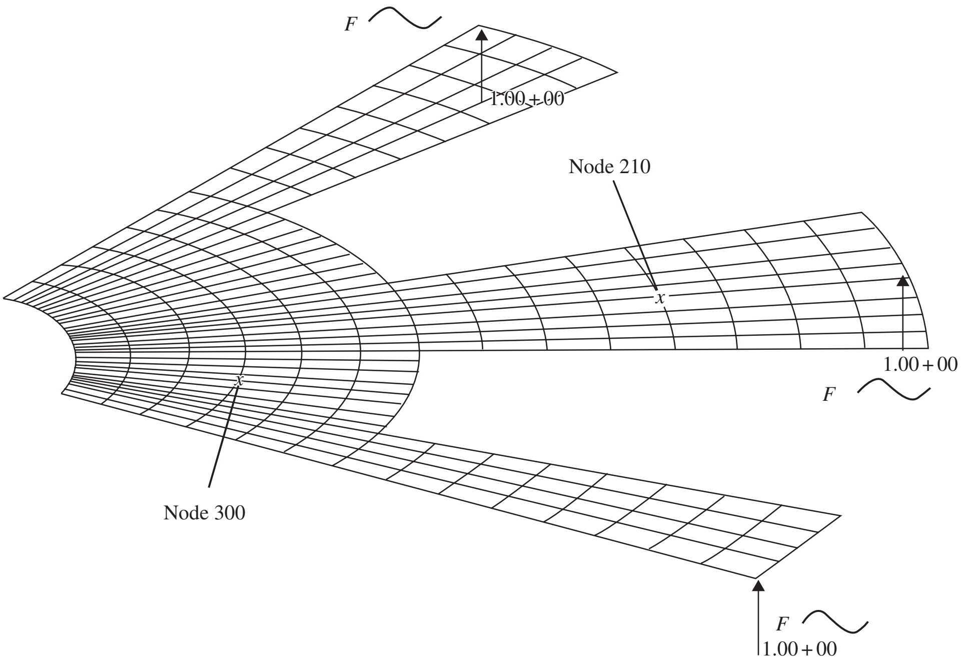


Figure 8.15. Quarter model of micro model with sinosoidal forces applied.

8.6.2 ABAQUS Input File

The ABAQUS input file for the problem described is shown below. Note that some parts are not shown due to the space available in this text.

```
*HEADING
TRANSIENT ANALYSIS OF MICRO MOTOR
**
*NODE
1, 5.46197E-7, 50.
2, 2.1809, 49.9524
3, 4.35774, 49.8097
⋮
380, 46.8304, 2.04468
381, 49.5722, 6.52638
382, 49.9524, 2.18099
383, 49.8097, 4.35783
**
**
```

Nodal cards

Define the coordinates of the nodes in the model. The first entry is the node ID, while the second and third are the *x* and *y* coordinates of the position of the node, respectively.

```
*ELEMENT, TYPE=S4, ELSET=MOTOR
1, 343, 342, 347, 348
2, 342, 341, 346, 347
3, 341, 340, 345, 346
⋮
317, 74, 65, 67, 75
318, 65, 55, 57, 67
319, 55, 43, 45, 57
320, 43, 29, 31, 45
**
**
```

Element (connectivity) cards

Define the element type and what nodes make up the element. S4 represents that it is a four nodal, shell element. The "ELSET=MOTOR" statement is simply for naming this set of elements so that it can be referenced when defining the material properties. In the subsequent data entry, the first entry is the element ID, and the following four entries are the nodes making up the element. The order of the nodes for all elements must be consistent and counter-clockwise.

```
*NSET, NSET=EDGE1
283, 298, 311, 322, 331, 338, 343, 348,
353, 358, 363, 368, 373, 376, 377
**
**
*NSET, NSET=EDGE2, GENERATE
1, 1, 1
6, 6, 1
11, 11, 1
16, 16, 1
21, 31, 1
**
**
```

Node sets

Sets of nodes defined to be used for referencing when defining boundary conditions.

```
*NSET, NSET=CENTER
21, 36, 49, 60, 69, 76, 83, 90,
97, 104, 111, 118, 127, 149, 169, 188,
205, 220, 233, 244, 253, 260, 267, 274,
275, 276, 277, 278, 279, 280, 281, 282,
283,
**
**
*SHELL SECTION, ELSET=MOTOR, MATERIAL=POLYSI
13.,
**
**
```

Property cards

Define properties to the elements of set "MOTOR". It will have the material properties defined under "POLYSI". The thickness of the elements is also defined in the data line.

```
*MATERIAL, NAME=POLYSI
**
*DENSITY
2.3E-15,
**
*ELASTIC, TYPE=ISO
169000., 0.262
**
**
**
**
**
```

Material cards

Define material properties under the name "POLYSI". Density and elastic properties are defined. TYPE=ISO represents isotropic properties.

```
*BOUNDARY, OP=NEW
DOF, 1,, 0.
DOF, 2,, 0.
DOF, 6,, 0.
EDGE1, YSYMM
EDGE2, XSYMM
CENTER, ENCASTRE
**
```

BC cards

Define boundary conditions. In this case, all the nodes along the centre circle are constrained to zero displacements. To simulate plate elements, DOFs 1, 2 and 6 are constrained for all the nodes in the model. Symmetrical conditions are also applied.

```
*AMPLITUDE, NAME=SINE, DEFINITION=PERIODIC
1,12.566,0,0
0,10
**
**
**
**
**
**
```

Amplitude curve

Define an amplitude curve that can be a function of time or frequency. Loads or boundary conditions can then be made to follow the defined amplitude curve. In this case, a periodic function of the Fourier series is defined. The name of this amplitude curve is given as “SINE”.

```
*STEP, INC=1000
**
*DYNAMIC, DIRECT, NOHAF
0.1, 1.0
**
*NSET, NSET=DOF, GENERATE
1, 383, 1
*NSET, NSET=FORCE
1, 143, 377
**
**
** FORCE
**
*CLOAD, OP=NEW, AMPLITUDE=SINE
FORCE, 3, 1.
**
**
```

Load cards

“CLOAD” defines concentrated loading on the node set “FORCE” defined earlier. The load follows the amplitude curve, “SINE”, defined earlier.

Control cards

Indicate the analysis step. In this case it is a “DYNAMIC” analysis, which performs a direct integration step to determine the transient response. The parameters following the keyword, DYNAMIC specify various parameters for the algorithm. The first entry in the data line specifies the duration of each time step and the second specifies the total time step.

```
*NODE   PRINT, FREQ=1
U,
V,
A,
*NODE   FILE, FREQ=1
U,
V,
A,
**
*END STEP
```

Output control cards

Define the output required. In this case, the nodal displacement components (U), velocity components (V) and acceleration components (A) are requested.

8.6.3 Solution Process

The significance of the information provided in the above input file is very similar to the previous case study. Therefore, this section will highlight the differences that are mainly used for the transient analysis.

The definition of amplitude curve is important here as it enables the load (or boundary condition) to be defined as a function of time here. In this case the load will follow the sinusoidal function defined in the amplitude curve block. The sinusoidal function is defined as a periodic function whereby the formula used is actually the Fourier series. The data lines in the amplitude curve block basically define the angular frequency and the other constants in the Fourier series.

The control card specifies that the analysis is a direct integration, transient analysis. In ABAQUS, Newmarks's method (Section 3.7.2) together with the Hilber–Hughes–Taylor operator [1978] applied on the equilibrium equations is used as the implicit solver for direct integration analysis. The time increment is specified to be 0.1 s, and the total time of the step is 1.0 s. As mentioned in Chapter 3, implicit methods involve solving of the matrix equation at each individual increment in time, therefore the analysis can be rather computationally expensive. The algorithm used by ABAQUS is quite complex, involving the capabilities of having automatic deduction of the required time increments. Details are beyond the scope of this book.

8.6.4 Result and Discussion

Upon the analysis of the problem defined by the input file above, the displacement, velocity and acceleration components throughout each individual time increment can be obtained

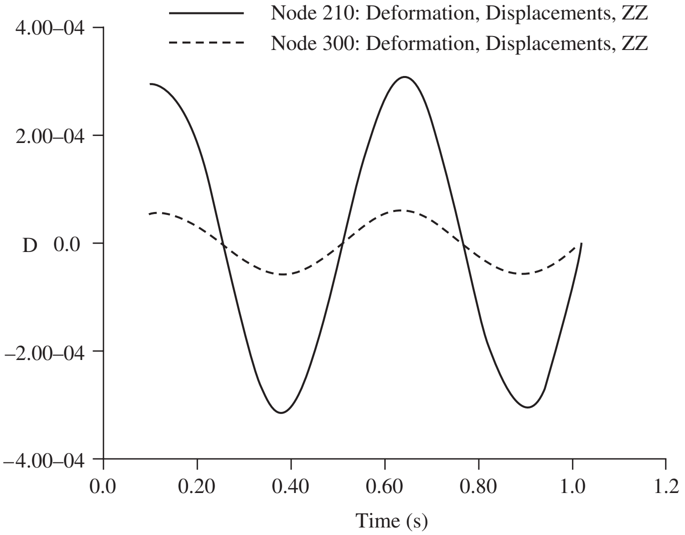


Figure 8.16. Displacement–time history at nodes 210 and 300.

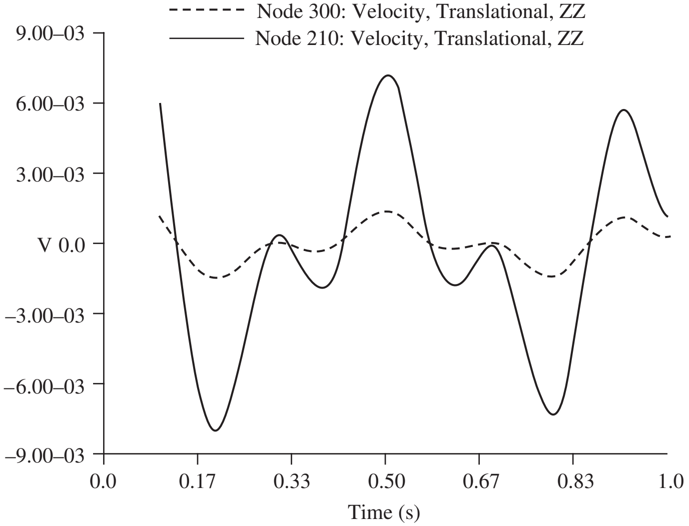


Figure 8.17. Velocity–time history at nodes 210 and 300.

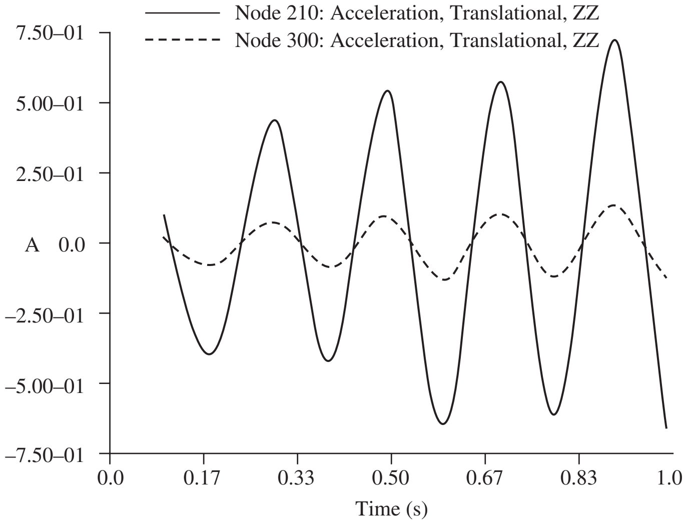


Figure 8.18. Acceleration–time history at nodes 210 and 300.

until the final time step specified. Therefore, we have what is known as the displacement–time history, the velocity–time history and the acceleration–time history, as shown in Figures 8.16, 8.17 and 8.18, respectively. The plots show the displacement, velocity and acceleration histories of nodes 210 and 300.

8.7 REVIEW QUESTIONS

1. If the plate were not homogenous but laminated, how would the finite element equation be different?
2. State the procedure to develop a triangular plate element.
3. How should one develop a four-node quadrilateral element? How should one develop an eight-node element with curved edges?
4. How many Gauss points are required to obtain the exact results for Eqs. (8.19) and (8.20)?

9

FEM FOR 3D SOLIDS

9.1 INTRODUCTION

A three-dimensional (3D) solid element can be considered to be the most general of all solid finite elements because all the field variables are dependent of x, y and z. An example of a 3D solid structure under loading is shown in Figure 9.1. As can be seen, the force vectors here can be in any arbitrary direction in space. A 3D solid can also have any arbitrary shape, material properties and boundary conditions in space. As such, there are altogether six possible stress components, three normal and three shear, that need to be taken into consideration. Typically, a 3D solid element can be a tetrahedron or hexahedron in shape with either flat or curved surfaces. Each node of the element will have three translational degrees of freedom. The element can thus deform in all three directions in space.

Since the 3D element is said to be the most general solid element, the truss, beam, plate, 2D solid and shell elements can all be considered to be special cases of the 3D element. So, why is there a need to develop all the other elements? Why not just use the 3D element to model everything? Theoretically, yes, the 3D element can actually be used to model all kinds of structurural components, including trusses, beams, plates, shells and so on. However, it can be very tedious in geometry creation and meshing. Furthermore, it is also most demanding on computer resources. Hence, the general rule of thumb is, that when a

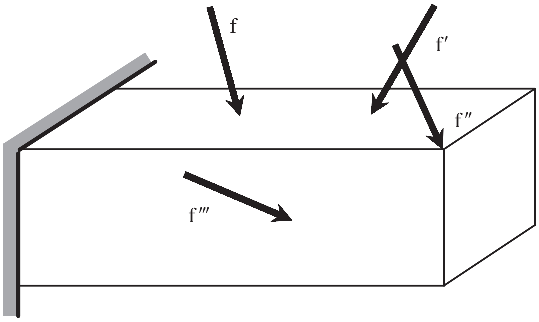


Figure 9.1. Example of a 3D solid under loadings.

structure can be assumed within acceptable tolerances to be simplified into a 1D (trusses, beams and frames) or 2D (2D solids and plates) structure, always do so. The creation of a 1D or 2D FEM model is much easier and efficient. Use 3D solid elements only when we have no other choices. The formulation of 3D solids elements is straightforward, because it is basically an extension of 2D solids elements. All the techniques used in 2D solids can be utilized, except that all the variables are now functions of x, y and z. The basic concepts, procedures and formulations for 3D solid elements can also be found in many existing books (see, e.g., Washizu, 1981; Rao, 1999; Zienkiewicz and Taylor, 2000; etc.).

9.2 TETRAHEDRON ELEMENT

9.2.1 Strain Matrix

Consider the same 3D solid structure as Figure 9.1, whose domain is divided in a proper manner into a number of *tetrahedron* elements (Figure 9.2) with four nodes and four surfaces, as shown in Figure 9.3. A tetrahedron element has four nodes, each having three DOFs

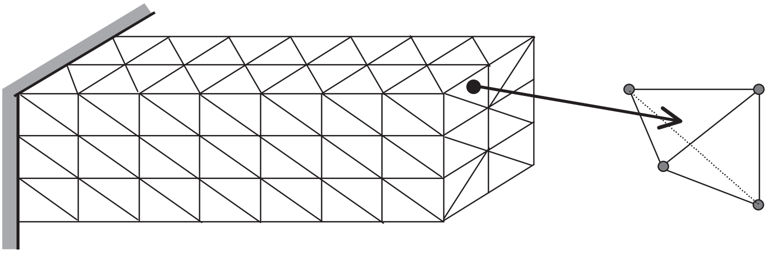


Figure 9.2. Solid block divided into four-node tetrahedron elements.

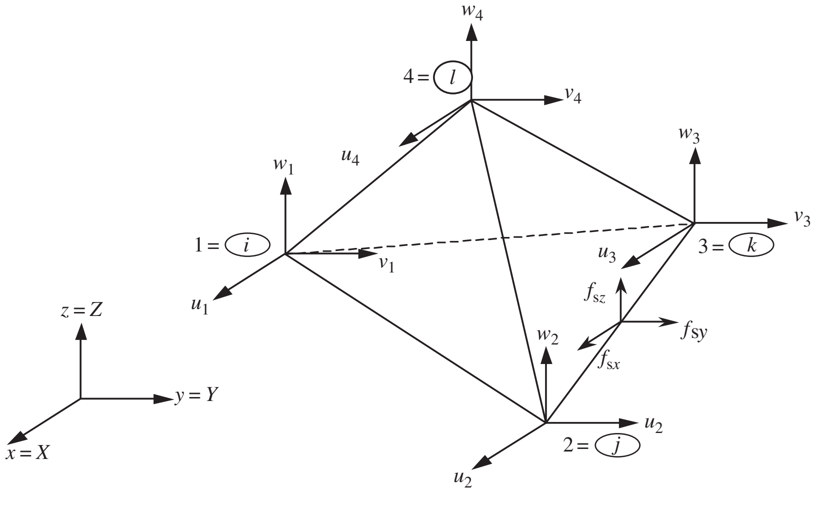


Figure 9.3. A tetrahedron element.

(u, v and w), making the total DOFs in a tetrahedron element twelve, as shown in Figure 9.3. The nodes are numbered 1, 2, 3 and 4 by the right-hand rule. The local Cartesian coordinate system for a tetrahedron element can usually be the same as the global coordinate system, as there are no advantages in having a separate local Cartesian coordinate system. In an element, the displacement vector $\mathbf{U}$ is a function of the coordinate x, y and z, and is interpolated by shape functions in the following form, which should by now be shown to be part and parcel of the finite element method:

$$\mathbf{U}^h(x, y, z) = \mathbf{N}(x, y, z)\mathbf{d}_e \tag{9.1}$$

where the nodal displacement vector, $\mathbf{d}_e$, is given as

$$\mathbf{d}_e = \left\{ \begin{matrix} u_1 \\ v_1 \\ w_1 \\ u_2 \\ v_2 \\ w_2 \\ u_3 \\ v_3 \\ w_3 \\ u_4 \\ v_4 \\ w_4 \end{matrix} \right\} \begin{matrix} \left.\vphantom{\begin{matrix} u_1 \\ v_1 \\ w_1 \end{matrix}}\right\} \text{displacements at node 1} \\ \left.\vphantom{\begin{matrix} u_2 \\ v_2 \\ w_2 \end{matrix}}\right\} \text{displacements at node 2} \\ \left.\vphantom{\begin{matrix} u_3 \\ v_3 \\ w_3 \end{matrix}}\right\} \text{displacements at node 3} \\ \left.\vphantom{\begin{matrix} u_4 \\ v_4 \\ w_4 \end{matrix}}\right\} \text{displacements at node 4} \end{matrix} \tag{9.2}$$

and the matrix of shape functions has the form

$$\mathbf{N} = \begin{array}{c} \begin{array}{cccc} \overbrace{\qquad\qquad}^{\text{node 1}} & \overbrace{\qquad\qquad}^{\text{node 2}} & \overbrace{\qquad\qquad}^{\text{node 3}} & \overbrace{\qquad\qquad}^{\text{node 4}} \end{array} \\ \begin{bmatrix} N_1 & 0 & 0 & N_2 & 0 & 0 & N_3 & 0 & 0 & N_4 & 0 & 0 \\ 0 & N_1 & 0 & 0 & N_2 & 0 & 0 & N_3 & 0 & 0 & N_4 & 0 \\ 0 & 0 & N_1 & 0 & 0 & N_2 & 0 & 0 & N_3 & 0 & 0 & N_4 \end{bmatrix} \end{array} \tag{9.3}$$

To develop the shape functions, we make use of what is known as the *volume coordinates*, which is a natural extension from the area coordinates for 2D solids. The use of the volume coordinates makes it more convenient for shape function construction and element matrix integration. The volume coordinates for node 1 is defined as

$$L_1 = \frac{V_{P234}}{V_{1234}} \tag{9.4}$$

where $V_{\mathrm{P}234}$ and V_{1234} denote, respectively, the volumes of the tetrahedrons P234 and 1234, as shown in Figure 9.4. The volume coordinate for node 2-4 can also be defined in the same

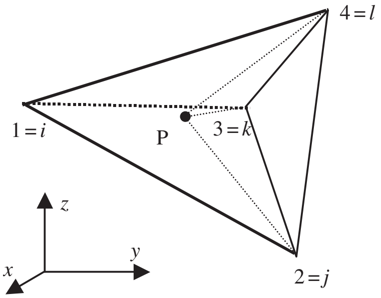


Figure 9.4. Volume coordinates for tetrahedron elements.

manner:

$$L_2 = \frac{V_{P134}}{V_{1234}}, \quad L_3 = \frac{V_{P124}}{V_{1234}}, \quad L_4 = \frac{V_{P123}}{V_{1234}} \tag{9.5}$$

The volume coordinate can also be viewed as the ratio between the distance of the point P and point 1 to the plane 234:

$$L_1 = \frac{d_{P-234}}{d_{1-234}}, \quad L_2 = \frac{d_{P-134}}{d_{1-234}}, \quad L_3 = \frac{d_{P-124}}{d_{1-234}}, \quad L_4 = \frac{d_{P-123}}{d_{1-234}} \tag{9.6}$$

It can easily be confirmed that

$$L_1 + L_2 + L_3 + L_4 = 1 \tag{9.7}$$

since

$$V_{P234} + V_{P134} + V_{P124} + V_{P123} = V_{1234} \tag{9.8}$$

It can also easily be confirmed that

$$L_i = \begin{cases} 1 & \text{at the home node } i \\ 0 & \text{at the remote nodes } jkl \end{cases} \tag{9.9}$$

Using Eq. (9.9), the relationship between the volume coordinates and Cartesian coordinates can be easily derived:

$$\begin{aligned} x &= L_1x_1 + L_2x_2 + L_3x_3 + L_4x_4 \\ y &= L_1y_1 + L_2y_2 + L_3y_3 + L_4y_4 \\ z &= L_1z_1 + L_2z_2 + L_3z_3 + L_4z_4 \end{aligned} \tag{9.10}$$

Equations (9.7) and (9.10) can then be expressed as a single matrix equation as follows:

$$\begin{Bmatrix} 1 \\ x \\ y \\ z \end{Bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \end{bmatrix} \begin{Bmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \end{Bmatrix} \tag{9.11}$$

The inversion of Eq. (9.11) will give

$$\begin{Bmatrix} L_1 \\ L_2 \\ L_3 \\ L_4 \end{Bmatrix} = \frac{1}{6V} \begin{bmatrix} a_1 & b_1 & c_1 & d_1 \\ a_2 & b_2 & c_2 & d_2 \\ a_3 & b_3 & c_3 & d_3 \\ a_4 & b_4 & c_4 & d_4 \end{bmatrix} \begin{Bmatrix} 1 \\ x \\ y \\ z \end{Bmatrix} \tag{9.12}$$

where

$$\begin{aligned} a_i &= \det \begin{bmatrix} x_j & y_j & z_j \\ x_k & y_k & z_k \\ x_l & y_l & z_l \end{bmatrix}, & b_i &= -\det \begin{bmatrix} 1 & y_j & z_j \\ 1 & y_k & z_k \\ 1 & y_l & z_l \end{bmatrix} \\ c_i &= -\det \begin{bmatrix} y_j & 1 & z_j \\ y_k & 1 & z_k \\ y_l & 1 & z_1 \end{bmatrix}, & d_i &= -\det \begin{bmatrix} y_j & z_j & 1 \\ y_k & z_k & 1 \\ y_l & z_l & 1 \end{bmatrix} \end{aligned} \tag{9.13}$$

in which the subscript i varies from 1 to 4, and j, k and l are determined by a cyclic permutation in the order of i, j, k, l. For example, if $i = 1$, then $j = 2$, $k = 3$, $l = 4$. When $i = 2$, then $j = 3$, $k = 4$, $l = 1$. The volume of the tetrahedron element V can be obtained by

$$V = \frac{1}{6} \times \det \begin{bmatrix} 1 & x_i & y_i & z_i \\ 1 & x_j & y_j & z_j \\ 1 & x_k & y_k & z_k \\ 1 & x_l & y_l & z_l \end{bmatrix} \tag{9.14}$$

The properties of L_i, as depicted in Eqs. (9.6) to (9.9), show that L_i can be used as the shape function of a four-nodal tetrahedron element:

$$N_i = L_i = \frac{1}{6V}(a_i + b_i x + c_i y + d_i z) \tag{9.15}$$

It can be seen from above that the shape function is a linear function of x, y and z, hence, the four-nodal tetrahedron element is a linear element. Note that from Eq. (9.14), the moment matrix of the linear basis functions will never be singular, unless the volume of the element is zero (or the four nodes of the element are in a plane). Based on Lemmas 2 and 3, we can be sure that the shape functions given by Eq. (9.15) satisfy the sufficient requirement of FEM shape functions.

It was mentioned that there are six stresses in a 3D element in total. The stress components are $\{\sigma_{xx} \quad \sigma_{yy} \quad \sigma_{zz} \quad \sigma_{yz} \quad \sigma_{xz} \quad \sigma_{xy}\}$. To get the corresponding strains, $\{\varepsilon_{xx} \quad \varepsilon_{yy} \quad \varepsilon_{zz} \quad \varepsilon_{yz} \quad \varepsilon_{xz} \quad \varepsilon_{xy}\}$, we can substitute Eq. (9.1) into Eq. (2.5):

$$\varepsilon = \mathbf{LU} = \mathbf{LNd}_e = \mathbf{Bd}_e \tag{9.16}$$

where the strain matrix **B** is given by

$$\mathbf{B} = \mathbf{LN} = \begin{bmatrix} \partial/\partial x & 0 & 0 \\ 0 & \partial/\partial y & 0 \\ 0 & 0 & \partial/\partial z \\ 0 & \partial/\partial z & \partial/\partial y \\ \partial/\partial z & 0 & \partial/\partial x \\ \partial/\partial y & \partial/\partial x & 0 \end{bmatrix} \mathbf{N} \tag{9.17}$$

Using Eq. (9.3), the strain matrix, **B**, can be obtained as

$$\mathbf{B} = \frac{1}{2V} \begin{bmatrix} b_1 & 0 & 0 & b_2 & 0 & 0 & b_3 & 0 & 0 & b_4 & 0 & 0 \\ 0 & c_1 & 0 & 0 & c_2 & 0 & 0 & c_3 & 0 & 0 & c_4 & 0 \\ 0 & 0 & d_1 & 0 & 0 & d_2 & 0 & 0 & d_3 & 0 & 0 & d_4 \\ c_1 & b_1 & 0 & c_2 & b_2 & 0 & c_3 & b_3 & 0 & c_4 & b_4 & 0 \\ 0 & d_1 & c_1 & 0 & d_2 & c_2 & 0 & d_3 & c_3 & 0 & d_4 & c_4 \\ d_1 & 0 & b_1 & d_2 & 0 & b_2 & d_3 & 0 & b_3 & d_4 & 0 & b_4 \end{bmatrix} \tag{9.18}$$

It can be seen that the strain matrix for a linear tetrahedron element is a constant matrix. This implies that the strain within a linear tetrahedron element is constant, and thus so is the stress. Therefore, the linear tetrahedron elements are also often referred to as a *constant strain element* or *constant stress element*, similar to the case of 2D linear triangular elements.

9.2.2 Element Matrices

Once the strain matrix has been obtained, the stiffness matrix $\mathbf{k}_e$ for 3D solid elements can be obtained by substituting Eq. (9.18) into Eq. (3.71). Since the strain is constant, the element strain matrix is obtained as

$$\mathbf{k}_e = \int_{V_e} \mathbf{B}^T \mathbf{cB} \, \mathrm{d}V = V_e \mathbf{B}^T \mathbf{cB} \tag{9.19}$$

Note that the material constant matrix **c** is given generally by Eq. (2.9).

The mass matrix can similarly be obtained using Eq. (3.75):

$$\mathbf{m}_e = \int_{V_e} \rho \mathbf{N}^T \mathbf{N} \, \mathrm{d}V = \int_{V_e} \rho \begin{bmatrix} \mathbf{N}_{11} & \mathbf{N}_{12} & \mathbf{N}_{13} & \mathbf{N}_{14} \\ \mathbf{N}_{21} & \mathbf{N}_{22} & \mathbf{N}_{23} & \mathbf{N}_{24} \\ \mathbf{N}_{31} & \mathbf{N}_{32} & \mathbf{N}_{33} & \mathbf{N}_{34} \\ \mathbf{N}_{41} & \mathbf{N}_{42} & \mathbf{N}_{43} & \mathbf{N}_{44} \end{bmatrix} \mathrm{d}V \tag{9.20}$$

where

$$\mathbf{N}_{ij} = \begin{bmatrix} N_i N_j & 0 & 0 \\ 0 & N_i N_j & 0 \\ 0 & 0 & N_i N_j \end{bmatrix} \tag{9.21}$$

Using the following formula [Eisenberg and Malvern, 1973],

$$\int_{V_e} L_1^m L_2^n L_3^p L_4^q \, \mathrm{d}V = \frac{m!n!p!q!}{(m+n+p+q+3)!} 6V_e \tag{9.22}$$

we can conveniently evaluate the integral in Eq. (9.20) to give

$$
\mathbf{m}_e = \frac{\rho V_e}{20}\begin{bmatrix}
2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
 & 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
 & & 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
 & & & 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
 & & & & 2 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
 & & & & & 2 & 0 & 0 & 1 & 0 & 0 & 1 \\
 & & & & & & 2 & 0 & 0 & 1 & 0 & 0 \\
 & & & & & & & 2 & 0 & 0 & 1 & 0 \\
 & & & & & & & & 2 & 0 & 0 & 1 \\
 & & sy. & & & & & & & 2 & 0 & 0 \\
 & & & & & & & & & & 2 & 0 \\
 & & & & & & & & & & & 2
\end{bmatrix} \tag{9.23}
$$

An alternative way to calculate the mass matrix for 3D solid elements is to use a special natural coordinate system, which is defined as shown in Figures 9.5–9.7. In Figure 9.5, the plane of ξ = constant is defined in such a way that the edge P–Q stays parallel to the edge 2–3 of the element, and point 4 coincides with point 4 of the element. When P moves to point 1, $\xi = 0$, and when P moves to point 2, $\xi = 1$. In Figure 9.6, the plane of η = constant is defined in such a way that the edge 1–4 on the triangle coincides with the edge 1–4 of the element, and point P stays on the edge 2–3 of the element. When P moves to point 2, $\eta = 0$, and when P moves to point 3, $\eta = 1$. The plane of ζ = constant is defined in Figure 9.7, in such a way that the plane P–Q–R stays parallel to the plane 1–2–3 of the element, and when P moves to point 4, $\zeta = 0$, and when P moves to point 2, $\zeta = 1$. In addition, the

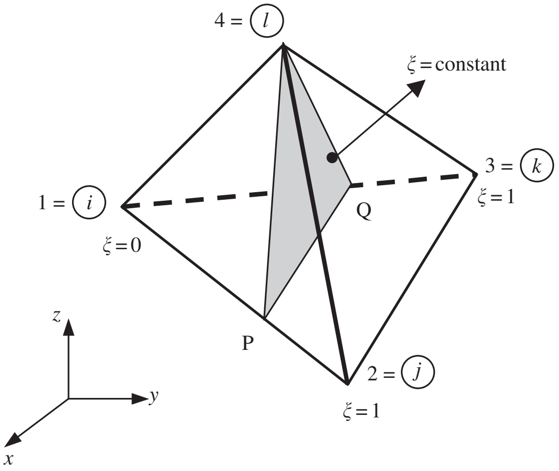


Figure 9.5. Natural coordinate, where ξ = constant.

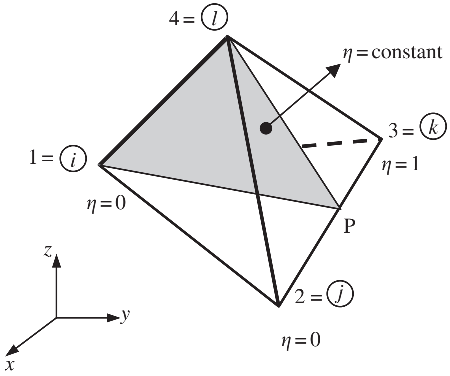


Figure 9.6. Natural coordinate, where η = constant.

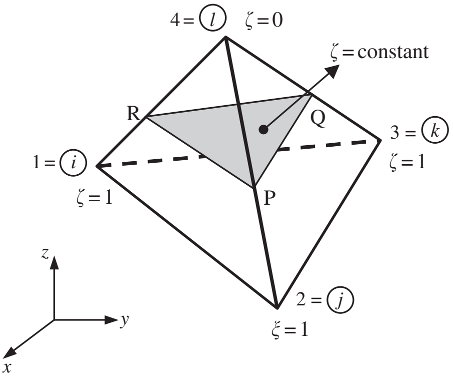


Figure 9.7. Natural coordinate, where ζ = constant.

plane 1–2–3 on the element sits on the x–y plane. Therefore, the relationship between xyz and $\xi\eta\zeta$ can be obtained in the following steps:

In Figure 9.8, the coordinates at point P are first interpolated using the x, y and z coordinates at points 2 and 3:

$$\begin{aligned} x_P &= \eta(x_3 - x_2) + x_2 \\ y_P &= \eta(y_3 - y_2) + y_2 \\ z_P &= 0 \end{aligned} \tag{9.24}$$

The coordinates at point B are then interpolated using the x, y and z coordinates at points 1 and P:

$$\begin{aligned} x_B &= \xi(x_P - x_1) + x_1 = \xi\eta(x_3 - x_2) + \xi(x_2 - x_1) + x_1 \\ y_B &= \xi(y_P - y_1) + y_1 = \xi\eta(y_3 - y_2) + \xi(y_2 - y_1) + y_1 \\ z_B &= 0 \end{aligned} \tag{9.25}$$

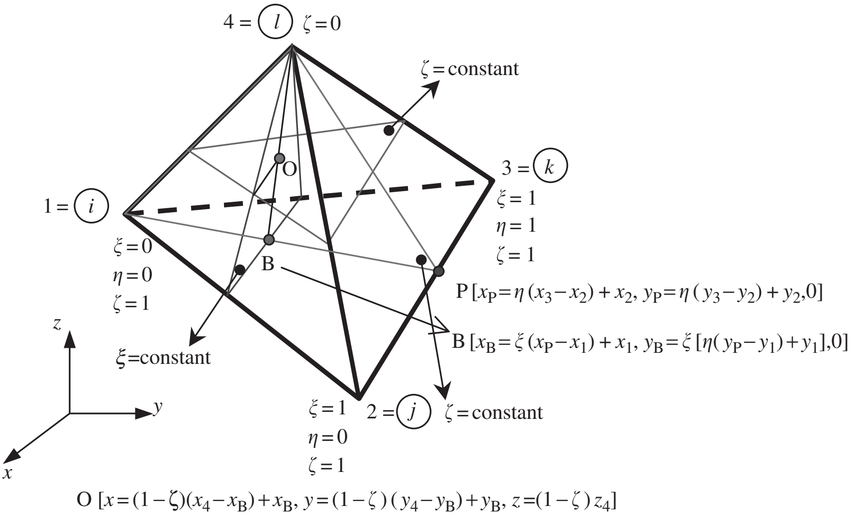

$4 = l$ $\zeta = 0$
$\zeta = \text{constant}$
$3 = k$
$\xi = 1$
$\eta = 1$
$\zeta = 1$
$1 = i$
O
$\xi = 0$
$\eta = 0$
$\zeta = 1$
B
P $[x_P = \eta(x_3 - x_2) + x_2, y_P = \eta(y_3 - y_2) + y_2, 0]$
B $[x_B = \xi(x_P - x_1) + x_1, y_B = \xi[\eta(y_P - y_1) + y_1], 0]$
z
$\xi = \text{constant}$
y
x
$\xi = 1$
$\eta = 0$
$\zeta = 1$
$2 = j$
$\zeta = \text{constant}$
O $[x = (1-\zeta)(x_4 - x_B) + x_B, y = (1-\zeta)(y_4 - y_B) + y_B, z = (1-\zeta)z_4]$

Figure 9.8. Cartesian coordinates *xyz* of point O in term of $\xi\eta\zeta$.

The coordinates at point O are finally interpolated using the x, y and z coordinates at points 4 and B:

$$\begin{aligned}
x &= x_4 - \zeta(x_4 - x_B) = x_4 - \zeta(x_4 - x_1) + \xi\zeta(x_2 - x_1) - \xi\zeta(x_2 - x_3) \\
y &= y_4 - \zeta(y_4 - y_B) = y_4 - \zeta(y_4 - y_1) + \xi\zeta(y_2 - y_1) - \xi\zeta(y_2 - y_3) \\
z &= (1 - \zeta)z_4
\end{aligned} \tag{9.26}$$

With this special natural coordinate system, the shape functions in the matrix of Eq. (9.3) can be written by inspection as

$$\begin{aligned}
N_1 &= (1 - \xi)\zeta \\
N_2 &= \xi\eta\zeta \\
N_3 &= \xi\zeta(1 - \eta) \\
N_4 &= (1 - \zeta)
\end{aligned} \tag{9.27}$$

The Jacobian matrix between xyz and $\xi\eta\zeta$ is required, and is given as

$$\mathbf{J} = \begin{bmatrix} \partial x/\partial\xi & \partial x/\partial\eta & \partial x/\partial\zeta \\ \partial y/\partial\xi & \partial y/\partial\eta & \partial y/\partial\zeta \\ \partial z/\partial\xi & \partial z/\partial\eta & \partial z/\partial\zeta \end{bmatrix} \tag{9.28}$$

Using Eqs. (9.26) and (9.27), the determinate of the Jacobian can be found to be

$$\det[\mathbf{J}] = \begin{vmatrix} \zeta x_{21} + \eta\zeta x_{31} & \xi\zeta x_{31} & -x_{41} + \xi x_{21} + \xi\eta x_{31} \\ \zeta y_{21} + \eta\zeta y_{31} & \xi\zeta y_{31} & -y_{41} + \xi y_{21} + \xi\eta y_{31} \\ 0 & z_4 & 0 \end{vmatrix} = -6V\xi\zeta^2 \tag{9.29}$$

The mass matrix can now be obtained as

$$\mathbf{m}_e = \int_{V_e} \rho \mathbf{N}^T \mathbf{N} \, \mathrm{d}V = \int_0^1 \int_0^1 \int_0^1 \rho \mathbf{N}^T \mathbf{N} \det[\mathbf{J}] \, \mathrm{d}\xi \, \mathrm{d}\eta \, \mathrm{d}\zeta \tag{9.30}$$

which gives

$$\mathbf{m}_e = -6V_e\rho \int_0^1 \int_0^1 \int_0^1 \xi\zeta^2 \begin{bmatrix} \mathbf{N}_{11} & \mathbf{N}_{12} & \mathbf{N}_{13} & \mathbf{N}_{14} \\ \mathbf{N}_{21} & \mathbf{N}_{22} & \mathbf{N}_{23} & \mathbf{N}_{24} \\ \mathbf{N}_{31} & \mathbf{N}_{32} & \mathbf{N}_{33} & \mathbf{N}_{34} \\ \mathbf{N}_{41} & \mathbf{N}_{42} & \mathbf{N}_{43} & \mathbf{N}_{44} \end{bmatrix} \mathrm{d}\xi \, \mathrm{d}\eta \, \mathrm{d}\zeta \tag{9.31}$$

where $\mathbf{N}_{ij}$ is given by Eq. (9.21), but in which the shape functions should be defined by Eq. (9.27). Evaluating the integrals in Eq. (9.31) would give the same mass matrix as in Eq. (9.23).

The nodal force vector for 3D solid elements can be obtained using Eqs. (3.78), (3.79) and (3.81). Suppose the element is loaded by a distributed force $\mathbf{f}_s$ on the edge 2–3 of the element as shown in Figure 9.3; the nodal force vector becomes

$$\mathbf{f}_e = \int_l [\mathbf{N}]^{\mathrm{T}} \Big|_{3\text{–}4} \begin{Bmatrix} f_{sx} \\ f_{sy} \\ f_{sz} \end{Bmatrix} \mathrm{d}l \tag{9.32}$$

If the load is uniformly distributed, f_{sx}, f_{sx} and f_{sz} are constants, and the above equation becomes

$$\mathbf{f}_e = \frac{1}{2} l_{3\text{–}4} \begin{Bmatrix} \{\mathbf{0}\}_{3\times 1} \\ \{\mathbf{0}\}_{3\times 1} \\ \begin{Bmatrix} f_{sx} \\ f_{sy} \\ f_{sz} \end{Bmatrix} \\ \begin{Bmatrix} f_{sx} \\ f_{sy} \\ f_{sz} \end{Bmatrix} \\ \{\mathbf{0}\}_{3\times 1} \\ \{\mathbf{0}\}_{3\times 1} \\ \{\mathbf{0}\}_{3\times 1} \\ \{\mathbf{0}\}_{3\times 1} \end{Bmatrix} \tag{9.33}$$

where $l_{3\text{–}4}$ is the length of the edge 3–4. Equation (9.33) implies that the distributed forces are equally divided and applied at the two nodes. This conclusion also applies to evenly distributed surface forces applied on any face of the element, and to evenly distributed body force applied on the entire body of the element. Finally, the stiffness matrix, $\mathbf{k}_e$, the mass matrix, $\mathbf{m}_e$, and the nodal force vector, $\mathbf{f}_e$, can be used directly to assemble the global FE equation, Eq. (3.96), without going through a coordinate transformation.

9.3 HEXAHEDRON ELEMENT

9.3.1 Strain Matrix

Consider now a 3D domain, which is divided in a proper manner into a number of *hexahedron elements* with eight nodes and six surfaces, as shown in Figure 9.9. Each hexahedron element has nodes numbered 1, 2, 3, 4 and 5, 6, 7, 8 in a counter-clockwise manner, as shown in Figure 9.10.

As there are three DOFs at one node, there is a total of 24 DOFs in a hexahedron element. It is again useful to define a *natural coordinate system* (ξ, η, ζ) with its origin at the centre of the transformed cube, as this makes it easier to construct the shape functions and to evaluate the matrix integration. The coordinate mapping is preformed in a similar manner as for quadrilateral elements in Chapter 7. Like the quadrilateral element, shape functions are also used to interpolate the coordinates from the nodal

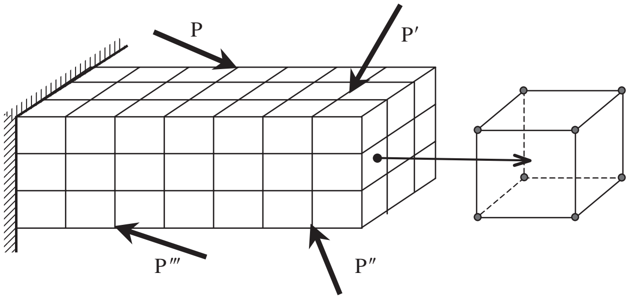


Figure 9.9. Solid block divided into eight-nodal hexahedron elements.

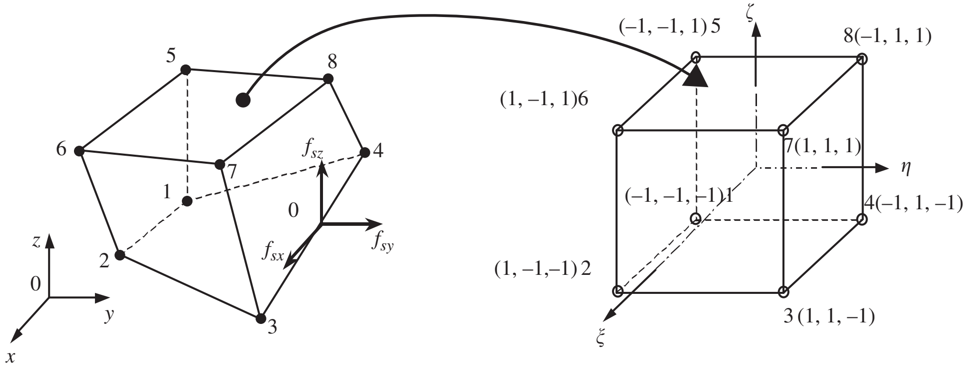


Figure 9.10. An eight-nodal hexahedron element and the coordinate systems.

coordinates:

$$
\begin{aligned}
x &= \sum_{i=1}^{8} N_i(\xi, \eta, \zeta) x_i \\
y &= \sum_{i=1}^{8} N_i(\xi, \eta, \zeta) y_i \\
z &= \sum_{i=1}^{8} N_i(\xi, \eta, \zeta) z_i
\end{aligned}
\tag{9.34}
$$

The shape functions are given in the local natural coordinate system as

$$
\begin{aligned}
N_1 &= \tfrac{1}{8}(1-\xi)(1-\eta)(1-\zeta) \\
N_2 &= \tfrac{1}{8}(1+\xi)(1-\eta)(1-\zeta) \\
N_3 &= \tfrac{1}{8}(1+\xi)(1+\eta)(1-\zeta) \\
N_4 &= \tfrac{1}{8}(1-\xi)(1+\eta)(1-\zeta) \\
N_5 &= \tfrac{1}{8}(1-\xi)(1-\eta)(1+\zeta) \\
N_6 &= \tfrac{1}{8}(1+\xi)(1-\eta)(1+\zeta) \\
N_7 &= \tfrac{1}{8}(1+\xi)(1+\eta)(1+\zeta) \\
N_8 &= \tfrac{1}{8}(1-\xi)(1+\eta)(1+\zeta)
\end{aligned}
\tag{9.35}
$$

or in a concise form,

$$
N_i = \tfrac{1}{8}(1+\xi\xi_i)(1+\eta\eta_i)(1+\zeta\zeta_i) \tag{9.36}
$$

where (ξ_i, η_i, ζ_i) denotes the natural coordinates of node I.

From Eq. (9.36), it can be seen that the shape functions vary linearly in the ξ, η and ζ directions. Therefore, these shape functions are sometimes called tri-linear functions. The shape function N_i is a three-dimensional analogy of that given in Eq. (7.54). It is very easy to directly observe that the tri-linear elements possess the delta function property. In addition, since all these shape functions can be formed using the common set of eight basis functions of

$$
1, \xi, \eta, \varsigma, \xi\eta, \xi\varsigma, \eta\varsigma, \xi\eta\varsigma \tag{9.37}
$$

which contain both constant and linear basis functions. Therefore, these shape functions can expect to possess both partitions of the unity property as well as the linear reproduction property (see Lemmas 2 and 3 in Chapter 3).

In a hexahedron element, the displacement vector $\mathbf{U}$ is a function of the coordinates x, y and z, and as before, it is interpolated using the shape functions

$$\mathbf{U} = \mathbf{N}\mathbf{d}_e \tag{9.38}$$

where the nodal displacement vector, $\mathbf{d}_e$ is given by

$$\mathbf{d}_e = \left\{ \begin{array}{c} \mathbf{d}_{e1} \\ \mathbf{d}_{e2} \\ \mathbf{d}_{e3} \\ \mathbf{d}_{e4} \\ \mathbf{d}_{e5} \\ \mathbf{d}_{e6} \\ \mathbf{d}_{e7} \\ \mathbf{d}_{e8} \end{array} \right\} \begin{array}{l} \text{displacement components at node 1} \\ \text{displacement components at node 2} \\ \text{displacement components at node 3} \\ \text{displacement components at node 4} \\ \text{displacement components at node 5} \\ \text{displacement components at node 6} \\ \text{displacement components at node 7} \\ \text{displacement components at node 8} \end{array} \tag{9.39}$$

in which

$$\mathbf{d}_{ei} = \left\{ \begin{array}{c} u_1 \\ v_1 \\ w_1 \end{array} \right\} \quad (i = 1, 2, \ldots, 8) \tag{9.40}$$

is the displacement at node i. The matrix of shape functions is given by

$$\mathbf{N} = \begin{bmatrix} \mathbf{N}_1 & \mathbf{N}_2 & \mathbf{N}_3 & \mathbf{N}_4 & \mathbf{N}_5 & \mathbf{N}_6 & \mathbf{N}_7 & \mathbf{N}_8 \end{bmatrix} \tag{9.41}$$

in which each sub-matrix, $\mathbf{N}_i$, is given as

$$\mathbf{N}_i = \begin{bmatrix} N_i & 0 & 0 \\ 0 & N_i & 0 \\ 0 & 0 & N_i \end{bmatrix} \quad (i = 1, 2, \ldots, 8) \tag{9.42}$$

In this case, the strain matrix defined by Eq. (9.17) can be expressed as

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{B}_2 & \mathbf{B}_3 & \mathbf{B}_4 & \mathbf{B}_5 & \mathbf{B}_6 & \mathbf{B}_7 & \mathbf{B}_8 \end{bmatrix} \tag{9.43}$$

whereby

$$\mathbf{B}_i = \mathbf{L}\mathbf{N}_i = \begin{bmatrix} \partial N_i/\partial x & 0 & 0 \\ 0 & \partial N_i/\partial y & 0 \\ 0 & 0 & \partial N_i/\partial z \\ 0 & \partial N_i/\partial z & \partial N_i/\partial y \\ \partial N_i/\partial z & 0 & \partial N_i/\partial x \\ \partial N_i/\partial y & \partial N_i/\partial x & 0 \end{bmatrix} \tag{9.44}$$

As the shape functions are defined in terms of the natural coordinates, ξ, η and ζ, to obtain the derivatives with respect to x, y and z in the strain matrix, the chain rule of partial differentiation needs to be used:

$$\begin{aligned}\frac{\partial N_i}{\partial \xi} &= \frac{\partial N_i}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial N_i}{\partial y}\frac{\partial y}{\partial \xi} + \frac{\partial N_i}{\partial z}\frac{\partial z}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} &= \frac{\partial N_i}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial N_i}{\partial y}\frac{\partial y}{\partial \eta} + \frac{\partial N_i}{\partial z}\frac{\partial z}{\partial \eta} \\ \frac{\partial N_i}{\partial \zeta} &= \frac{\partial N_i}{\partial x}\frac{\partial x}{\partial \zeta} + \frac{\partial N_i}{\partial y}\frac{\partial y}{\partial \zeta} + \frac{\partial N_i}{\partial z}\frac{\partial z}{\partial \zeta}\end{aligned} \tag{9.45}$$

which can be expressed in the matrix form

$$\begin{Bmatrix} \partial N_i/\partial \xi \\ \partial N_i/\partial \eta \\ \partial N_i/\partial \zeta \end{Bmatrix} = \mathbf{J} \begin{Bmatrix} \partial N_i/\partial x \\ \partial N_i/\partial y \\ \partial N_i/\partial z \end{Bmatrix} \tag{9.46}$$

where **J** is the *Jacobian matrix* defined by

$$\mathbf{J} = \begin{bmatrix} \partial x/\partial \xi & \partial y/\partial \xi & \partial z/\partial \xi \\ \partial x/\partial \eta & \partial y/\partial \eta & \partial z/\partial \eta \\ \partial x/\partial \zeta & \partial y/\partial \zeta & \partial z/\partial \zeta \end{bmatrix} \tag{9.47}$$

Recall that the coordinates, x, y and z are interpolated by the shape functions from the nodal coordinates. Hence, substitute the interpolation of the coordinates, Eq. (9.34), into Eq. (9.47), which gives

$$\mathbf{J} = \begin{bmatrix} \frac{\partial N_1}{\partial \xi} & \frac{\partial N_2}{\partial \xi} & \frac{\partial N_3}{\partial \xi} & \frac{\partial N_4}{\partial \xi} & \frac{\partial N_5}{\partial \xi} & \frac{\partial N_6}{\partial \xi} & \frac{\partial N_7}{\partial \xi} & \frac{\partial N_8}{\partial \xi} \\ \frac{\partial N_1}{\partial \eta} & \frac{\partial N_2}{\partial \eta} & \frac{\partial N_3}{\partial \eta} & \frac{\partial N_4}{\partial \eta} & \frac{\partial N_5}{\partial \eta} & \frac{\partial N_6}{\partial \eta} & \frac{\partial N_7}{\partial \eta} & \frac{\partial N_8}{\partial \eta} \\ \frac{\partial N_1}{\partial \zeta} & \frac{\partial N_2}{\partial \zeta} & \frac{\partial N_3}{\partial \zeta} & \frac{\partial N_4}{\partial \zeta} & \frac{\partial N_5}{\partial \zeta} & \frac{\partial N_6}{\partial \zeta} & \frac{\partial N_7}{\partial \zeta} & \frac{\partial N_8}{\partial \zeta} \end{bmatrix} \begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_4 & y_4 & z_4 \\ x_5 & y_5 & z_5 \\ x_6 & y_6 & z_6 \\ x_7 & y_7 & z_7 \\ x_8 & y_8 & z_8 \end{bmatrix} \tag{9.48}$$

or

$$\mathbf{J} = \begin{bmatrix} \sum_{i=1}^{8} x_i \partial N_i/\partial \xi & \sum_{i=1}^{8} y_i \partial N_i/\partial \xi & \sum_{i=1}^{8} z_i \partial N_i/\partial \xi \\ \sum_{i=1}^{8} x_i \partial N_i/\partial \eta & \sum_{i=1}^{8} y_i \partial N_i/\partial \eta & \sum_{i=1}^{8} z_i \partial N_i/\partial \eta \\ \sum_{i=1}^{8} x_i \partial N_i/\partial \zeta & \sum_{i=1}^{8} y_i \partial N_i/\partial \zeta & \sum_{i=1}^{8} z_i \partial N_i/\partial \zeta \end{bmatrix} \tag{9.49}$$

Equation (9.46) can be re-written as

$$\begin{Bmatrix} \partial N_i/\partial x \\ \partial N_i/\partial y \\ \partial N_i/\partial z \end{Bmatrix} = \mathbf{J}^{-1} \begin{Bmatrix} \partial N_i/\partial \xi \\ \partial N_i/\partial \eta \\ \partial N_i/\partial \zeta \end{Bmatrix} \tag{9.50}$$

which is then used to compute the strain matrix, **B**, in Eqs. (9.43) and (9.44), by replacing all the derivatives of the shape functions with respect to x, y and z to those with respect to ξ, η and ζ.

9.3.2 Element Matrices

Once the strain matrix, **B**, has been computed, the stiffness matrix, $\mathbf{k}_e$, for 3D solid elements can be obtained by substituting **B** into Eq. (3.71):

$$\mathbf{k}_e = \int_{V_e} \mathbf{B}^{\mathrm{T}}\mathbf{c}\mathbf{B}\,\mathrm{d}A = \int_{-1}^{+1}\int_{-1}^{+1}\int_{-1}^{+1} \mathbf{B}^{\mathrm{T}}\mathbf{c}\mathbf{B}\det[\mathbf{J}]\,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta \tag{9.51}$$

Note that the matrix of material constant, **c**, is given by Eq. (2.9). As the strain matrix, **B**, is a function of ξ, η and ζ, evaluating the integrations in Eq. (9.51) can be very difficult. Therefore, the integrals are performed using a numerical integration scheme. The Gauss integration scheme discussed in Section 7.3.4 is often used to carry out the integral. For three-dimensional integrations, the Gauss integration is sampled in three directions, as follows:

$$I = \int_{-1}^{+1}\int_{-1}^{+1}\int_{-1}^{+1} f(\xi,\eta)\,\mathrm{d}\xi\,\mathrm{d}\eta = \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l} w_i w_j w_k f(\xi_i,\eta_j,\zeta_j) \tag{9.52}$$

To obtain the mass (inertia) matrix for the hexahedron element, substitute the shape function matrix, Eq. (9.41), into Eq. (3.75):

$$\mathbf{m}_e = \int_{V_e} \rho\mathbf{N}^T\mathbf{N}\,\mathrm{d}V = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} \rho\mathbf{N}^T\mathbf{N}\det[\mathbf{J}]\,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta \tag{9.53}$$

The above integral is also usually carried out using Gauss integration. If the hexahedron is rectangular with dimensions of $a \times b \times c$, the determinate of the Jacobian matrix is simply given by

$$\det[\mathbf{J}] = abc = V_e \tag{9.54}$$

and the mass matrix can be explicitly obtained as

$$\mathbf{m}_e = \begin{bmatrix} \mathbf{m}_{11} & \mathbf{m}_{12} & \mathbf{m}_{13} & \mathbf{m}_{14} & \mathbf{m}_{15} & \mathbf{m}_{16} & \mathbf{m}_{17} & \mathbf{m}_{18} \\ & \mathbf{m}_{22} & \mathbf{m}_{23} & \mathbf{m}_{24} & \mathbf{m}_{25} & \mathbf{m}_{26} & \mathbf{m}_{27} & \mathbf{m}_{28} \\ & & \mathbf{m}_{33} & \mathbf{m}_{34} & \mathbf{m}_{35} & \mathbf{m}_{36} & \mathbf{m}_{37} & \mathbf{m}_{38} \\ & & & \mathbf{m}_{44} & \mathbf{m}_{45} & \mathbf{m}_{46} & \mathbf{m}_{47} & \mathbf{m}_{48} \\ & & & & \mathbf{m}_{55} & \mathbf{m}_{56} & \mathbf{m}_{57} & \mathbf{m}_{58} \\ & & & & & \mathbf{m}_{66} & \mathbf{m}_{67} & \mathbf{m}_{68} \\ & sy. & & & & & \mathbf{m}_{77} & \mathbf{m}_{78} \\ & & & & & & & \mathbf{m}_{88} \end{bmatrix} \tag{9.55}$$

where

$$
\begin{aligned}
\mathbf{m}_{ij} &= \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} \rho abc \mathbf{N}_i \mathbf{N}_j \,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta \\
&= \rho abc \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} \begin{bmatrix} N_i & 0 & 0 \\ 0 & N_i & 0 \\ 0 & 0 & N_i \end{bmatrix} \begin{bmatrix} N_j & 0 & 0 \\ 0 & N_j & 0 \\ 0 & 0 & N_j \end{bmatrix} \mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta \\
&= \rho abc \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} \begin{bmatrix} N_iN_j & 0 & 0 \\ 0 & N_iN_j & 0 \\ 0 & 0 & N_iN_j \end{bmatrix} \mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta
\end{aligned} \tag{9.56}
$$

or

$$
\mathbf{m}_{ij} = \begin{bmatrix} m_{ij} & 0 & 0 \\ 0 & m_{ij} & 0 \\ 0 & 0 & m_{ij} \end{bmatrix} \tag{9.57}
$$

in which

$$
\begin{aligned}
m_{ij} &= \rho abc \int_{-1}^{+1}\int_{-1}^{+1} N_iN_j \,\mathrm{d}\xi\,\mathrm{d}\eta\,\mathrm{d}\zeta \\
&= \frac{\rho abc}{64}\int_{-1}^{+1}(1+\xi_i\xi)(1+\xi_j\xi)\,\mathrm{d}\xi \int_{-1}^{+1}(1+\eta_i\eta)(1+\eta_j\eta)\,\mathrm{d}\eta \\
&\quad \times \int_{-1}^{+1}(1+\zeta_i\zeta)(1+\zeta_j\zeta)\,\mathrm{d}\zeta \\
&= \frac{\rho hab}{8}\left(1+\tfrac{1}{3}\xi_i\xi_j\right)\left(1+\tfrac{1}{3}\eta_i\eta_j\right)\left(1+\tfrac{1}{3}\zeta_i\zeta_j\right)
\end{aligned} \tag{9.58}
$$

As an example, m_{33} is calculated as follows:

$$
m_{33} = \frac{\rho abc}{8}\left(1+\tfrac{1}{3}\times 1\times 1\right)\left(1+\tfrac{1}{3}\times 1\times 1\right)\left(1+\tfrac{1}{3}\times 1\times 1\right) = 8\times\frac{\rho abc}{216} \tag{9.59}
$$

The other components of the mass matrix for a rectangular hexahedron element are:

$$
\begin{aligned}
& m_{11} = m_{22} = m_{33} = m_{44} = m_{55} = m_{66} = m_{77} = m_{88} = \frac{8\rho abc}{216} \\
& m_{12} = m_{23} = m_{34} = m_{56} = m_{67} = m_{78} = m_{14} = m_{58} = m_{15} = m_{26} = m_{37} \\
& \qquad = m_{48} = \frac{4\rho abc}{216} \\
& m_{13} = m_{24} = m_{16} = m_{25} = m_{36} = m_{47} = m_{57} = m_{68} = m_{27} = m_{38} = m_{45} \\
& \qquad = m_{18} = \frac{2\rho abc}{216} \\
& m_{17} = m_{28} = m_{35} = m_{46} = \frac{1\rho abc}{216}
\end{aligned} \tag{9.60}
$$

Note that the equalities in the above equation can be easily figured out by observing the relative geometric positions of the nodes in the cube element. For example, the relative geometric positions of nodes 1–2 is equivalent to the relative geometric positions of nodes 2–3, and the relative geometric positions of nodes 1–7 is equivalent to the relative geometric positions of nodes 2–8. If we write the portion of the mass matrix corresponding to only one translational direction, say the x direction, we have

$$\mathbf{m}_e = \frac{\rho abc}{216}\begin{bmatrix} 8 & 4 & 2 & 4 & 4 & 2 & 1 & 2 \\ & 8 & 4 & 2 & 2 & 4 & 2 & 1 \\ & & 8 & 4 & 1 & 2 & 4 & 2 \\ & & & 8 & 2 & 1 & 2 & 4 \\ & & & & 8 & 4 & 2 & 4 \\ & & & & & 8 & 4 & 2 \\ & sy. & & & & & 8 & 4 \\ & & & & & & & 8 \end{bmatrix} \tag{9.61}$$

The mass matrices corresponding to only the y and z directions are exactly the same as $\mathbf{m}_e$.

The nodal force vector for a rectangular hexahedron element can be obtained using Eqs. (3.78), (3.79) and (3.81). Suppose the element is loaded by a distributed force $\mathbf{f}_s$ on edge 3–4 of the element, as shown in Figure 9.10; the nodal force vector becomes

$$\mathbf{f}_e = \int_l [\mathbf{N}]^{\mathrm{T}}\Big|_{3\text{–}4} \begin{Bmatrix} f_{sx} \\ f_{sy} \\ f_{sz} \end{Bmatrix} \mathrm{d}l \tag{9.62}$$

If the load is uniformly distributed, f_{sx}, f_{sx} and f_{sz} are constants, and the above equation becomes

$$\mathbf{f}_e = \frac{1}{2} l_{3\text{–}4} \begin{Bmatrix} \{\mathbf{0}\}_{3\times1} \\ \{\mathbf{0}\}_{3\times1} \\ \begin{Bmatrix} f_{sx} \\ f_{sy} \\ f_{sz} \end{Bmatrix} \\ \begin{Bmatrix} f_{sx} \\ f_{sy} \\ f_{sz} \end{Bmatrix} \\ \{\mathbf{0}\}_{3\times1} \\ \{\mathbf{0}\}_{3\times1} \\ \{\mathbf{0}\}_{3\times1} \\ \{\mathbf{0}\}_{3\times1} \end{Bmatrix} \tag{9.63}$$

where $l_{3\text{–}4}$ is the length of edge 3–4. Equation (9.63) implies that the distributed forces are equally divided and applied at the two nodes. This conclusion suggests also to evenly distribute surface forces applied on any face of the element, and to evenly distribute body forces applied on the entire body of the element.

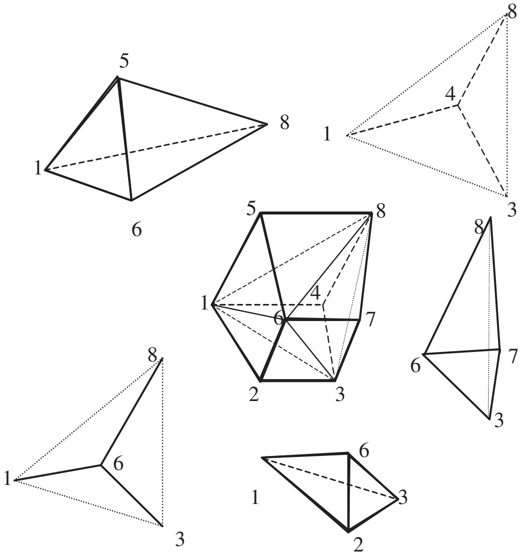


Figure 9.11. A hexahedron broken up into five tetrahedrons.

9.3.3 Using Tetrahedrons to form Hexahedrons

An alternative method of formulating hexahedron elements is to make use of tetrahedron elements. This is built upon the fact that a hexahedron can be said to be made up of numerous tetrahedrons. Figure 9.11 shows how a hexahedron can be made up of five tetrahedrons. Of course, this is not the only way that a hexahedron can be made up of five tetrahedrons, and it can also be made up of six tetrahedrons, as shown in Figure 9.12. Similarly, there is more than one way of dividing a hexahedron into six tetrahedrons. In this way, the element matrices for a hexahedron can be formed by assembling all the matrices for the tetrahedron elements, each of which is developed in Section 9.2.2. The assembly is done in a similar way to the assembly between elements.

9.4 HIGHER ORDER ELEMENTS

9.4.1 Tetrahedron Elements

Two higher order tetrahedron elements with 10 and 20 nodes are shown in Figures 9.13(a) and (b), respectively. The 10-node tetrahedron element is a quadratic element. Compared with the linear tetrahedron element (four-nodal) developed earlier, six additional nodes are added at the middle of the edges of the element. In developing the 10-nodal tetrahedron element, a complete polynomial up to second order can be used. The shape functions for

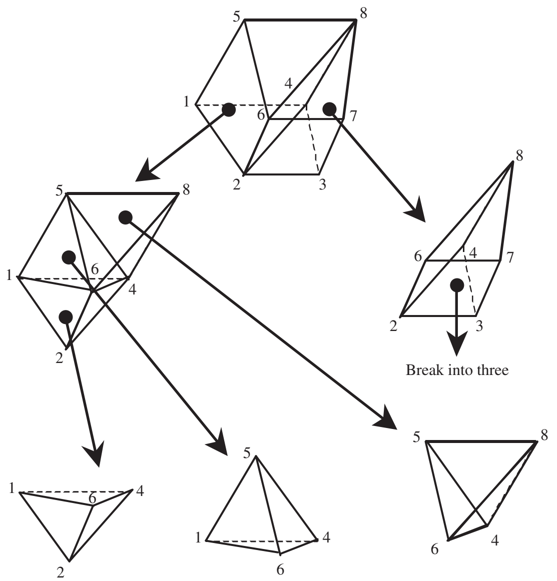


Figure 9.12. A hexahedron broken up into six tetrahedrons.

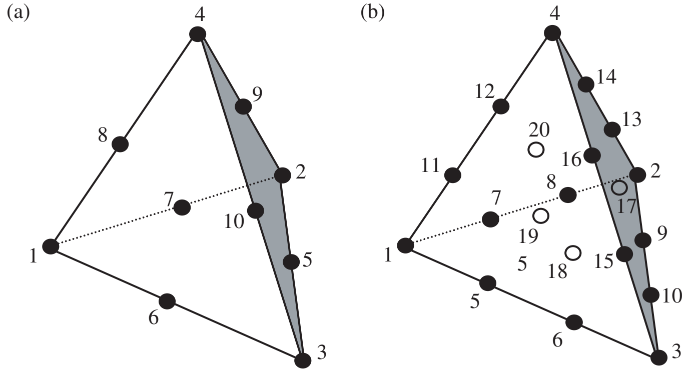


Figure 9.13. Higher order 3D tetrahedron elements. (a) 10-node tetrahedron element; (b) 20-node tetrahedron element.

this quadratic tetrahedron element in the volume coordinates are given as follows:

$$\begin{aligned} &N_i = (2L_i - 1)L_i \quad \text{for corner nodes } i = 1, 2, 3, 4 \\ &\left.\begin{aligned} N_5 &= 4L_2L_3 \\ N_6 &= 4L_1L_3 \\ N_7 &= 4L_1L_2 \\ N_8 &= 4L_1L_4 \\ N_9 &= 4L_2L_4 \\ N_{10} &= 4L_3L_4 \end{aligned}\right\} \text{for mid-edge nodes} \end{aligned} \tag{9.64}$$

where L_i is the volume coordinate, which is the same as the shape function for the linear tetrahedron elements given by Eq. (9.15).

The 20-node tetrahedron element is a cubic element. Compared with the linear tetrahedron element (four-nodal) developed earlier, two additional nodes are added evenly on each edge of the element, and four-node central-face nodes are added at the geometry centre of each triangular surface of the element. In developing the 20-nodal tetrahedron element, a complete polynomial up to third order can be used. The shape functions for this cubic tetrahedron element in the volume coordinates are given as follows:

$$\begin{aligned}
&N_i = \tfrac{1}{2}(3L_i - 1)(3L_i - 2)L_i \quad \text{for corner nodes } i = 1, 2, 3, 4\\
&\left.\begin{array}{ll}
N_5 = \frac{9}{2}(3L_1 - 1)L_1L_3 & N_{11} = \frac{9}{2}(3L_1 - 1)L_1L_4\\
N_6 = \frac{9}{2}(3L_3 - 1)L_1L_3 & N_{12} = \frac{9}{2}(3L_4 - 1)L_1L_4\\
N_7 = \frac{9}{2}(3L_1 - 1)L_1L_2 & N_{13} = \frac{9}{2}(3L_2 - 1)L_2L_4\\
N_8 = \frac{9}{2}(3L_2 - 1)L_1L_2 & N_{14} = \frac{9}{2}(3L_4 - 1)L_2L_4\\
N_9 = \frac{9}{2}(3L_2 - 1)L_2L_3 & N_{15} = \frac{9}{2}(3L_3 - 1)L_3L_4\\
N_{10} = \frac{9}{2}(3L_3 - 1)L_2L_3 & N_{16} = \frac{9}{2}(3L_4 - 1)L_3L_4
\end{array}\right\} \text{for edge nodes}\\
&\left.\begin{array}{l}
N_{17} = 27L_2L_3L_4\\
N_{18} = 27L_1L_2L_3\\
N_{19} = 27L_1L_3L_4\\
N_{20} = 27L_1L_2L_4
\end{array}\right\} \text{for centre surface nodes}
\end{aligned} \tag{9.65}$$

where L_i is the volume coordinate, which is the same as the shape function for the linear tetrahedron elements given by Eq. (9.15).

9.4.2 Brick Elements

Lagrange type elements

The Lagrange type brick elements can be developed in precisely the same manner as the 2D rectangular elements described in Chapter 7. Consider a brick element with $n_d = (n+1)(m+1)(p+1)$ nodes shown in Figure 9.14. The element is defined in the domain of $(-1 \le \xi \ge 1, -1 \le \eta \ge 1, -1 \le \zeta \ge 1)$ in the natural coordinates ξ, η and ζ. Due to the regularity of the nodal distribution along the ξ, η and ζ directions, the shape function of the element can be simply obtained by multiplying one-dimensional shape functions with respect to the ξ, η and ζ directions using the Lagrange interpolants defined in Eq. (4.82) [Zienkiewicz *et al.*, 2000]:

$$N_i = N_I^{1D} N_J^{1D} N_K^{1D} = l_I^n(\xi) l_J^m(\eta) l_K^p(\varsigma) \tag{9.66}$$

Due to the delta function property of the 1D shape functions given in Eq. (4.83), it is easy to confirm that the N_i given by Eq. (9.66) also has the delta function property.

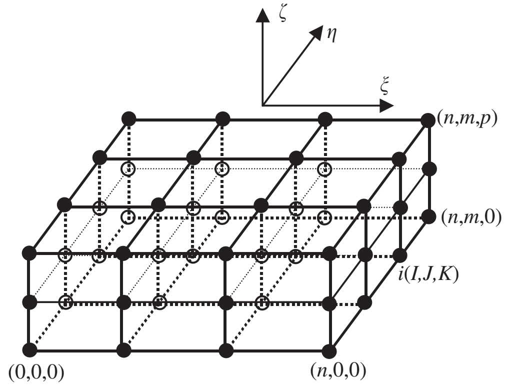


Figure 9.14. Brick element of arbitrary high orders.

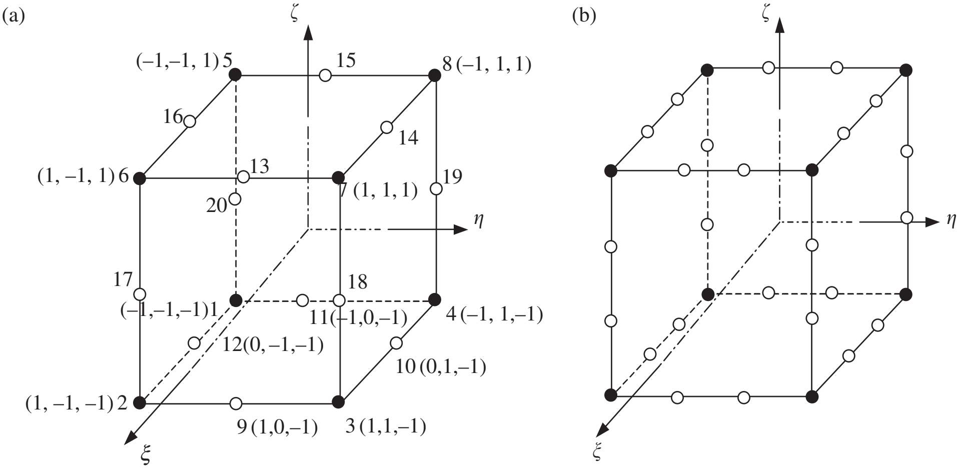


Figure 9.15. High order 3D serendipity elements. (a) 20-node quadratic element; (b) 32-node cubic element.

Serendipity type elements

The method used in constructing the Lagrange type of elements is very systematic. However, the Lagrange type of elements is not very widely used, due to the presence of the interior nodes. A serendipity type of brick elements without interior nodes is created by inspective construction methods as described in Chapter 7 for 2D rectangular elements.

Figure 9.15(a) shows a 20-nodal tri-quadratic element. The element has eight corner nodes and twelve mid-side nodes. The shape functions in the natural coordinates for the quadratic brick element are given as follows:

$$\begin{aligned}
N_j &= \tfrac{1}{8}(1+\xi_j\xi)(1+\eta_j\eta)(1+\varsigma_j\varsigma)(\xi_j\xi+\eta_j\eta+\varsigma_i\varsigma-2) \\
&\qquad\qquad\qquad\qquad\qquad\qquad \text{for corner nodes } j = 1,\ldots,8 \\
N_j &= \tfrac{1}{4}(1-\xi^2)(1+\eta_j\eta)(1+\varsigma_j\varsigma) \quad \text{for mid-side nodes } j = 10, 12, 14, 16 \\
N_j &= \tfrac{1}{4}(1-\eta^2)(1+\xi_j\xi)(1+\varsigma_j\varsigma) \quad \text{for mid-side nodes } j = 9, 11, 13, 15 \\
N_j &= \tfrac{1}{4}(1-\varsigma^2)(1+\xi_j\xi)(1+\eta_j\eta) \quad \text{for mid-side nodes } j = 17, 18, 19, 20
\end{aligned} \tag{9.67}$$

where (ξ_j, η_j) are the natural coordinates of node j. It is very easy to observe that the shape functions possess the delta function property. The shape function is constructed by simple inspections, making use of the shape function properties. For example, for corner node 2 (where $\xi_2 = 1$, $\eta_2 = -1$, $\zeta_2 = -1$), the shape function N_2 has to pass the following four planes as shown in Figure 9.16 to ensure its vanishing at remote nodes:

$$\begin{aligned}
1+\xi = 0 &\Rightarrow \text{vanishes at nodes } 1, 4, 5, 8, 11, 15, 19, 20 \\
\eta - 1 = 0 &\Rightarrow \text{vanishes at nodes } 3, 4, 7, 8, 10, 14, 18, 19 \\
\varsigma - 1 = 0 &\Rightarrow \text{vanishes at nodes } 5, 6, 7, 8, 13, 14, 15, 16 \\
\xi - \eta - \varsigma - 2 = 0 &\Rightarrow \text{vanishes at nodes } 9, 12, 17
\end{aligned} \tag{9.68}$$

The shape N_2 can then be immediately written as

$$N_2 = C(1+\xi)(1-\eta)(1-\varsigma)(\xi-\eta-\varsigma-2) \tag{9.69}$$

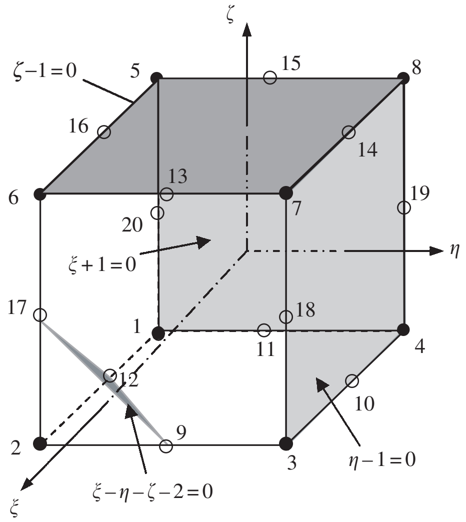


Figure 9.16. Construction of 20-node serendipity element. Four flat planes passing through the remote nodes of node 2 are used.

where C is a constant to be determined using the condition that it has to be unity at node 2 at ($\xi_2 = 1, \eta_2 = -1, \zeta_2 = -1$), which gives

$$C = \frac{1}{(1+1)(1-(-1))(1-(-1))(1-(-1)-(-1)-2)} = \frac{1}{8} \tag{9.70}$$

We finally have

$$N_2 = \tfrac{1}{8}(1+\xi_2\xi)(1+\eta_2\eta)(1+\varsigma_2\varsigma)(\xi_2\xi+\eta_2\eta+\varsigma_2\varsigma-2) \tag{9.71}$$

which is the first equation in Eq. (9.67) for $j = 1$.

Shape functions at all the other corner nodes can be constructed in exactly the same manner. As for the mid-side nodes, say node 9, we enforce the shape function passing through the following four planes, as shown Figure 9.17.

$$\begin{aligned}
1+\xi &= 0 \quad \Rightarrow \quad \text{vanishes at nodes } 1, 4, 5, 8, 11, 15, 19, 20 \\
\eta-1 &= 0 \quad \Rightarrow \quad \text{vanishes at nodes } 3, 4, 7, 8, 10, 14, 18, 19 \\
\varsigma-1 &= 0 \quad \Rightarrow \quad \text{vanishes at nodes } 5, 6, 7, 8, 13, 14, 15, 16 \\
\eta+1 &= 0 \quad \Rightarrow \quad \text{vanishes at nodes } 1, 2, 5, 6, 12, 13, 16, 17
\end{aligned} \tag{9.72}$$

The shape N_5 can then be immediately written as

$$N_9 = C(1-\eta^2)(1+\xi)(1-\varsigma) \tag{9.73}$$

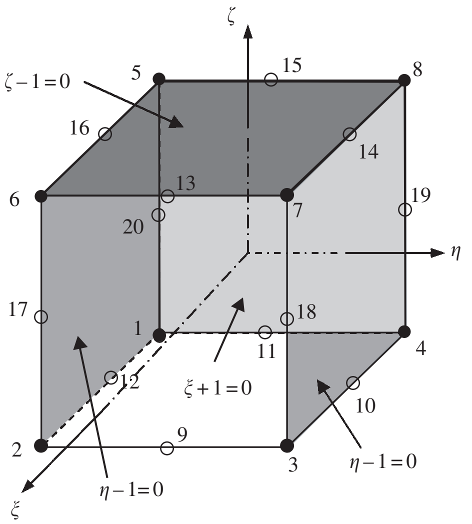


Figure 9.17. Construction of 20-node serendipity element. Four flat planes passing through the remote nodes of node 2 are used.

where C is a constant to be determined using the condition that it has to be unity at node 5 at ($\xi_9 = 1$, $\eta_9 = 0$, $\zeta_9 = -1$), which gives

$$C = \frac{1}{(1-\eta^2)(1+\xi)(1-\varsigma)} = \frac{1}{(1-0^2)(1+1)(1-(-1))} = \frac{1}{4} \tag{9.74}$$

We finally have

$$N_9 = \frac{1}{4}(1-\eta^2)(1+\xi_9\xi)(1+\varsigma_9\varsigma) \tag{9.75}$$

which is the third equation in Eq. (9.67) for $j = 9$.

Because the delta function property is used for the construction of shape functions given in Eq. (9.67), they of course, possess, the delta function property. It can easily be seen that all the shape functions can be formed using the following common set of basis functions:

$$\begin{aligned} &1, \xi, \eta, \varsigma\xi\eta, \eta\varsigma, \xi\varsigma, \xi^2, \eta^2, \varsigma^2, \\ &\xi\eta\varsigma, \xi\eta^2, \xi\varsigma^2, \eta\xi^2, \eta\varsigma^2, \varsigma\xi^2, \varsigma\eta^2, \xi^2\eta\varsigma, \eta^2\xi\varsigma, \xi\eta\varsigma^2 \end{aligned} \tag{9.76}$$

that are linearly-independent and contain all the linear terms. From Lemmas 2 and 3, we confirm that the shape functions are partitions of unity, and at least linear field reproduction. Hence, they satisfy the sufficient requirements for FEM shape functions.

Following the similar procedure, the shape functions for the 32-node tri-cubic element shown in Figure 9.15(b) can be written as

$$\begin{aligned} N_j &= \tfrac{1}{64}(1+\xi_j\xi)(1+\eta_j\eta)(1+\varsigma_j\varsigma)(9\xi^2+9\eta^2+9\varsigma^2-19) \\ &\quad \text{for corner nodes } j = 1, \ldots, 8 \\ N_j &= \tfrac{9}{64}(1-\xi^2)(1+9\xi_j\xi)(1+\eta_j\eta)(1+\varsigma_j\varsigma) \\ &\quad \text{for side nodes with } \xi_j = \pm\tfrac{1}{3}, \eta_j = \pm 1 \text{ and } \varsigma_j = \pm 1 \\ N_j &= \tfrac{9}{64}(1-\eta^2)(1+9\eta_j\eta)(1+\xi_j\xi)(1+\varsigma_j\varsigma) \\ &\quad \text{for side nodes with } \eta_j = \pm\tfrac{1}{3}, \xi_j = \pm 1 \text{ and } \varsigma_j = \pm 1 \\ N_j &= \tfrac{9}{64}(1-\varsigma^2)(1+9\varsigma_j\varsigma)(1+\xi_j\xi)(1+\eta_j\eta) \\ &\quad \text{for side nodes with } \varsigma_j = \pm\tfrac{1}{3}, \xi_j = \pm 1 \text{ and } \eta_j = \pm 1 \end{aligned} \tag{9.77}$$

The reader is encouraged to figure out what are the planes that should be used to form the shape functions listed in Eq. (9.77). When $\zeta = \zeta_i = 1$, the above equations reduce to a two-dimensional case of serendipity quadratic and cubic elements defined by Eqs. (7.107), (7.111) and (7.113).

9.5 ELEMENTS WITH CURVED SURFACES

Using high order elements, elements with curved surfaces can be used in the modelling. Two relatively frequently used higher order elements of curved edges are shown in Figure 9.18(a). In formulating these types of elements, the same mapping technique used for the linear

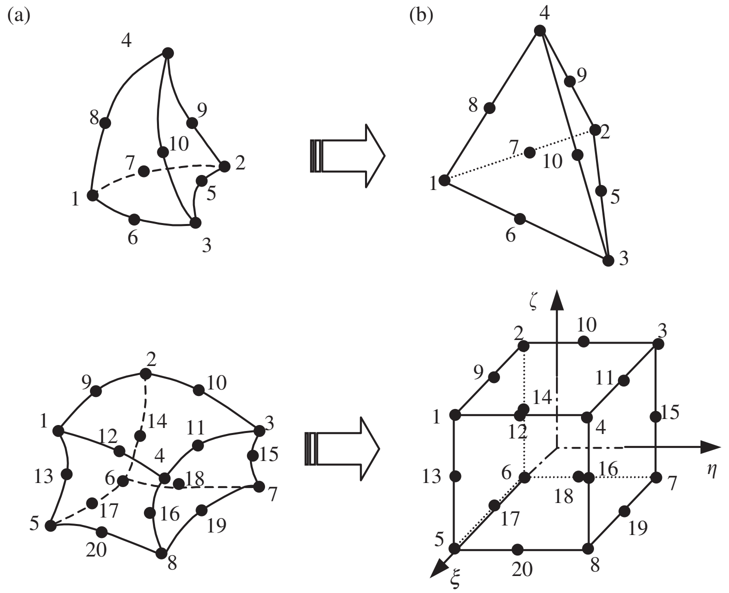


Figure 9.18. 3D solid elements with curved surfaces. (a) Elements with curved surfaces in the physical coordinate system; (b) brick elements obtained by mapping.

hexadron elements (Section 9.3) can be used. In the physical coordinate system, elements with curved edges are first formed in the problem domain as shown in Figure 9.18(a). These elements are then mapped into the natural coordinate system using Eq. (9.34). The elements mapped in the natural coordinate system will have straight edges, as shown in Figure 9.18(b).

Higher order elements of curved surfaces are often used for modelling curved boundaries. Note that elements with excessively curved edges may cause problems in the numerical integration. Therefore, more elements should be used where the curvature of the boundary is large. In addition, it is recommended that in the internal portion of the domain, an element with straight edges should be used whenever possible.

9.6 CASE STUDY: STRESS AND STRAIN ANALYSIS OF A QUANTUM DOT HETEROSTRUCTURE

Quantum dots are clusters of atoms nanometres in size, usually made from semiconducting materials like silicon, cadmium selenide or gallium arsenide. What makes quantum dots interesting is that they have unusual electrical and optical properties, hence they have the potential for use in a wide variety of novel electronic devices, including light emitting diodes, photovoltaic cells, and quantum semiconductor lasers. An interesting way of fabricating such quantum dot structures is to actually grow the dots directly by depositing a thin film

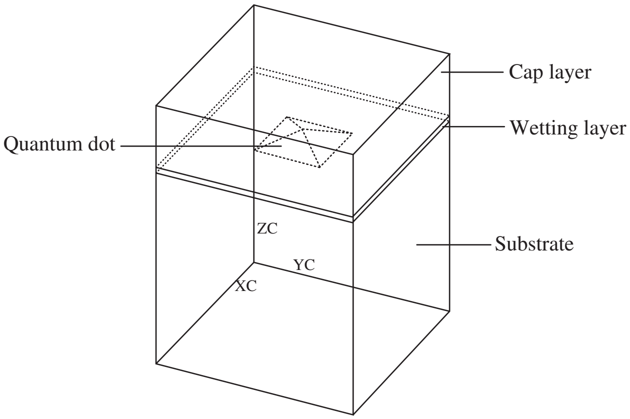


Figure 9.19. Schematic representation of a quantum dot heterostructure.

layer of material on a substrate under appropriate growth conditions. Usually, the thin film layer is of a different material to the substrate material, and thus such structures are also generally known as heterostructures. This growth mode is due to the mismatch in the lattice parameters of the different materials, and is known as the Stranski–Krastanow (SK) growth mode.

Study of such quantum dot heterostructures is actually a very active research area at the time at which this book was written. In this case study, an example of modelling a 3D finite element model to analyse the stress distribution in and around such structures will be shown. The stress distribution actually affects the electrical and optical properties of the quantum dot structure. Furthermore, the quantum dot formation in multiple thin film layers is also very much dependent on this stress distribution.

Figure 9.19 shows a schematic representation of a quantum dot grown on top of the substrate and embedded in a cap layer. This is just a single quantum dot, and can probably be considered as a single basic unit of the heterostructure. In reality, there could be many of such quantum dots distributed on top of a layer of substrate. It can also be seen from the figure that the quantum dot is usually pyramidical or trapezoidal in shape. The pyramidal shape of the quantum dot is often approximated by using a 2D axisymmetric model of a cone by many analysts. However, it should be noted that using a 2D axisymmetric model is not fully representative of the pyramidal shape, and for the purpose of this chapter, this case study will be using the 3D solid element to model the structure.

9.6.1 Modelling

Meshing

As mentioned, the modelling of any 3D structure is generally more complex and tedious. In this case, eight-nodal, hexahedron elements are being used for meshing of the 3D geometry. It can be seen from Figure 9.19 that the problem domain is very much symmetrical.

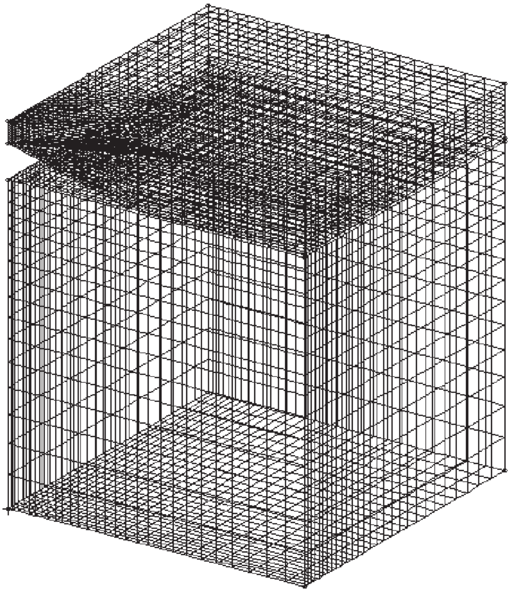


Figure 9.20. 3D mesh of the matrix.

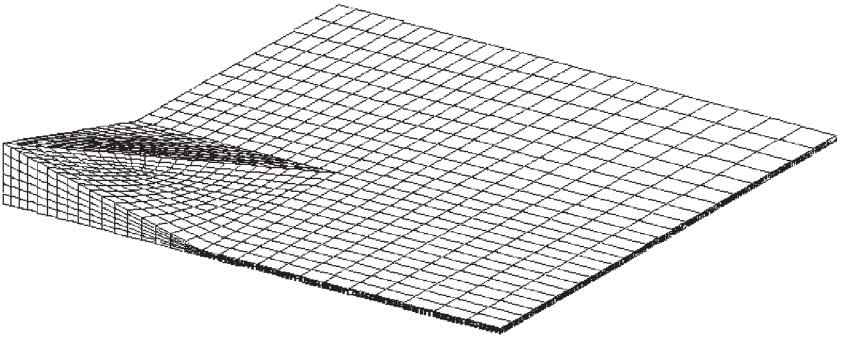


Figure 9.21. 3D mesh of the island.

Therefore, to work on a more manageable problem, a quarter of the model is being modelled using mirror symmetry. Note that it is also possible to use a one-eighth model, which then requires the use of Multi-Point Constraints (MPC) equations (see Chapter 11).

Proper meshing in this case is very important, as it has been found that a poor mesh usually yields bad results. The 3D mesh of the heterostructure is shown in Figures 9.20 and 9.21. The model is generally divided into two main parts geometrically for the analysts to distinguish them more conveniently. The parts of the heterostructure comprising the substrate and the cap layer, as shown in Figure 9.19, is grouped together as the matrix; and the parts of the heterostructure comprising the wetting layer and the quantum dot itself

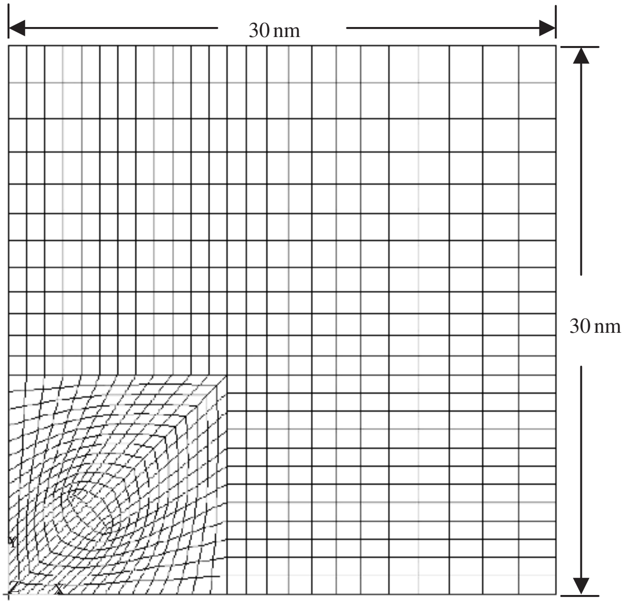


Figure 9.22. Plan view of finite element mesh of quantum dot heterostructure.

Table 9.1. Material properties of GaAs and InAs

Material	E (Gpa)	υ
GaAs	86.96	0.31
InAs	51.42	0.35

are grouped together as the island. Figure 9.22 also shows a plan view of the mesh of the island (or matrix). It can be seen how smaller elements are concentrated at and around the pyramidal quantum dot. To generate the mesh here, the analyst has employed the aid of automatic mesh generators that can still mesh the relatively complex shape of the pyramid with hexahedron elements. Some mesh generators may not be able to achieve this, and one may end up with either tetrahedron elements or a mixture of both hexahedron and tetrahedron elements.

Material properties

In this case study, the heterostructure system of Indium Arsenide (InAs) quantum dots embedded in a Gallium Arsenide (GaAs) substrate and cap layer is analysed. Therefore, the matrix part of the model will be of the material GaAs and the island part of the model will be of the material InAs. This is an example of the convenience of dividing the model into these two parts. It is assumed here that the materials have isotropic properties, listed in Table 9.1.

Constraints and boundary conditions

As the model is a symmetric quarter model, symmetrical boundary conditions must be applied. Here it means that the nodes on the planes corresponding to $x = 0\,\text{nm}$, $y = 0\,\text{nm}$, $x = 30\,\text{nm}$ and $y = 30\,\text{nm}$ have their displacement components normal to their respective planes constrained. Another displacement boundary condition would be the base of the matrix, where all the displacement components of the nodes are constrained.

There is also a contact constraint condition imposed between the outer surfaces of the island and the surfaces of the cap layer and the substrate in the matrix. Contact modelling is a relatively advanced technique, and will not be covered in this book. Basically, contact modelling is used to model the sliding or the movement between two surfaces. ABAQUS offers a 'tied' contact condition where the two surfaces in contact are actually tied to one another. This 'tied' contact condition is used here to model the bonding between the island (InAs) and the matrix (GaAs).

There is actually no load acting on this model. Rather, thermal expansivity is being made use of to simulate the strain induced due to the lattice mismatch between GaAs and InAs. The strain induced due to the lattice mismatch can be calculated from the lattice parameters to be -0.067. To represent this lattice mismatch, a corresponding thermal expansion coefficient of $\alpha_T = 0.067$ is applied to the elements in the island, and the temperature is raised by 1 K. This would effectively result in an expansion of the island, and because it is constrained by the matrix, thermal strain corresponding to the lattice mismatch strain is induced. Note that this thermal expansion does not take place in the physical case, but is just used to produce the mismatch strain. This thermal strain actually contributes to the force vector in the finite element equations.

9.6.2 ABAQUS Input File

Part of the ABAQUS input file for the problem defined above is shown below. As the problem is quite large, the full input file would consist of a large amount of data defining the nodes, elements, and so on. As such, the full data will not be included here and some parts of the input file that have been explained in previous case studies will not be explained again here.

```
*HEADING, SPARSE
Calculation of stress distribution in quantum dot structure
**
*NODE
 :
 :
**
** Elements are divided into two main parts: ISLAND and MATRIX
** Elements used are 8-nodal, hexahedral elements (C3D8)
**
```

Nodal cards

Node ID, x-coordinate, y-coordinate, z-coordinate.

```
*ELEMENT, TYPE=C3D8, ELSET=ISLAND
⋮
*ELEMENT, TYPE=C3D8, ELSET=MATRIX
⋮
```

Element (connectivity) cards

Element ID, node 1, node 2, node 3, . . ., node 8.

```
**
**
*NSET, NSET=ISLAND
⋮
```

Node set

Nodes in ISLAND are grouped in a node set named ISLAND.

```
**
*NSET, NSET=BASE
 ⋮
```

Node set

Nodes on base surface grouped in a node set named BASE.

```
**
*NSET, NSET=FIXED_X
 ⋮
```

Node set

Nodes to be constrained in x direction grouped in node set, FIXED_X.

```
**
*NSET, NSET=FIXED_Y
 ⋮
```

Node set

Nodes to be constrained in y direction grouped in node set, FIXED_Y.

```
**
*SOLID SECTION, ELSET=ISLAND, MATERIAL=INAS
1.,
**
**
*SOLID SECTION, ELSET=MATRIX, MATERIAL=GAAS
1.,
```

Property cards

Define properties to the elements of sets “ISLAND” and “MATRIX”. It will have the material properties defined under “INAS” and “GASS”, respectively.

```
**
** GaAs
**
*MATERIAL, NAME=GAAS
**
*ELASTIC, TYPE=ISO
   86.96,      0.31
**
** InAs
**
*MATERIAL, NAME=INAS
**
```

Material cards

Define material properties under the name “GAAS” and “INAS”. Elastic properties are defined. TYPE = ISO represents isotropic properties. Note that for “INAS”, the thermal expansion coefficient is defined under *EXPANSION.

```
*ELASTIC, TYPE=ISO
          51.42, 0.35
**
*EXPANSION, TYPE=ISO
0.067,
**
** Displacement boundaries
**
*BOUNDARY, OP=NEW
BASE, ENCASTRE
FIXED_X, XSYMM
FIXED_Y, YSYMM
**
** contact1
**
```

BC cards

The nodes grouped under BASE, FIXED_X and FIXED_Y are given the corresponding constraints. ENCASTRE represents fully clamped in boundary; XSYMM represents conditions symmetrical to plane x = constant; YSYMM represents conditions symmetrical to plane y = constant.

```
*SURFACE DEFINITION, NAME=M20
⋮
```

Defines contact surfaces.

```
*SURFACE DEFINITION, NAME=S20
⋮
```

Defines contact surfaces.

```
**
*CONTACT PAIR, INTERACTION=I20, ADJUST=0.0001, TIED
S20, M20
*SURFACE INTERACTION, NAME=I20
**
** contact2
**
*SURFACE DEFINITION, NAME=M21
⋮
*SURFACE DEFINITION, NAME=S21
⋮
```

Contact cards

Contact conditions are being defined here. The surfaces to be in contact with are defined in *SURFACE DEFINITION. The contact conditions are then specified in *CONTACT PAIR and *SURFACE INTERACTION. Details will not be shown here, since it is beyond the scope of this book.

```
**
*CONTACT PAIR, INTERACTION=I21, ADJUST=0.0001, TIED
S21, M21
*SURFACE INTERACTION, NAME=I21
**
**
*INITIAL CONDITIONS, TYPE=TEMPERATURE
ISLAND, 300.
**
**
```

IC cards

Initial temperature conditions are being defined and nodes in ISLAND are set at a temperature of 300 K.

```
*STEP, AMPLITUDE=RAMP
Linear Static Analysis
**
**
*STATIC
**
**
**
*TEMPERATURE, OP=NEW
ISLAND,       301.
**
**
*NODE PRINT, FREQ=1
U,
*NODE FILE, FREQ=1
U,
**
*EL PRINT, POS=INTEG, FREQ=1
S,
E,
*EL FILE, POS=INTEG, FREQ=1
S,
E,
**
*END STEP
```

Control cards

Indicate the STATIC analysis procedure.

Load cards

The load here is temperature and the nodes in the ISLAND are given a temperature of 301K, which implies a raise of 1K from the initial conditions.

Output control cards

Define the output requested. In this case, the displacements, U, the stresses, S and strains, E.

The input file provides the information ABAQUS needs to perform tasks like forming the stiffness matrix and the force vector (no mass matrix, since this is a static analysis). The full input file may consist of many pages, which is common for large problems.

9.6.3 Solution Process

The information provided in the input file is very similar to previous case studies in this book. The nodal and element connectivity information is read for the formulation of the element matrices. The element type used here is C3D8, which represents a 3D, hexahedral element with eight nodes. More 3D element types are also available in the ABAQUS element library. The material properties provided in the input file will also be used to formulate the element stiffness matrix (Eq. (9.51)). Recall that the integration in the stiffness matrix is usually carried out using the Gauss integration scheme sampled in three directions, and in ABAQUS, the default number of integration points per face of the hexahedral element is four, making the total number of integration points per element 24. All the element matrices will be assembled together using the connectivity information provided. Application of the boundary conditions and the thermal strain induced by the thermal expansion is carried out by the specifications in the boundary cards and the load card. Finally, the finite element equation will be solved using the algorithm for static analysis, as discussed in Chapter 6.

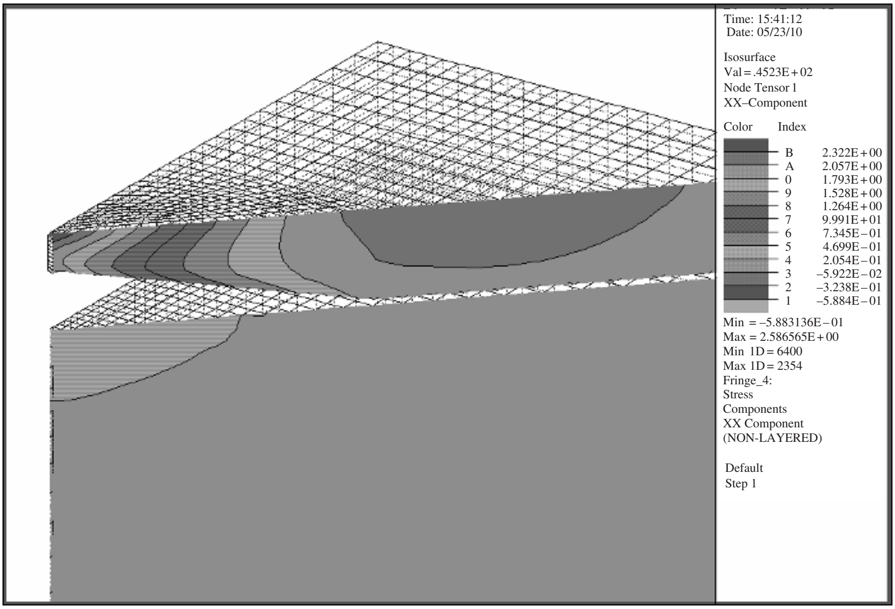


Figure 9.23. Stress, σ_{xx} distribution of plane $\theta = 45°$ in matrix.

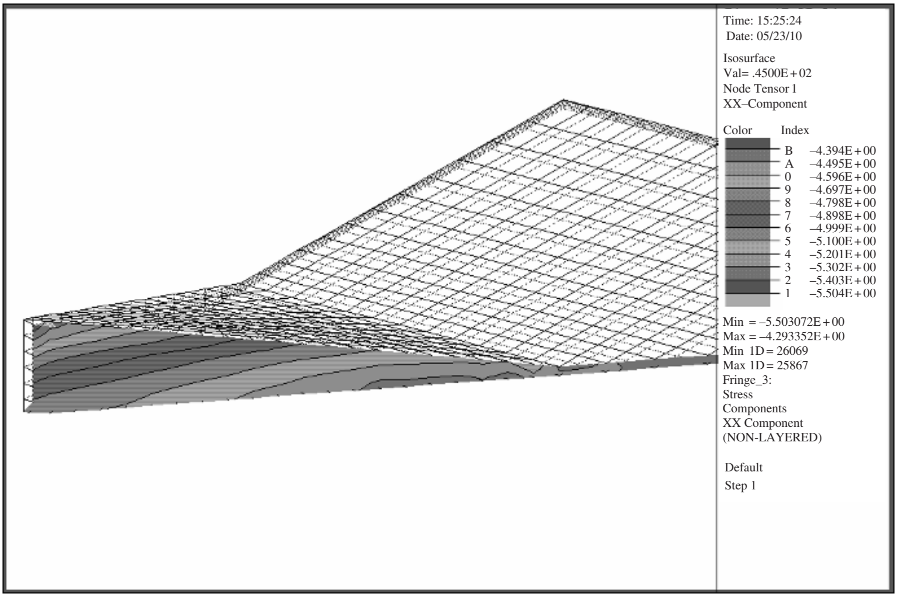


Figure 9.24. Stress, σ_{xx}, distribution of plane $\theta = 45°$ in island.

9.6.4 Result and Discussion

Running the problem in ABAQUS, we are able to get the stress distribution and the strain distribution as requested in the input file. Figures 9.23 and 9.24 show the stress distribution obtained in the matrix and island, respectively, of a particular plane of $\theta = 45°$, where θ is measured from the x–z plane counter-clockwise. Despite the extra effort in the meshing of a 3D model, the advantage is that it enables the analyst to view, in this case, the stress distribution in any arbitrary plane in the entire model. This would be difficult to achieve if a 2D, axis-symmetric approximation is carried out instead. From the stress distribution, one can observe that there are compressive stresses in the island and tensile stresses in the matrix area above the island. In the island, there is also stress relaxation in the quantum dot, with the maximum stress relaxation at the tip of the pyramidal quantum dot. This actually verifies the thermodynamics aspect of quantum dot formation, since the formation of a quantum dot results in a lower energy level (lower elastic strain energy). The tensile stress in the matrix area above the quantum dot is also important, as this stress actually causes a subsequent quantum dot to be formed directly above the buried quantum dot when a subsequent InAs layer is deposited.

9.7 REVIEW QUESTIONS

1. Can 3D solid elements be used for solving 2D plane stress and plane strain problems? Give justification to your answer.
2. What is the defference between using tetrahedron elements and hexahedron elements derived using an assembly of tetrahedron elements? Can they give the same results for the same problem? Give justification to your answer.
3. Can one develop pentahedron elements? How?
4. How many Gauss points should be used for evaluating mass and stiffness matrices for four-node tetrahedron elements? Give justification to your answer.
5. How many Gauss points should be used for evaluating mass and stiffness matrices for eight-node hexahedron elements? Give justification to your answer.
6. If a higher order shape function is used, do Eqs. (9.30) and (9.63) still hold? Give justification to your answer.

10

SPECIAL PURPOSE ELEMENTS

10.1 INTRODUCTION

This chapter introduces some special purpose elements and methods that are particularly designed for specific circumstances. They are used for very specific purposes to either simplify meshing and calculation, or to obtain better accuracy, which usual elements cannot obtain. These include crack tip elements, infinite elements, finite strip elements and strip elements. The characteristics of these elements are summarized in Table 10.1.

Table 10.1. Special elements

Elements	Application	Approach/features
Crack tip element	Simulation of problem domain with cracks	Use mapping techniques to create singular stress field; Reduce elements density near crack tips
Infinite elements	Simulation problems with an infinite or semi-infinite domain	Using mapping techniques to create fields that decay with distance; Very few elements are needed to model infinite boundary
Finite strip element	Model structures with regular geometric domain	Shape function with series expansion in one direction; Very few elements needed; Simple boundary conditions
Strip element	Model structures with regular geometric domain	Semianalytical approach; Very few elements needed; Arbitrary boundary conditions including infinite boundaries

10.2 CRACK TIP ELEMENTS

In fracture mechanics, much interest for analysts is on the tip of the crack, as it is a singularity point where the stress field becomes mathematically infinite. When modelled with the conventional, polynomial-based finite elements discussed in previous chapters, the finite element approximations are usually quite bad unless a very dense mesh consisting of numerous small elements is modelled around the crack tip. This may not prove feasible at times, and is highly inefficient when computational resources are limited. It will be a better option to model such a problem with what is known as crack tip elements, sometimes known as *singularity elements*. Such an element was introduced at almost the same time by Henshell and Shaw [1975] and Barsoum [1976, 1977].

From theories of linear elastic fracture mechanics, the stresses near the crack tip are characterized by the stress intensity factor, K_I, in Mode I fracture as

$$\begin{bmatrix} \sigma_{xx} \\ \sigma_{xy} \\ \sigma_{yy} \end{bmatrix} = \frac{K_I}{\sqrt{2\pi r}} \cos\frac{\theta}{2} \begin{bmatrix} 1 - \sin\theta/2 \sin 3\theta/2 \\ \sin\theta/2 \sin 3\theta/2 \\ 1 + \sin\theta/2 \sin 3\theta/2 \end{bmatrix} \tag{10.1}$$

and the displacement near the crack tip is expressed as

$$\begin{bmatrix} u \\ v \end{bmatrix} = \frac{K_I\sqrt{r}}{2G\sqrt{2\pi}} \begin{bmatrix} \cos\theta/2(\kappa - 1 + 2\sin^2\theta/2) \\ \sin\theta/2(\kappa + 1 - 2\cos^2\theta/2) \end{bmatrix} \tag{10.2}$$

where G is the shear modulus, and $\kappa = 3 - 4\upsilon$ (plane strain) or $(3 - \upsilon)/(1+\upsilon)$ (plane stress). Mode I fracture is considered to be the opening of the crack, as shown in Figure 10.1, and r and θ are as shown. From Eqs. (10.1) and (10.2), it can be seen that the stress varies inversely with $\sqrt{r}$ and the displacement varies proportionally with $\sqrt{r}$. Note the presence of singularity of the stresses at the crack tip itself when r approaches zero.

To approximate the behaviour of the stresses and displacements near the crack tip according to the theories of fracture mechanics, a special eight-nodel, quadratic, isoparametric element as shown in Figure 10.2 can be formulated. This element is exactly the same as the usual eight-nodel isoperimetric quadratic element, except that the middle modes on the edges to the crack tip are moved by a quarter of the edge length towards the crack tip. The following explains how the stress singularity is created by this simple modification.

Consider the element side joining nodes 1, 2 and 3 of the isoparametric quadratic element, as shown in Figure 10.3. Following formulation of the conventional eight-node

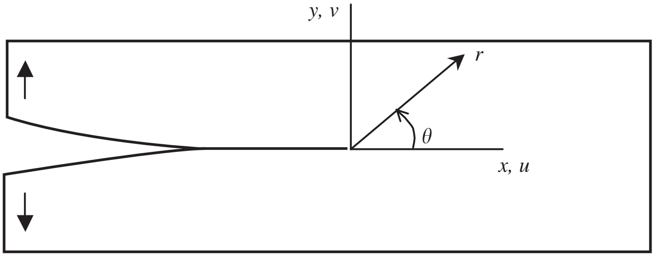


Figure 10.1. Mode I crack opening deformation.

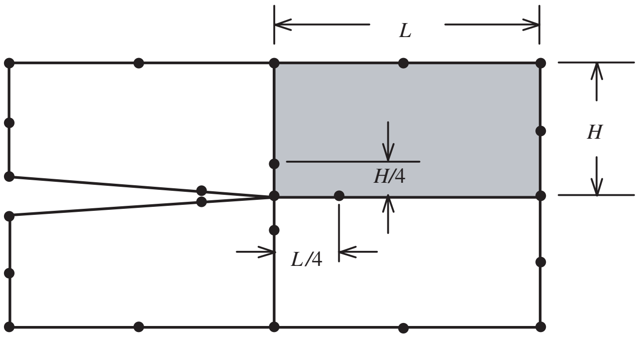


Figure 10.2. Modelling of crack tip with crack tip elements.

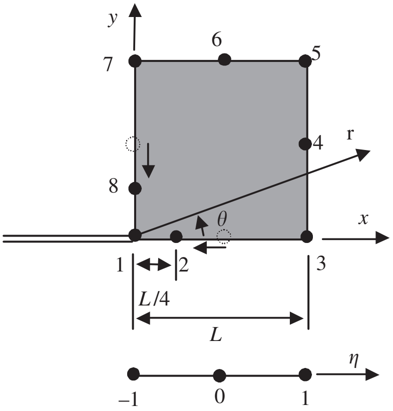


Figure 10.3. Eight-node, isoparametric, quadratic crack tip element.

element, the coordinate x and displacement u are both interpolated by shape functions as follows:

$$x = -0.5\eta(1-\eta)x_1 + (1+\eta)(1-\eta)x_2 + 0.5\eta(1+\eta)x_3 \tag{10.3}$$

$$u = -0.5\eta(1-\eta)u_1 + (1+\eta)(1-\eta)u_2 + 0.5\eta(1+\eta)u_3 \tag{10.4}$$

Let both x and u be measured from node 1, and let the mid-side node 2 be moved to the quarter-point node 2. For a side of length L, we have

$$x_1 = 0, \quad x_2 = L/4, \quad x_3 = L, \quad u_1 = 0 \tag{10.5}$$

Substitution of Eq. (10.5) into Eqs. (10.3) and (10.4) leads to

$$x = 0.25(1+\eta)(1-\eta)L + 0.5\eta(1+\eta)L \tag{10.6}$$

$$u = (1+\eta)(1-\eta)u_2 + 0.5\eta(1+\eta)u_3 \tag{10.7}$$

Simplifying the above equations will give us

$$x = 0.25(1+\eta)^2 L \tag{10.8}$$

$$u = (1+\eta)[(1-\eta)u_2 + 0.5\eta u_3] \tag{10.9}$$

Now, we know that along the x-axis, $x = r$. Therefore,

$$r = 0.25(1+\eta)^2 L \quad \text{or} \quad (1+\eta) = 2\sqrt{\frac{r}{L}} \tag{10.10}$$

Substitution of Eq. (10.10) into Eq. (10.9) leads to

$$u = 2(\sqrt{r}/\sqrt{L})[(1-\eta)u_2 + 0.5\eta u_3] \tag{10.11}$$

Notice that by shifting the middle node to the quarter position, the displacement now follows a behaviour that is proportional to $\sqrt{r}$. Furthermore, the strain is given by

$$\frac{\partial u}{\partial x} = \frac{\partial u}{\partial \eta}\frac{\partial \eta}{\partial x} \tag{10.12}$$

where from Eqs. (10.8) and (10.10),

$$\frac{\partial x}{\partial \eta} = 0.5(1+\eta)L = \sqrt{r}\sqrt{L} \tag{10.13}$$

Thus, by using Eqs. (10.9), (10.12) and (10.13),

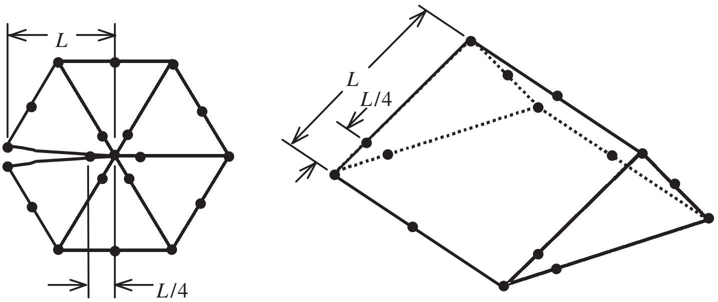
$$\frac{\partial u}{\partial x} = \frac{1}{\sqrt{r}}\frac{1}{\sqrt{L}}\left[-2\eta u_2 + \left(\frac{1}{2}+\eta\right)u_3\right] \tag{10.14}$$

It is noted that the strain given by Eq. (10.14) is inversely proportional to $\sqrt{r}$, and since the stress is directly proportional to the strain, this can also be said for the stress. Therefore, it can be seen that by shifting the middle node, 2, to the quarter position, we are able to obtain an approximation that follows the behaviour of the stresses and displacements near the crack tip, as predicted by fracture mechanics. Similar procedures can be applied to the other side consisting of nodes 1, 7 and 8.

Other types of crack tip elements with different shapes can also be obtained, and some examples are shown in Figure 10.4.

10.3 METHODS FOR INFINITE DOMAINS

There are many problems in real life that actually involve an infinite or semi-infinite domain. For example, the radiation of heat from a point source into space, and the propagation of waves on the surface of the ground and under the ocean, and so on. For the above problems, the strength of the heat radiation and the amplitude of the waves vanish at infinity. So far, the finite element method we have discussed in this book all comes with a finite boundary.



Triangular crack tip elements

A 3D, wedge crack tip element

Figure 10.4. Examples of crack tip elements.

In fact, all elements introduced are with closed boundaries. Therefore, if these elements are used to model an infinite domain, then the boundary will affect the results obtained. For the propagation of waves, any finite boundary will reflect the waves, and this will result in the superposition of the transmitted and reflected waves. Similarly, for other problems, approximations using conventional finite elements will thus be inaccurate. Intuitively, one might think that one solution to modelling the infinite or semi-infinite domain is to place the finite boundary far away from the area of interest. The question of 'how far' is far enough will then set in, and besides, this method would usually require an excessively large number of elements to model regions that the analyst has little interest in. To overcome such difficulties caused by an infinite or semi-infinite domain, many methods have been proposed, of which one of the most effective and efficient is the use of *infinite elements*.

10.3.1 Infinite Element Formulated by Mapping (Bettess, 1992)

An infinite element is created by using shape functions to approximate a sequence of the decaying form:

$$\frac{C_1}{r} + \frac{C_2}{r^2} + \frac{C_3}{r^3} + \cdots \tag{10.15}$$

where C_i are arbitrary constants and r is the radial distance from the origin or *pole*, which can be arbitrarily fixed. Consider the 1D mapping of the line OPQ, which coincides with the x-axis, as shown in Figure 10.5. Like the finite element formulation for all isoparametric elements, the coordinates are interpolated from the nodal coordinates, thus let us propose that

$$x = -\frac{\xi}{1-\xi} x_O + \left(1 + \frac{\xi}{1-\xi}\right) x_Q \tag{10.16}$$

From Eq. (10.16), it can be observed that $\xi = 0$ corresponds to $x = x_Q$, $\xi = 1$ corresponds to $x = \propto$, and $\xi = -1$ corresponds to $x = (x_Q - x_O)/2 = x_P$. As mentioned, r is the

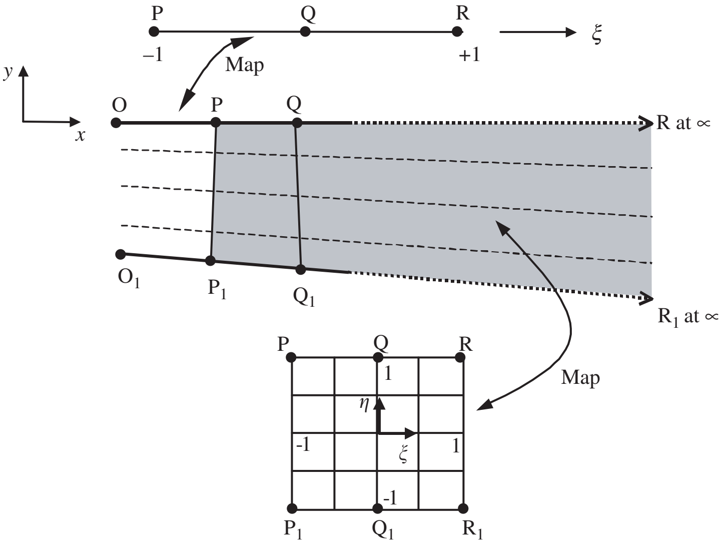


Figure 10.5. Infinite line and 2D element mapping.

distance measured from the origin or pole, and we assume that the pole is at O. Therefore,

$$r = x - x_O \tag{10.17}$$

Solving Eq. (10.16) for ξ would give

$$\xi = 1 - \frac{x_Q - x_O}{x - x_O} = 1 - \frac{x_Q - x_O}{r} \tag{10.18}$$

If the unknown variable, say u, is approximated by a polynomial function such as

$$u = \alpha_0 + \alpha_1\xi + \alpha_2\xi^2 + \alpha_3\xi^3 + \cdots \tag{10.19}$$

then substituting Eq. (10.18) into (10.19) would give us a series of the form given in Eq. (10.15), with the linear shape function in ξ corresponding to $1/r$ terms, the quadratic shape function to $1/r^2$, and so on.

A generalization to 2D or 3D can be achieved by simple products of the 1D, infinite mapping shown above, with a standard type of shape function in η (and ζ for 3D) direction in the manner shown in Figure 10.5. First, we generalize the interpolation of Eq. (10.16) for any straight line in the x, y and z space:

$$\begin{aligned} x &= -\frac{\xi}{1-\xi}x_{O_1} + \left(1 + \frac{\xi}{1+\xi}\right)x_{Q_1} \\ y &= -\frac{\xi}{1-\xi}y_{O_1} + \left(1 + \frac{\xi}{1+\xi}\right)y_{Q_1} \\ z &= -\frac{\xi}{1-\xi}z_{O_1} + \left(1 + \frac{\xi}{1+\xi}\right)z_{Q_1} \end{aligned} \tag{10.20}$$

Then we complete the interpolation and map the whole domain of $\xi\eta(\zeta)$ by adding a standard interpolation in the $\eta(\zeta)$ directions. Thus, we can write for element PP$_1$QQ$_1$RR$_1$ of Figure 10.5

$$x = N_1(\eta)\left[-\frac{\xi}{1-\xi}x_O\left(1 + \frac{\xi}{1-\xi}\right)x_Q\right] + N_0(\eta)\left(-\frac{\xi}{1-\xi}x_{O_1} + \frac{\xi}{1-\xi}x_{Q_1}\right) \tag{10.21}$$

with

$$N_1(\eta) = \frac{1+\eta}{2}, \quad N_0(\eta) = \frac{1-\eta}{2} \tag{10.22}$$

and map the points as shown. In a similar manner, quadratic interpolations could also be used. These infinite elements can be joined to a standard finite element mesh as shown in Figure 10.6.

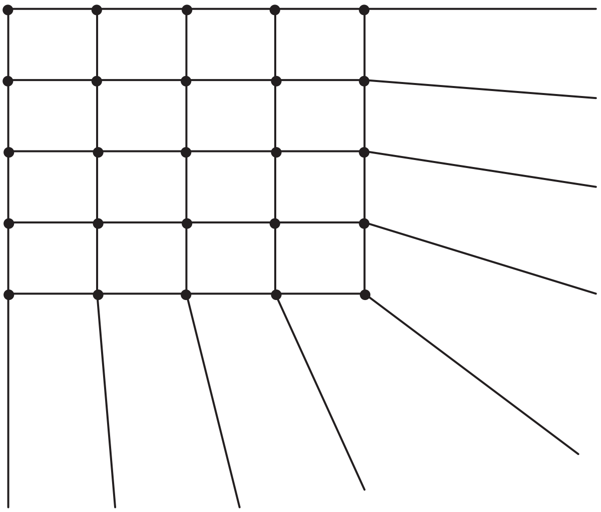


Figure 10.6. Infinite elements attached to standard finite element mesh.

10.3.2 Gradual Damping Elements

Using elements with gradually increased artificial damping elements attached on the regular finite element mesh is a very efficient way to model vibration problems with infinite boundaries. This method was proposed by Liu [1994] and Liu and Quek [2001]. While on the topic of modelling for an infinite domain, it is our intention to demonstrate that there are certain situations where infinite approximations are not easily achieved. One such application is in the study of lamb wave propagation in infinite plates or beams. Lamb waves are dispersive waves that involve multiple characteristic reflections with the top and bottom surface of the plate as it progresses along the plate, as shown in Figure 10.7. Such waves are actually very much more complex than the usual plane or transverse waves. There are, of course, many numerical methods and analytical methods available to solve such problems. The strip element method, introduced at the end of this chapter, can be used effectively for such problems. However, there is the restriction of meshing for irregular geometry, since it involves strip elements. The finite element method is still one of the most versatile methods available for any kind of geometry and applications. The problem is the modelling of the infinite domain for studying the propagation characteristics without interference from reflected waves. It has been proposed to use a gradual damping method to model an infinite plate for such a purpose. This method uses conventional finite elements, and the infinite domain is approximated by adding additional elements with a gradual increase in damping to damp down the amplitude of the propagating waves. Sets of finite elements are attached outside the area of interest of the analyst, as shown in Figure 10.8. The following is a brief description of the method.

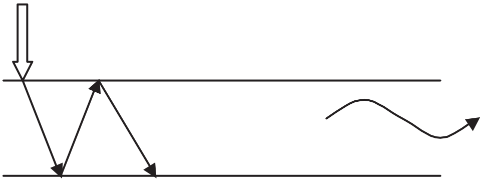


Figure 10.7. Dispersive characteristic of lamb wave propagation.

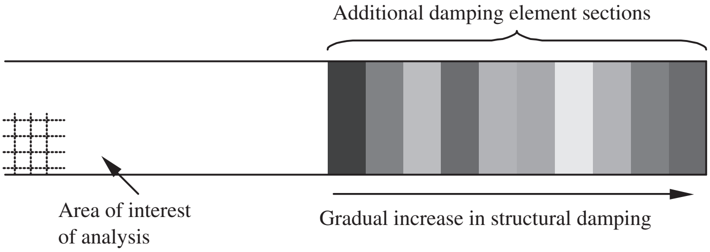

Additional damping element sections
Area of interest
of analysis
Gradual increase in structural damping

Figure 10.8. Damping element sets attached outside area of interest.

Structural damping is considered in the formulation to represent the internal damping of the material as well as in the artificial boundary so as to damp down the wave oscillations. The energy dissipated in one cycle of an oscillation by a viscous damping force is directly proportional to the frequency of the oscillation and the square of the amplitude of vibration, given by

$$W_d = \pi c\omega |u|^2 \tag{10.23}$$

where ω is the angular frequency and c is the damping coefficient. However, the energy dissipated per cycle is independent of the frequency over a wide frequency range for most structural metals. Therefore, we can let

$$c(\omega) = \frac{H}{\omega} \tag{10.24}$$

where H is a damping function, such that the energy dissipated is independent of the angular frequency. The equation of motion for the plate with damping under a harmonic load can then be written as

$$[\mathbf{m}]\{\ddot{\mathbf{u}}\} + [\mathbf{c}]\{\dot{\mathbf{u}}\} + [\mathbf{k}]\{\mathbf{u}\} = \{\mathbf{f}\}\exp(i\omega t) \tag{10.25}$$

where $\mathbf{c}$ is the global matrix of damping coefficients. To create an artificial boundary to damp down the oscillations, a section of elements (outside the area of interest of the analyst) near the finite boundary is first divided into n element sets. The damping coefficient, and hence the damping force defined for each of these sets, is gradually increased from the innermost set to the set next to the finite boundary. For a harmonic force,

$$\begin{aligned}\text{damping force} &= -c\dot{u} = -\frac{H}{\omega}\dot{u} \\ &= -iHu \end{aligned} \tag{10.26}$$

Therefore, Eq. (10.25) can be written as

$$[\mathbf{k} + i\mathbf{H}]\{\mathbf{u}\} + [\mathbf{m}]\{\ddot{\mathbf{u}}\} = \{\mathbf{f}\}\exp(i\omega t) \tag{10.27}$$

The complex matrix $[\mathbf{k} + i\mathbf{H}]$ is known as the complex stiffness, and can be obtained by replacing Young's modulus E by a complex one, $E(1 + i\alpha)$, where α is the material loss factor. By doing so, the complex stiffness matrix can be expressed as

$$[\mathbf{k} + i\mathbf{H}] = [\mathbf{k} + i\alpha\mathbf{k}] \tag{10.28}$$

Hence, from Eqs. (10.28) and (10.24),

$$-[\mathbf{c}]_e\{\dot{\mathbf{u}}\}_e = -\frac{\alpha}{\omega}[\mathbf{k}]_e\{\dot{\mathbf{u}}\}_e = -\alpha[\mathbf{k}]_e\{\mathbf{u}\} \tag{10.29}$$

To gradually increase this damping force, Young's modulus for the kth damping element set added beyond the area of interest can be expressed as

$$E_k = E + i\alpha_0\zeta^k E \quad k = 0, 1, 2, \ldots, n-1 \tag{10.30}$$

where α_0 can be regarded as the initial material loss factor for the artificial damping boundary, and ζ is a constant factor.

This exponential function ensures that the rate of increase in damping is low at the beginning and becomes higher as k increases. This will prevent a sudden increase in the damping that will itself cause reflection of the propagating wave. To determine the value of ζ, which is required to provide sufficient damping, an iterative procedure of increasing ζ is used until the responses obtained for two (or more) cases of different boundary conditions at the ends show no significant differences. This is based on the concept that the damping has done its job such that the effects of the boundary are no longer significant. Hence, the two criteria for achieving the required damping are

1. Sufficient damping such that the effect of the boundary is negligible.
2. Damping is gradual enough such that there is no reflection caused by a sudden damped condition.

This method has been shown to give good approximations for wave propagations in an infinite domain, and the main advantage is the versatility of the finite element method for meshing any complex or irregular geometry. The method can be easily applied using most of the commercial software packages.

10.3.3 Coupling of FEM and BEM

Another effective method of dealing with infinite domains is to use the FEM coupled with the Boundary Element Method (BEM). The FEM is used in the interior portions of the problem domain where the problem is very complex (nonlinear, inhomogeneous, etc.), and the BEM is used for the exterior portion that can extend to infinity. Much research work has been done in this area. An example can be found in Liu [1992] for wave propagation problems.

10.3.4 Coupling of FEM and SEM

Coupling of the FEM with the Strip Element Method (SEM; see Section 10.5) can also effectively handle infinite domains. In such a combination, the FEM is used in the interior portions of the problem domain where the problem is very complex (anisotropy, nonlinear, inhomogeneous, complex geometry, etc.), and the SEM is used for the exterior portion that can extend to infinity. This combination is applicable for domains of anisotropic materials (see Liu [2002]).

10.4 FINITE STRIP ELEMENTS

Using finite strip elements instead of the conventional finite elements can be a very effective method for solving structural problems involving regular geometry and simple boundary conditions. This method was developed by Y. K. Cheung in 1968. In his method, the structure is divided into 2D strip or 3D prism or layer sub-domains. This method usually requires the geometry of the structure to be constant along one or two coordinate axes so

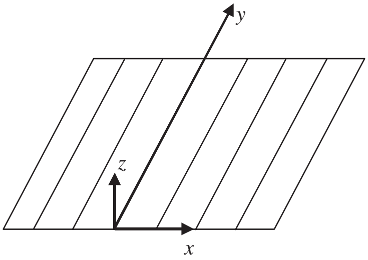


Figure 10.9. Finite strip elements used in a plate.

that the width of the strip or the cross-section of the prism or layer will not change from one end to the other.

Consider the plate modelled with finite strip elements shown in Figure 10.9. It is assumed that the strips are connected to each other along a discrete number of nodal lines that coincide with the longitudinal boundaries of the strip. The finite strip method can be considered as a special form of the finite element procedure using the displacement approach. The standard finite element approach normally uses polynomial shape functions in all directions, but the finite strip method only uses simple polynomials in some directions, and continuously differentiable smooth series in the other directions. The general form of the displacement function for the finite strip method is thus given as a product of polynomials and series. Hence, for each strip shown in Figure 10.9, the displacement function is given as

$$w = \sum_{m=1}^{r} f_m(x) Y_m \tag{10.31}$$

where $f_m(x)$ is the polynomial shape function and Y_m is the continuous series that is able to satisfy the boundary conditions at the structure boundary.

The choice of $f_m(x)$ and Y_m in Eq. (10.31) is very important, as it affects the convergence to the correct results. As a rule, the series part, Y_m, should satisfy the end conditions of the strip. For example, for a simply supported plate strip in bending, the displacement function should be able to satisfy the conditions of both deflection, w, and curvature, $\partial^2 w/\partial x^2$, being equal to zero at the two ends. The polynomial part, on the other hand, must also be able to represent a state of constant strain in the x direction, to ensure that the strain will converge towards the true strain distribution as the mesh is further refined. As a whole, the displacement function or shape function must also satisfy the compatibility of displacements along boundaries with neighbouring strips.

As an example, let us consider the plate shown in Figure 10.9 to be simply supported at both ends in the y direction. To satisfy the conditions at the ends would mean satisfying $Y(0) = 0, Y''(0) = 0, Y(a) = 0$ and $Y''(a) = 0$, where a is the length of the strip. A suitable series function would thus be

$$Y_m(y) = \sin\left(\frac{\mu_m y}{a}\right) \qquad \mu_m = \pi, 2\pi, 3\pi, \ldots, m\pi \tag{10.32}$$

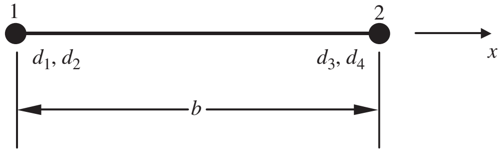


Figure 10.10. A strip element cross-section with two nodes.

As in the finite element method, the choice of shape functions depends upon the number of nodes in the x-direction for each strip element, and also on the nodal degrees of freedom. For example, in the case of a straight line with two nodes, as shown in Figure 10.10, and if the displacements and their first derivatives are the nodal parameters, then the polynomial part of the displacement function can be given as

$$f_m(x) = \begin{bmatrix} C_1 & C_2 & C_3 & C_4 \end{bmatrix} \begin{Bmatrix} d_1^m \\ d_2^m \\ d_3^m \\ d_4^m \end{Bmatrix} \tag{10.33}$$

where d_1 to d_4 are the nodal parameters, and the functions C_i are given as

$$\begin{aligned} C_1(x) &= \frac{2x^3}{b^3} - \frac{3x^2}{b^2} + 1 \\ C_2(x) &= \frac{x^3}{b^2} - \frac{2x^2}{b} + x \\ C_3(x) &= -\frac{2x^3}{b^3} + \frac{3x^2}{b^2} \\ C_4(x) &= \frac{x^3}{b^2} - \frac{x^2}{b} \end{aligned} \tag{10.34}$$

Therefore, if Eq. (10.33) is substituted back into Eq. (10.31), the displacement function can be written as

$$w(x, y) = \sum_{m=1}^{r} \left(C_1(x)d_1^m + C_2(x)d_2^m + C_3(x)d_3^m + C_4(x)d_4^m \right) Y_m(y) \tag{10.35}$$

or

$$w(x, y) = \sum_{m=1}^{r} \begin{bmatrix} N_1^m & N_2^m & N_3^m & N_4^m \end{bmatrix} \begin{Bmatrix} d_1^m \\ d_2^m \\ d_3^m \\ d_4^m \end{Bmatrix} \tag{10.36}$$

where

$$N_i^m(x, y) = C_i(x)Y_m(y) \quad i = 1, 2, 3, 4 \tag{10.37}$$

are the shape functions for the stated example above. The shape functions would therefore vary from problem to problem with different choices of both the polynomial part and the continuous series part. Once the shape function has been formulated, the remaining procedure is actually similar to that of using the finite element method. Based on the constitutive equations and the variational principles discussed when formulating the finite element equations, the corresponding stiffness matrices and load vectors can be similarly obtained. This is then followed by assembly of these matrices for different finite strip elements to form the global matrices. Finally, the matrix equation can then be solved using standard matrix solution techniques. The size of the matrices obtained is usually smaller as compared to that using the conventional finite element method, and this makes the solving of the equations a relatively easier task. The above procedures are applied to a 2D plate modelled with strip elements. A similar approach can also be used when formulating the shape functions, and hence the element matrices for 3D prisms or layers.

10.5 STRIP ELEMENT METHOD (SEM)

The Strip Element Method (SEM) was proposed by Kausel and Roesset (1977) and Tassoulas and Kausel (1983) for solids of isotropic materials and Liu and co-workers [Liu *et al.*, 1994, 1995; Liu and Xi, 2001] for solids of anisotropic materials. The SEM is a semianalytic method for stress analysis of solids and structures. It has been mainly applied for solving wave propagating in composite laminates. The SEM is a semi-exact method that discretizes the problem domain in one or two directions. Polynomial shape functions are then used in these directions, together with the weak forms of the system equation, to produce a set of dimension-reduced special differential equations. These differential equations are then solved analytically. The dimension of the final discretized system equations would be therefore reduced by one order. Details can be found in a monograph by Liu and Xi [2001]. Due to the semianalytic nature of the SEM, it is applicable for problems of arbitrary boundary conditions, including the infinite boundary conditions.

The coupling of the SEM and FEM has also been proposed by Liu [2002] for wave scattering problems in composites. In such a combination, the FEM is used for small domains of complex geometry, and the SEM is used for bulky domains of regular geometry.

11

MODELLING TECHNIQUES

11.1 INTRODUCTION

In this chapter, various modelling techniques will be introduced. Many of the materials are from NAFEMS (1986). Some of these techniques are a must when carrying out finite element analysis to ensure the reliability and accuracy of the results obtained. With developments in computer hardware and software, a FEM analysis can now be performed very easily. Therefore, FEM packages are very often used as a 'black box' for many actual design projects by analysts who may not have a proper background in finite element analysis. However, improper use of commercial software can lead to erroneous results, often hidden behind colourful stress plots or other post-processed results without the knowledge of the analyst. Having described the theories and procedures of the FEM, readers should have quite a good idea on what is really going on in a commercial FE software package. The primary objective of this chapter is therefore to throw some additional light into the black box, so that readers can avoid unnecessary mistakes in creating a FEM model when using a commercial package.

Another reason for learning some of these modelling techniques is to improve efficiency in computing the finite element results, as well as the accuracy of the results. An experienced analyst should be able to obtain accurate results with as little effort in modelling and computer resources as possible. The efficiency of the FE analysis is measured by the effort to accuracy ratio, as shown in Figure 11.1. For example, the use of a symmetrical model to simulate a problem with symmetrical geometry can greatly reduce the modelling and computation time with even more accurate numerical results. Therefore, a good analysis requires more than just meshing up the problem domain with elements. To come up with a good finite element model, the following factors need to be considered:

- Computational and manpower resources that limit the scale of the FEM model.
- Requirement on results that defines the purpose and hence the methods of analysis.
- Mechanical characteristics of the geometry of the problem domain that determine the type of elements to use.
- Boundary conditions.
- Loading and initial conditions.

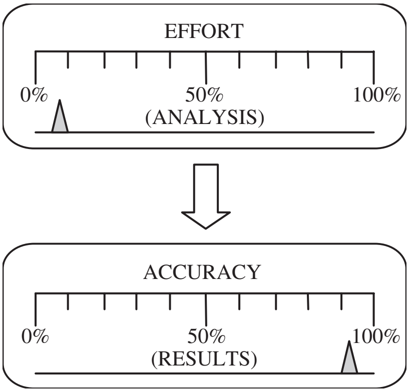


Figure 11.1. Minimum effort to yield maximum accuracy.

11.2 CPU TIME ESTIMATION

Despite advances in the computer industry, computer resources can still be one of the decisive factors on how complex a finite element model can be built. The CPU time required for a static analysis can be roughly estimated using the following simple relation (called the *complexity* of a linear algebraic system):

$$t_{\mathrm{CPU}} \propto n_{\mathrm{dof}}^{\alpha} \tag{11.1}$$

where n_{dof} is the number of total degrees of freedom in the finite element equation system, and α is a constant in the range of 2.0 to 3.0, depending on the different solvers used in the FEM package and the structure of the stiffness matrix.

One of the very important factors that affect α is the bandwidth of the stiffness matrix, as illustrated in Figure 11.2. A smaller bandwidth leads to a smaller value of α, and hence a faster computation. From the direct assembly procedure described in Example 4.2, it is clear that bandwidth depends upon the difference in the global node number assigned to the elements. The element that has the biggest difference in nodal number controls the bandwidth of the global stiffness matrix. The bandwidth can be changed even for the same FEM model by changing the global numbering of the nodes. Therefore, tools have been developed for minimizing the bandwidth through a re-numbering of nodes. Most FEM packages are equipped with one or more such tools. All the user needs to do is use the tool to minimize the bandwidth after meshing the problem domain. This simple operation can sometimes drastically reduce the CPU time. A very simple method for minimizing the difference of nodal numbers, and hence the bandwidth, can be found in Liu [2002].

Equation (11.1) clearly indicates that a finer mesh with a large number of Degrees Of Freedom (DOFs) results in an exponentially increasing computational time. This implies the importance of reducing the DOFs. Many techniques discussed in this chapter are related

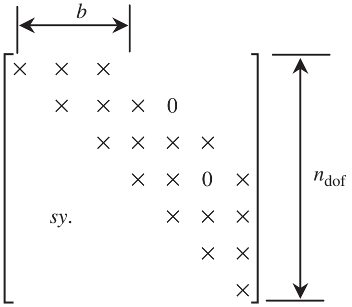


Figure 11.2. Schematic of the structure of the stiffness matrix.

to the reduction of DOFs. Our aims are

1. to create an FEM model with minimum DOFs by using elements of as low a dimension as possible, and
2. to use as coarse a mesh as possible, and use fine meshes only for important areas. These have to be done without sacrificing any accuracy in the results.

11.3 GEOMETRY MODELLING

Actual structures are usually very complex. The analyst should decide on how, where possible, to reduce a complex geometry to a manageable one. The first issue the analyst needs to consider is what type of elements should be used: 3D elements? 2D (2D solids, plates and shells) elements? Or 1D (truss and beam) elements? This requires a good understanding of the mechanics of the problem. As mentioned in Chapter 9, 3D elements can be used for modelling all types of structures. However, it can be extremely expensive if 3D elements are used everywhere in the entire problem domain, because it will definitely lead to a huge number of DOFs. Therefore, for complex problems, the mesh is often a combination of different types of elements created by taking full geometrical advantage of the problem domain. The analyst should analyse the problem in hand, examine the geometry of the problem domain, and try to make use of 2D and 1D elements for areas or parts of the structure that satisfy the assumptions which lead to the formulation of 2D or 1D elements. Usually, 2D elements should be used for areas/parts that have a plate- or shell-like geometry, and 1D elements should be used for areas/parts that have a bar- or arch-like geometry. 3D elements are only used for bulky parts of the structure to which 2D or 1D elements cannot apply. This process is very important, because the use of 2D and 1D elements can drastically reduce the DOFs.

As shown in Figure 11.3, in modelling the geometry for areas or parts where 3D elements are to be used, 3D objects that have the same geometrical shapes as the structure have to be created. For areas or parts where 2D elements are to be used, only the neutral surfaces that are often the geometrical mid surfaces need to be created. For areas or parts where 1D

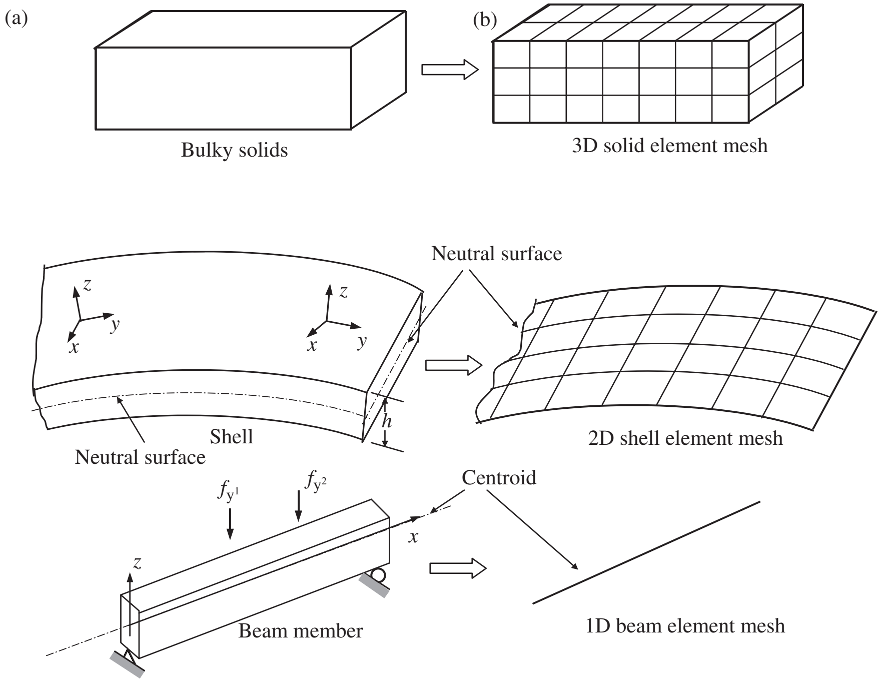


Figure 11.3. Geometrical modelling. (a) Physical geometry of the structural parts; (b) geometry created in FEM models.

elements are to be used, only the neutral axes that are often the geometrical mid axes need to be created. Therefore, the additional advantage of using 2D and 1D elements is that the task of creating geometry is drastically reduced.

At the interfaces between different types of elements, techniques of modelling joints can be used, which will be discussed in detail in Section 11.9. These techniques are required because the type of DOFs at a node is different for different types of elements, due to the difference in theories of mechanics discussed in Chapter 2. Table 11.1 lists the number of DOFs for some different types of elements.

The required result is another important factor when it comes to the creation of the problem domain. For example, analysts will usually give a detailed modelling of the geometry for areas where critical results are expected. Note that many structures are now designed using Computer Aided Design (CAD) packages. Therefore, the geometry of the structure would already have been created electronically. Most commercial preprocessors of FEM software packages can read certain formats of CAD files. Making use of these files can reduce the effort in creating the geometry of the structure, but it requires a certain amount of effort to modify the CAD geometry to be suitable for FEM meshing. There is also ongoing research activity to automatically convert proper 3D geometries into 2D and 1D geometry for a FEM mesh, but to-date there is no such commercial package available.

Table 11.1. Type of elements and number of DOFs at a node

No.	Description	DOFs at a node
1	2D frame analysis (using 2D frame element)	3 (2 translations and 1 rotation)
2	3D frame analysis (using 3D frame element)	6 (3 translations and 3 rotations)
3	2D analysis for plane strain or plane stress analysis	2 (translational displacements)
4	3D analysis for solids with general geometries and loading conditions	3 (translational displacements)
5	2D analysis for axisymmetric solids with axisymmetric or asymmetric loading	2 (translational displacements)
6	plate bending analysis for out-of-plane loading (bending effects only)	3 (1 translation and 2 rotations)
7	general plate and assembled plate analysis with general loading conditions (combined membrane and bending effects)	5 or 6 (3 translations and 2 or 3 rotations)
8	general shell analysis for shell structures (coupled membrane and bending effects)	5 or 6 (3 translations and 2 or 3 rotations)
9	1D analysis for axisymmetric shells with axisymmetric loading (membrane and bending effects)	3 (2 translations and 1 rotation)

11.4 MESHING

11.4.1 Mesh Density

To minimize the DOFs, we often create a mesh of varying density. The mesh only needs to be finer in areas of importance, such as areas of interest, and expected zones of stress concentration, such as at re-entrant corners, holes; slots; notches; or cracks. An example of a finite element mesh exhibiting mesh density transition is shown in Figure 11.4. In this example of the sprocket-chain system, the focus of the analysis is the contact forces between the sprocket and the chain. Hence, the region at the centre of the sprocket is actually not that critical, and the mesh used at that region is relatively coarse.

In using FEM packages, control of the mesh density is often performed by using so-called mesh seeds. The mesh seeds are created before meshing after the geometry has been created. All the user needs to do is place denser mesh seeds in the areas of importance.

11.4.2 Element Distortion

It is not always possible to have regularly shaped elements for irregular geometries. Irregular or *distorted* elements are acceptable in the FEM, but there are limitations, and one needs to control the degree of element distortion in the process of mesh generation. The distortions are measured against the *basic shape* of the element, which are

- Square ⇒ Quadrilateral elements
- Isosceles triangle ⇒ Triangle elements

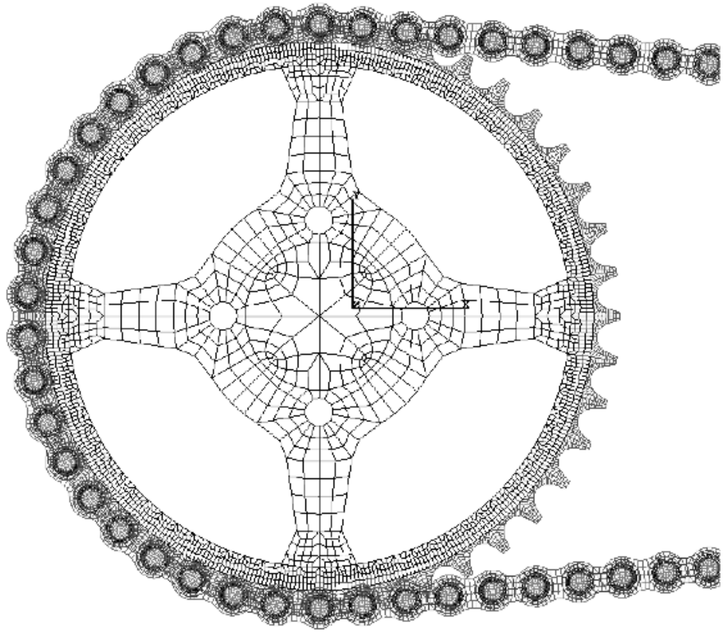
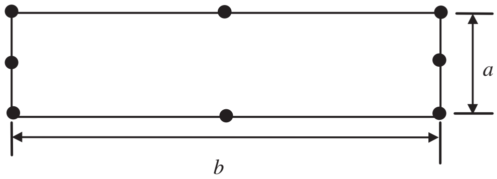


Figure 11.4. Finite element mesh for a sprocket-chain system (Courtesy of the Institute of High Performance Computing and SunStar Logistics(s) Pte Ltd(s)).

- Cube ⇒ Hexahedron elements
- Isosceles tetrahedron ⇒ Tetrahedron elements

Five possible forms of element distortions and their rough limits are listed as follows:

1. *Aspect ratio* distortion (elongation of element) (Figure 11.5).
2. *Angular* distortion of the element (Figure 11.6), where any included angle between edges approaches either 0° or 180° (skew and taper).
3. *Curvature* distortion of element (Figure 11.7), where the straight edges from the element are distorted into curves when matching the nodes to the geometric points.
4. *Volumetric* distortion occurs in concave elements. As discussed in Chapter 6, in calculating the element stiffness matrix, a mapping is performed in order to transfer the irregular shape of the element in the *physical coordinate system* into a regular one in the non-dimensional *natural coordinate system*. For concave elements, there are areas outside the elements (see the shadowed area in Figure 11.8) that will be transformed into an internal area in the natural coordinate system. The element volume integration for the shadowed area based on the natural coordinate system will thus result in a negative value. A few unacceptable shapes of quadrilateral elements are shown in Figure 11.9.



$$\frac{b}{a} \leq \begin{cases} 3 & \text{for stress analysis} \\ 10 & \text{for displacement analysis} \end{cases}$$

Figure 11.5. Aspect distortion.

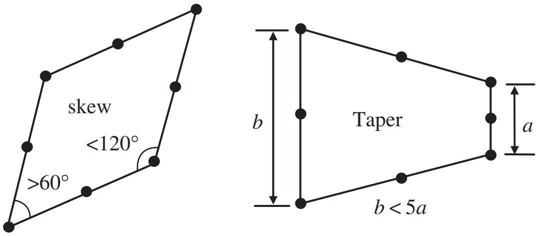


Figure 11.6. Angular distortion.

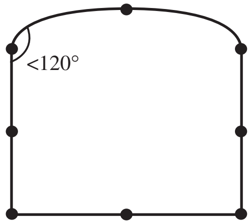


Figure 11.7. Curvature distortion.

5. *Mid-node position* distortion occurs with higher order elements where there are mid nodes. The mid node should be placed as close as possible to the middle of the element edge. The limit for mid-node displacement away from the middle edge of the element is a quarter of the element edge, as shown in Figure 11.10. The reason is that this shifting of mid nodes can result in a singular stress field in the elements, as discussed in Section 10.2.

Many FEM package preprocessors provide a tool for analysing the element distortion rate for a created mesh. All the user needs to do is invoke the tool after the mesh has been created before submitting it for analysis. A report of the distortion rates will be generated for the analyst's examination.

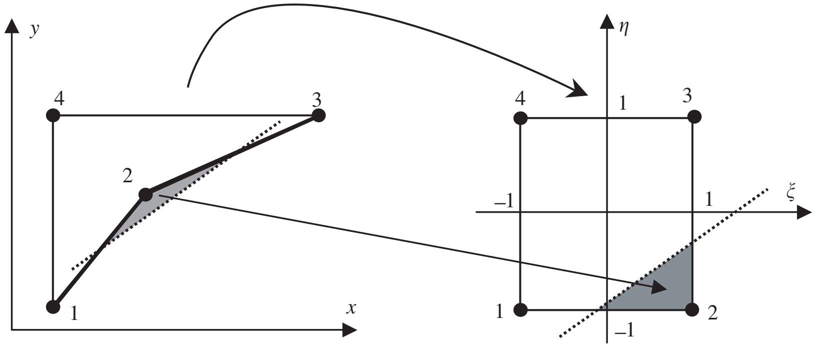


Figure 11.8. Mapping between the physical coordinate ($x - y$) and the natural coordinate ($\xi - \eta$) for heavily volumetrically distorted elements leads to mapping of an area outside the physical element into an interior area in the natural coordinates.

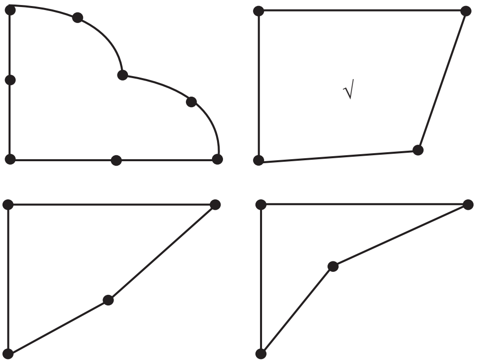


Figure 11.9. Unacceptable shapes of quadrilateral elements.

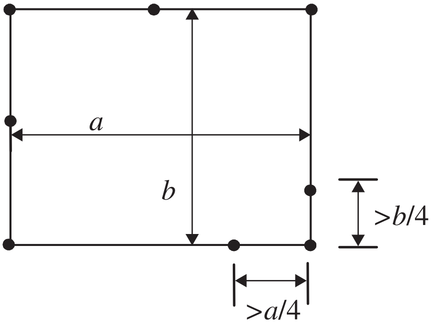


Figure 11.10. The limit for mid-node displacing away from the middle edge of the element.

11.5 MESH COMPATIBILITY

In Chapter 3, when Hamilton's principle is used for deriving the FEM equation, it is required that the displacement has to be admissible, which demands continuity of the displacement field in the entire problem domain. A mesh is said to be compatible if the displacements are continuous along all edges between all the elements in the mesh. The use of different types of elements in the same mesh or improper connection of elements can result in an incompatible mesh. Detailed reasons for mesh incompatibility and methods for fixing or avoiding an incompatible mesh are discussed next.

11.5.1 Different Order of Elements

Mesh incompatibility issues can arise when we have a transition between different mesh densities, or when we have meshes comprised of different element types. When a quadratic element is joined with one or more linear elements, as shown in Figure 11.11, incompatibility arises due to the difference in the orders of shape functions used. The eight-node quadratic element in Figure 11.11 has a quadratic shape function, which implies that the deformation along the edge follows a quadratic function. On the other hand, the linear shape function used in the four-node linear element in Figure 11.11 will result in a linear deformation along each element edge. For the case shown in Figure 11.11(a), the displacement of nodes 1 and 3 for the quadratic element and the linear elements are the same, but deformation of the edges between nodes 1 and 3 will be different. Assuming that nodes 1 and 2 stay still, and node 3 moves a distance, the deformation of these edges is then as shown by the dotted lines in Figure 11.11. A crack-like behaviour is clearly observed, which can lead to severely erroneous results. For the case shown in Figure 11.11(b), the displacements of nodes 1, 2 and 3 for the quadratic element and two linear elements are the same, but deformation of the edges between nodes 1 and 2 and nodes 2 and 3 will be different. If nodes 1 and 3 stay still, and node 2 moves a distance, the deformation of these edges is as shown by the dotted lines in Figure 11.11. Again, a crack-like behaviour is clearly observed.

Solutions for this kind of problem of an incompatible mesh are:

1. Use the same type of elements throughout the entire problem domain. This is the simplest solution and is a usual practice, as complete compatibility is automatically satisfied if the same elements are used as shown in Figure 11.12.

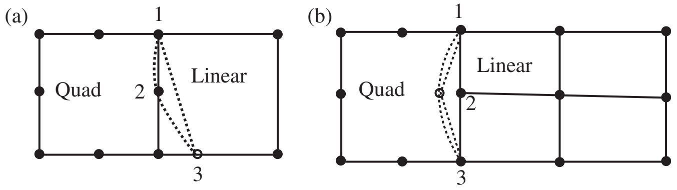


Figure 11.11. Incompatible mesh caused by the different shape functions along a common edge of the quadratic and linear elements. (a) A quadratic element connected to one linear element; (b) a quadratic element connected to two linear elements.

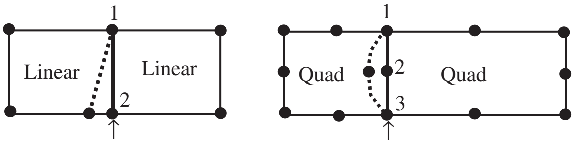


Figure 11.12. Use of elements of the same type with complete edge-to-edge connection automatically ensures mesh compatibility.

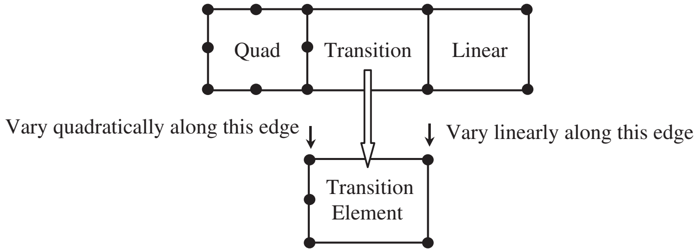

Quad
Transition
Linear
Vary quadratically along this edge
Vary linearly along this edge
Transition
Element

Figure 11.13. A transition element with five nodes used to connect linear and quadratic elements to ensure mesh compatibility.

2. When elements of different orders of shape functions have to be used for some reason, such as in p-adaptive analysis, use *transition elements* whose shape functions have different orders on different edges. An example of a transition element is shown in Figure 11.13. The five-node element shown can behave in a quadratic fashion on the left edge and linearly on the other edges. In this way, the compatibility of the mesh can be guaranteed.
3. Another method used to enforce mesh compatibility is to use *multipoint constraints* (MPC) equations. MPCs can be used to enforce compatibility for the cases shown in Figure 11.11(a). This method is more complicated, and requires the ability to create MPC equations. The use of MPC will be covered in Section 11.10.

11.5.2 Straddling Elements

Straddling elements can also result in mesh incompatibility, as illustrated in Figure 11.14. Although the order of the shape functions of these connected elements is the same, the straddling can result in an incompatible deformation of edges between nodes 1 and 2, and 2 and 3, as indicated by the dotted lines in Figure 11.14. This is because in the assembly of elements, the FEM requires only the continuity of the *displacements* (not the derivatives) at nodes between elements.

The method for fixing the problem of the mesh incompatibility of straddling elements is to make sure that there are no straddling elements in the mesh. Most mesh generators are designed not to produce such an element mesh. However, care needs to be taken in the process of creating a mesh manually.

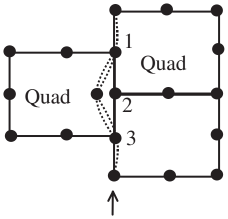


Figure 11.14. Incompatible mesh caused by straddling along the common edge of the same order of elements.

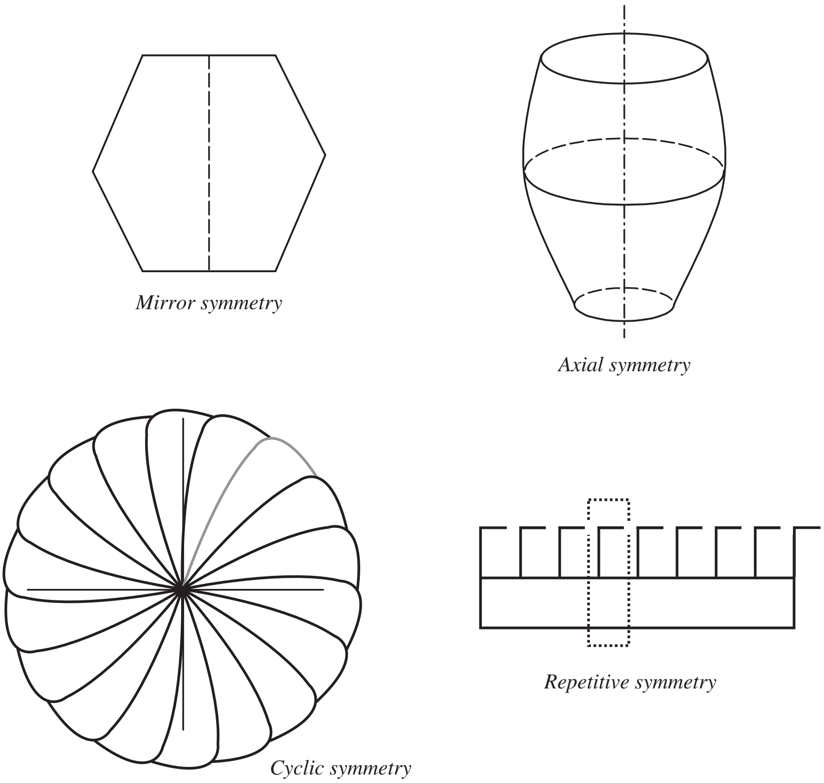
11.6 USE OF SYMMETRY

Many structures and objects exhibit some form of symmetry. Figure 11.15 shows the different types of common structural symmetry. Objects such as a drinks can can exhibit axial symmetry, and even huge structures such as the Eiffel Tower in Paris exhibits mirror symmetry. An experienced analyst will take full advantage of such symmetries in structures to simplify their modelling process, as well as to reduce the DOFs and hence computational time required for the analysis. Imagine a full finite element model of the Eiffel Tower consisting of, say, 100,000 elements. Because of the mirror symmetry, one can actually perform the analysis by modelling just a quarter of the whole structure, and the number of elements will be reduced to just 25,000 elements. The total DOFs of the system will also be reduced to a quarter. Using Eq. (11.1) with $\alpha = 3$, it can be found that the CPU time will be reduced to $(1/4)^3 = (1/64)$th of that required for solving the full model. The significance is astonishing! Furthermore, as only a quarter model is required, the time taken for the analyst to create the model is also reduced. In addition, the accuracy of the analysis can be improved as the equation system becomes much smaller and the numerical error in computation will reduce. However, proper techniques have to be used to make full use of the structural symmetry. This section will deal with some of these techniques.

11.6.1 Mirror Symmetry or Plane Symmetry

Mirror symmetry is the symmetry about a particular plane, and it is the most prevailing case of symmetry. A half of the structure is the mirror image of another. The position of the mirror is called the plane of symmetry. A structure is said to have mirror structural symmetry if there is symmetry of geometry, support conditions and material properties. Many actual structures exhibit this type of symmetry. Some of these structures are actually symmetrical about a particular plane, while others are symmetric with respect to multiple planes. Take, for example, a cubic block as shown in Figure 11.16. One can actually use the property of single-plane-symmetry and model a half model, or one can use that of two-plane-symmetry to further reduce the finite element model to a quarter of the original structure. In fact, more planes of symmetry can also be used to model just a one-eighth model, and in that case, it would be similar to the case of cyclic symmetry, which will be discussed later.

Consider the symmetric 2D solid shown in Figure 11.17. The 2D solid is symmetric with respect to an axis of symmetry of $x = c$. The right half of the domain is modelled with


Mirror symmetry

Axial symmetry

Cyclic symmetry

Repetitive symmetry

Figure 11.15. Different types of structural symmetry.

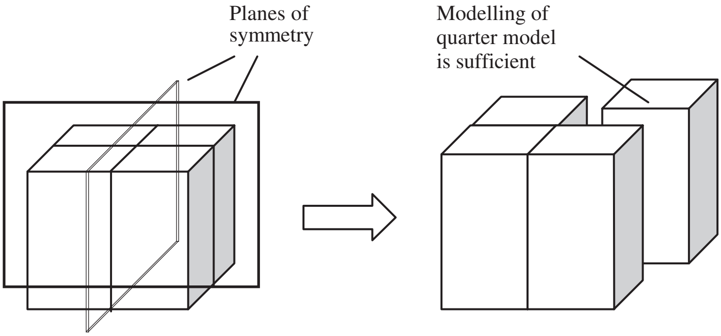


Figure 11.16. Modelling a cubic block with two planes of symmetry.

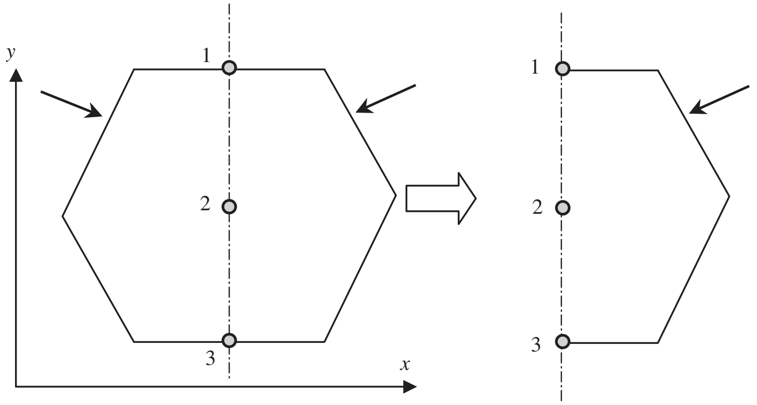


Figure 11.17. 2D solid with an axis of symmetry at $x = c$. The right half of the domain is modelled with impositions of symmetric boundary conditions at nodes on the symmetric axis.

impositions of the following symmetric boundary conditions at nodes on the symmetric axis:

$$\begin{aligned} u_1 &= 0 \\ u_2 &= 0 \\ u_3 &= 0 \end{aligned} \tag{11.2}$$

where u_i ($i = 1, 2, 3$) denotes the displacements in the x-direction at node i. Equation (11.2) gives a set of Single Point Constraint (SPC) equations, because in each equation there is only one unknown (or one DOF) involved. This kind of SPC can be simply imposed by removing a corresponding row and column in the global system equations, as demonstrated in Examples 4.1 and 4.2.

Loading conditions on a symmetrical structure must also be taken into consideration. A loading is considered symmetric if the loading can also be ‘reflected’ off a particular plane, as shown in Figure 11.18. In this case, the problem is symmetric because the whole structure, its support conditions, as well as its loading, is symmetrical about the plane $x = 0$. An analysis of half of the whole beam structure using the symmetrical boundary condition at $x = 0$ would yield as complete a solution as that of the full model with at least less than a quarter of the effort.

A problem can also be anti-symmetric if the structure is symmetric but the loading is anti-symmetric, as shown in Figure 11.19. Again, modelling half of the structure can also yield a complete solution using an *anti-symmetric* boundary condition, which would be different on the plane of symmetry. In the simple example shown in Figure 11.19, the anti-symmetric boundary condition is that the deformation at the plane of symmetry is zero. Note that the rotation at the plane of symmetry need not be zero, in contrast with the case of symmetric loading.

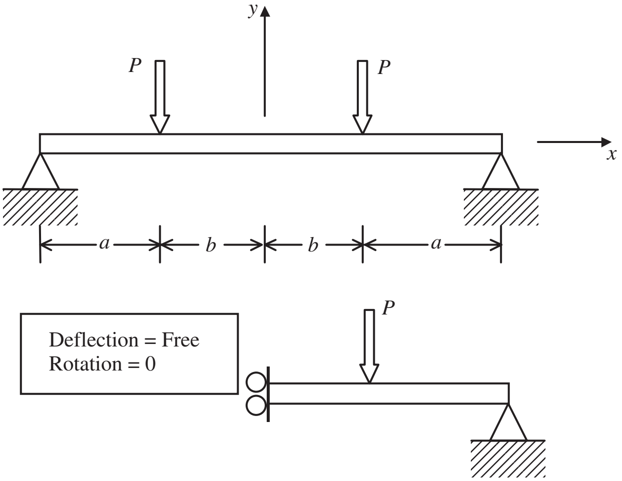


Figure 11.18. Simply supported symmetric beam structure.

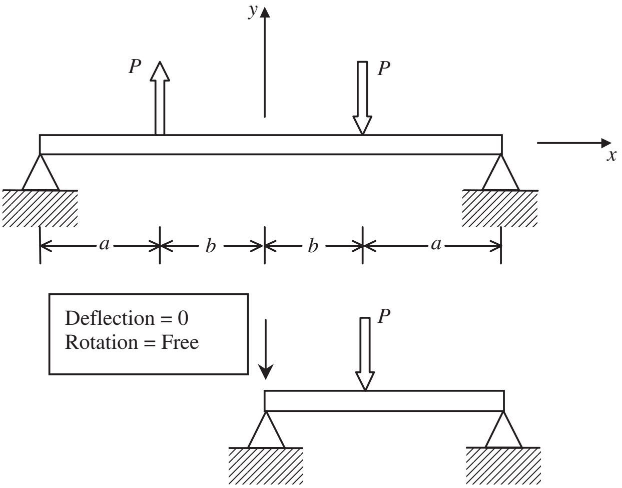


Figure 11.19. Simply supported anti-symmetric beam structure.

The following general rules can be applied when deciding the boundary conditions at the plane of symmetry. If the problem is symmetric, as shown in Figure 11.18, then:

1. There are no translational displacement components normal to the plane of symmetry.
2. There are no rotational displacement components with respect to the axis that is parallel to the plane of symmetry.

Table 11.2. Boundary conditions for symmetric loading

Plane of symmetry	u	v	w	θ_x	θ_y	θ_z
xy	Free	Free	Fix	Fix	Fix	Free
yz	Fix	Free	Free	Free	Fix	Fix
zx	Free	Fix	Free	Fix	Free	Fix

Table 11.3. Boundary conditions for anti-symmetric loading

Plane of symmetry	u	v	w	θ_x	θ_y	θ_z
xy	Fix	Fix	Free	Free	Free	Fix
yz	Free	Fix	Fix	Fix	Free	Free
zx	Fix	Free	Fix	Free	Fix	Free

If the problem is anti-symmetric as shown in Figure 11.19, then:

1. There are no translational displacement components parallel to the plane of symmetry.
2. There are no rotational displacement components with respect to the axis that is normal to the plane of symmetry.

Tables 11.2 and 11.3 give a complete list of conditions for symmetry and anti-symmetry for general three-dimensional cases.

Any load can be decomposed into a symmetric load and an anti-symmetric load, therefore as long as the structure is symmetric (in geometry, material and boundary conditions), one can always take advantage of the symmetry. Consider now a case shown in Figure 11.20(a), where the simply supported beam structure is symmetric structurally, but the loading is asymmetric (neither symmetric nor anti-symmetric). The structure can always be treated as a combination of (a) the same structure with symmetric loading, and (b) the same structure with anti-symmetric loading. In this case, one needs to solve two problems, with each problem having half the number of DOFs if the whole structure is modelled. One of the problems is symmetrical while the other is anti-symmetric.

Figure 11.21 shows a more complex example of how a framework with asymmetric loading conditions can be analysed using half of the framework with symmetric and anti-symmetric conditions. Adding up one problem of the same structure subjected to a symmetric loading with another of the same structure subject to anti-symmetric loading is equivalent to analysing the full frame structure with asymmetric loading. In this example, there is actually a frame member that is at the plane of symmetry. The properties of this frame member on the symmetric plane also need to be halved for the two half models. This means that all the properties which contribute to the stiffness matrix for this member need to be halved. If this is a dynamic analysis, then the density also needs to be halved.

Dynamic problems can also be solved in a similar manner using the symmetric or anti-symmetric properties, for example, if symmetric boundary conditions are imposed

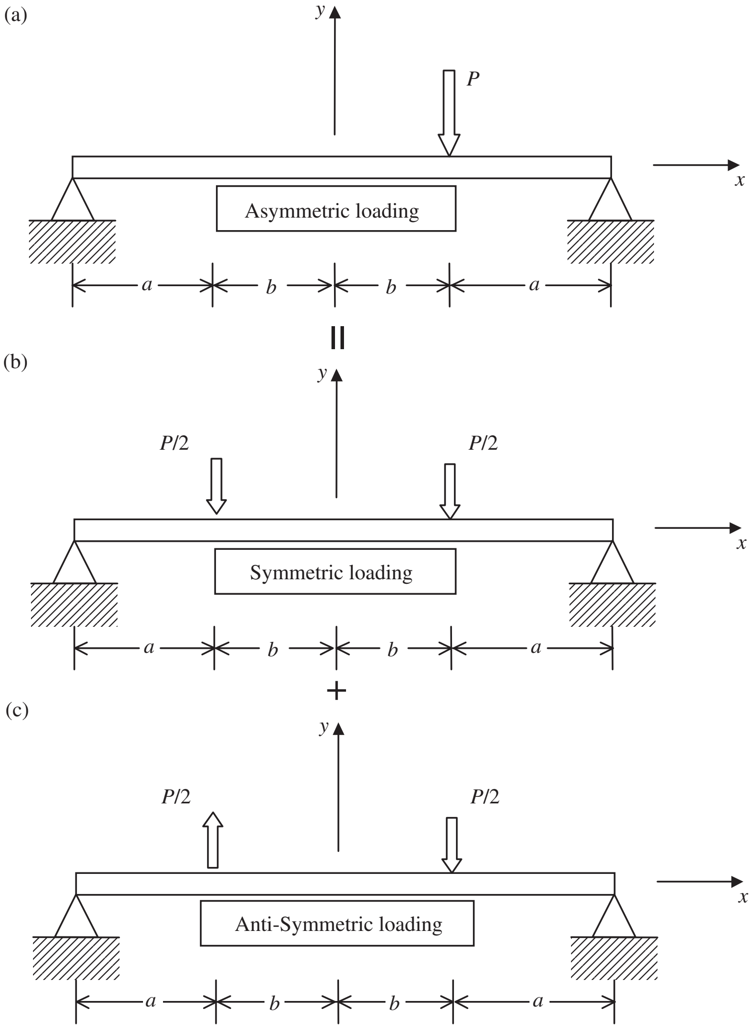


Figure 11.20. Simply supported symmetric beam structure subject to an asymmetric load. (a) A structure with an asymmetric load; (b) the same structure with a symmetric load; (c) the same structure with an anti-symmetric load.

on a simple beam structure and a natural frequency analysis is carried out. The natural frequencies obtained will correspond only to the symmetrical modes. To obtain the anti-symmetrical modes, anti-symmetrical boundary conditions need to apply to the model. Figure 11.22 shows the symmetric and anti-symmetric conditions for vibration analysis in a simply supported beam.

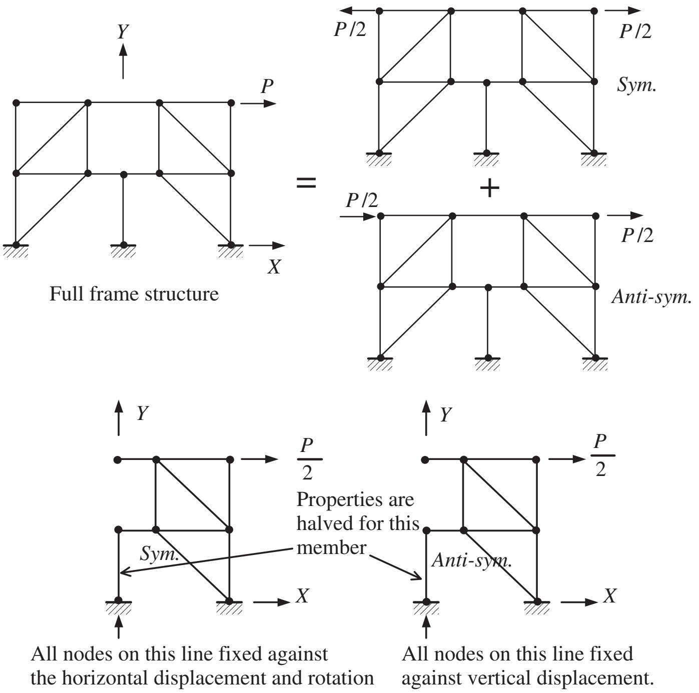

Y
P
X
Full frame structure
=
P/2
P/2
Sym.
+
P/2
P/2
Anti-sym.
Y
P
2
Properties are halved for this member
Sym.
X
All nodes on this line fixed against the horizontal displacement and rotation
Y
P
2
Anti-sym.
X
All nodes on this line fixed against vertical displacement.

Figure 11.21. Using symmetry to analyse symmetrical framework with asymmetric loading.

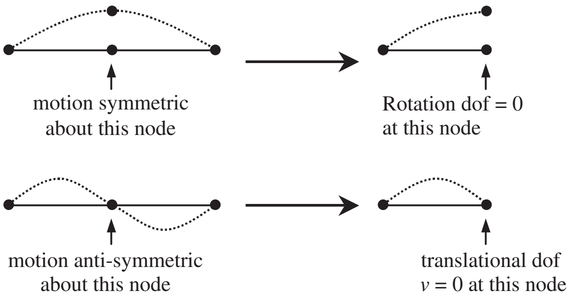

motion symmetric about this node
Rotation dof = 0 at this node
motion anti-symmetric about this node
translational dof $v = 0$ at this node

Figure 11.22. Using symmetric and anti-symmetric conditions for free vibration analysis.

11.6.2 Axial Symmetry

A solid or structure is said to have axial symmetry when the solid can be generated by rotating a planar shape about an axis. Hence, such a solid can be modelled by simply using a special type of 2D or 1D element, called an axisymmetric element. In this way, a 3D solid can be modelled simply by using 1D or 2D elements that will greatly reduce the modelling and computational effort. For example, a cylindrical shell structure can be modelled using

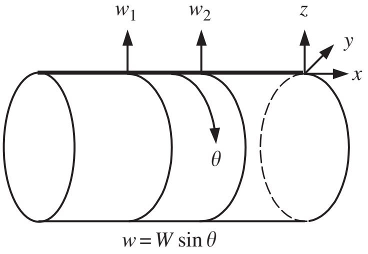


Figure 11.23. A cylindrical shell structure modelled using 1D axisymmetric elements.

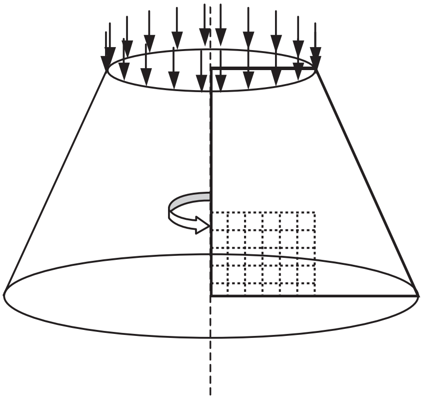


Figure 11.24. A 3D structure modelled using 2D axisymmetric elements.

1D axisymmetric beam elements, as shown in Figure 11.23. Figure 11.24 shows an example of a 3D solid under axially symmetric loads, which can be modelled using 2D axisymmetric elements.

The formulation of 1D or 2D axisymmetric elements is much similar to the 1D or 2D elements developed in earlier chapters, except that all the equations need to be expressed in the polar coordinate system instead of the Cartesian coordination system. The shapes for 2D axisymmetric elements are the same as those in Chapter 7. Generally speaking, the use of axisymmetric elements requires fewer computational resources compared to a full 3D discretization. Axisymmetric elements are readily available in most finite element software packages, and the use of these elements is similar to their counterpart of regular 1D or 2D elements.

Similar to the plane symmetry problems, the loadings applied on an axial symmetric structure do not have to be axial symmetric or axial anti-symmetric. Any axial asymmetric load can be expressed in a Fourier superimposition of both axial symmetric and axial anti-symmetric components in the θ direction (see Figure 11.23). Therefore, the problem can always be decomposed into two sets of axial symmetric and axial anti-symmetric problems, as long as the structure is axial symmetric (in geometry, material and boundary support).

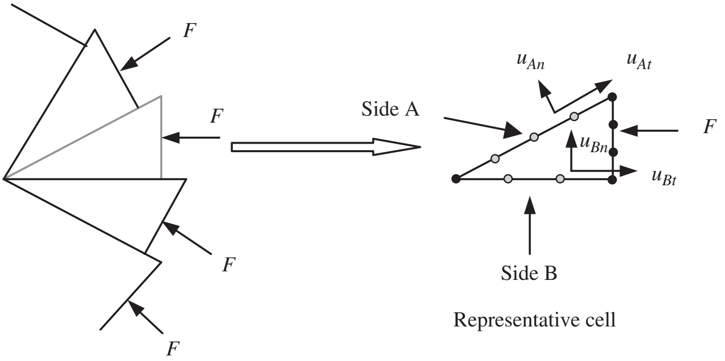


Figure 11.25. Representative cell isolated from a cyclic symmetric structure and the cyclic symmetrical conditions on the cell.

11.6.3 Cyclic Symmetry

Cyclic symmetry prevails in problems where both geometry and loading appear as repeated sectors. In such a case, a complete solution can be obtained by analysing only one sector as a *representative cell* with a set of cyclic boundary conditions on the boundaries of the cell, as shown in Figure 11.25. The cyclic symmetric boundary condition for the problem shown in Figure 11.25 should be that all the variables along side A must match exactly those on side B. Constraint equations at all the corresponding nodes along sides A and B can therefore be written as

$$u_{An} = u_{Bn} \tag{11.3}$$

$$u_{At} = u_{Bt} \tag{11.4}$$

Note that in Eqs. (11.3) and (11.4), both u_{An} and u_{Bn} (or u_{At} and u_{Bt}) are unknowns. Thus, Eqs. (11.3) and (11.4) are constraint equations that involve more than one DOF in one equation. These types of constraint equations are termed *Multi-Point Constraint* (MPC) equations, which have to be imposed by modifying the global system equations. The imposition of MPCs is, however, more tedious than that of SPC, detailed in Chapter 4. In the imposition of SPC, all one need do is remove (or modify) the corresponding rows and column in the system equations (see Examples 4.1 and 4.2). The imposition of MPC requires the use of either penalty method or the method of Lagrange multipliers, that are detailed in Section 11.11.

11.6.4 Repetitive Symmetry

Repetitive symmetry prevails in structures consisting of continuously repeating sections under certain loading conditions (usually in the direction of a repeating section), as shown in Figure 11.26. In such a case, only one section needs to be modelled and analysed. Similar to cyclic symmetry, constraint equations are used for the corresponding nodes at

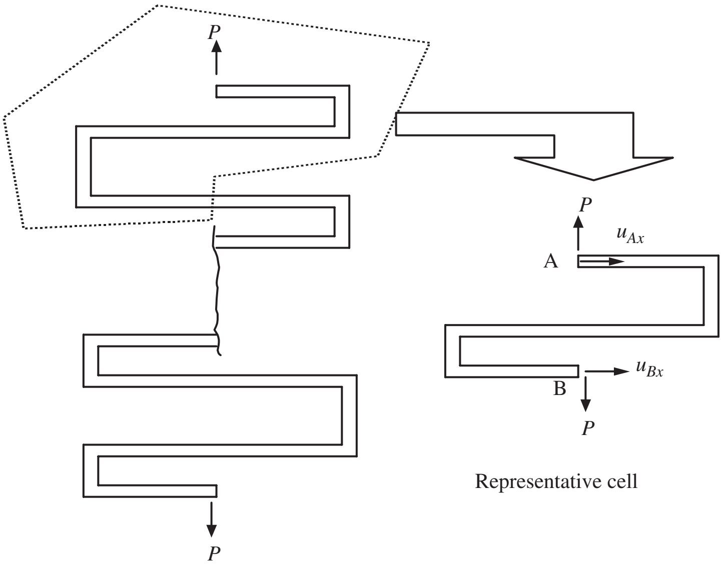


Figure 11.26. Representative cell isolated from a repetitive symmetric structure and the repetitive symmetrical conditions on the cell.

the sectioned surface, such that

$$u_{Ax} = u_{Bx} \tag{11.5}$$

which is again an MPC equation.

11.7 MODELLING OF OFFSETS

11.7.1 Methods for Modelling Offsets

In the modelling of beams, plates and shells, the elements are usually defined on the neutral surface (often the geometric middle surface) of the structure, as shown in Figure 11.3. For elements that are not collinear or coplanar, there will be a distance of *offset* between the nodes in the FEM mode, which are connected together in the physical structure. Figure 11.27 shows a typical case of two beams with different thickness joined at the corner. In the finite element mesh, however, the two corner nodes are apart. In this case, there are two offsets, α and β. In such cases, proper techniques may be a need to model the *offset* in order to simulate the actual connected situation.

If the offsets are too small compared to the length of the beam l, we can often ignore it, and the connection is simply modelled by extending the corner nodes to the joint point (see Figure 11.27). If the offsets are too large, it has to be treated using proper modelling techniques. Whether offsets should be modelled depends upon the engineering judgment

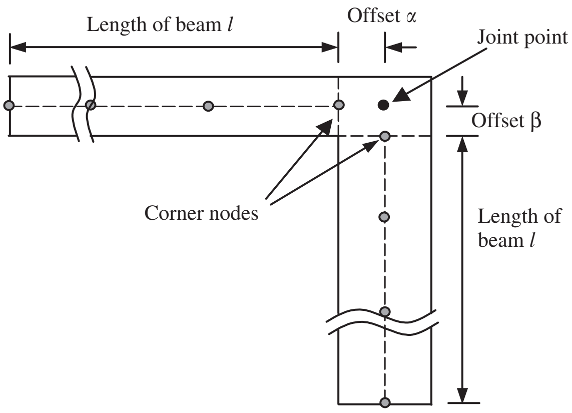


Figure 11.27. Offsets at the joint of two beams with different thickness.

of the analyst. A rough guideline shown below can be followed where l is the length of the beam:

1. $\alpha < l/100$, offset can be safely ignored
2. $l/100 < \alpha < l/5$, offset needs to be modelled
3. $\alpha > l/5$, ordinary beam, plate and shell elements should not be used. Use 2D or 3D solid elements.

Modelling of offsets is usually even more crucial in shells as there are both in-plane and out-of-plane forces in the formulation, and a small variation of nodal distance can give rise to a significant difference in results. There are three methods often used to model the offsets:

1. *Very stiff element*. Use an artificial element with very high stiffness (high Young's modulus, large cross-sectional area or second moment of area) to connect the two corner nodes (see Figure 11.28(a)). This is usually done by increasing the Young's modulus by, say, 10^6 times for the very stiff element. This method is very simple and convenient to use, but is usually not recommended because too large a difference in stiffness among the elements in the same FE model can lead to ill-conditioning of the final set of global stiffness matrices.
2. *Rigid element*. Use an artificial element with a property of a 'rigid element' to connect the corner nodes (see Figure 11.28(b)). This method is good, and is available in many commercial software packages; all the analyst needs to do is create an element in between the two corner nodes and then assign it as a 'rigid element'. The treatment of the rigid element in the software package is the same as the next method using MPC equations, but the formulation and implementation is automated in the software package, so the user does not have to formulate these MPC equations.

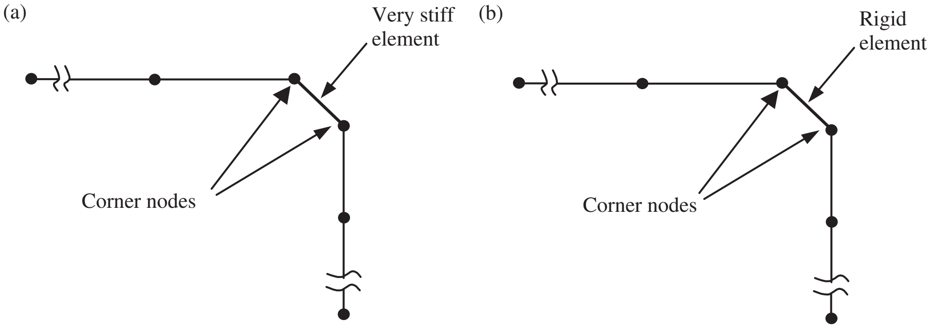


Figure 11.28. Modelling of offsets using an artificial element connecting the two corner nodes. (a) Use of very stiff element; (b) use of rigid element.

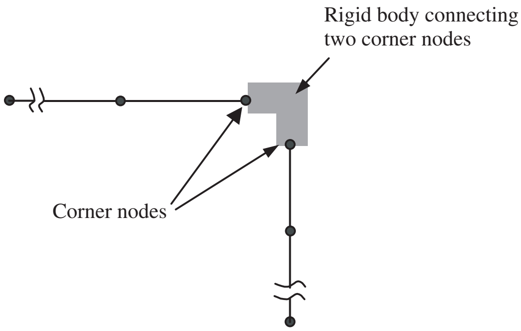


Figure 11.29. Modelling of offsets using MPC equations created using a 'rigid body' connecting two corner nodes.

3. *MPC equations*. If the 'rigid element' is not available in the software package, we then need to create multipoint constraints (MPC) equations to establish the relationship between the DOFs at the two corner nodes. This set of MPC equations is then input in the preprocessor. Use of MPC equations is supported by most FEM software packages. The MPC equations can be created using an artificial rigid body that connects these two corner nodes, as shown in Figure 11.29. The detailed process is discussed in the next subsection.

11.7.2 Creation of MPC Equations for Offsets

The basic idea of using MPC is to create a set of MPC equations that gives the relation between the DOFs of the two separated nodes. It assumes that the two corner nodes are connected by a rigid body. MPC equations are then derived using the simple kinematic relations of the DOFs on the rigid body. The procedure of deriving the MPC equations for the case in Figure 11.30 is described as follows.

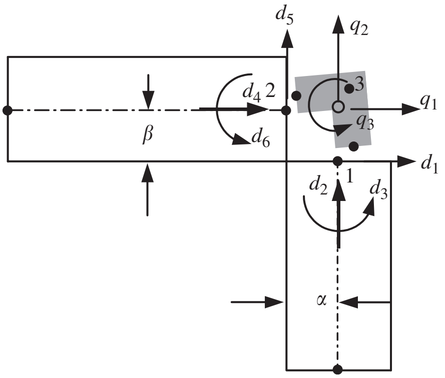


Figure 11.30. A rigid body is used for deriving the MPC equations.

First, assume that nodes 1 and 2 are perfectly connected to the rigid body, and that the rigid body has a movement of q_1, q_2 and q_3 with reference to point 3, as shown in Figure 11.30. Then, enforce points 1 and 2 to follow the rigid body movement, and calculate the resultant displacements at nodes 1 and 2 in terms of q_1, q_2 and q_3, to obtain

$$\begin{aligned} d_1 &= q_1 + \beta q_3 \\ d_2 &= q_2 \\ d_3 &= q_3 \\ d_4 &= q_1 \\ d_5 &= q_2 - \alpha q_3 \\ d_6 &= q_3 \end{aligned} \tag{11.6}$$

Finally, eliminating the DOFs for the rigid body q_1, q_2 and q_3 from the above six equations, we obtain three MPC equations:

$$\begin{aligned} d_1 - \beta d_3 - d_4 &= 0 \\ d_2 - \alpha d_3 - d_5 &= 0 \\ d_3 - d_6 &= 0 \end{aligned} \tag{11.7}$$

which gives the relationship between the six DOFs at nodes 1 and 2.

The number of MPC equations one should have can be determined by the following equation:

$$N\Big|_{\text{Equation of MPC}} = N\Big|_{\substack{\text{DOFs at all nodes to be}\\ \text{connected by the rigid body}}} - N\Big|_{\substack{\text{DOFs of the}\\ \text{rigid body}}} \tag{11.8}$$

Another example of an offset is often seen in structures called stiffened plates and shells, where a beam is fixed to a plate as a stiffener, as shown in Figure 11.31. In this case, the

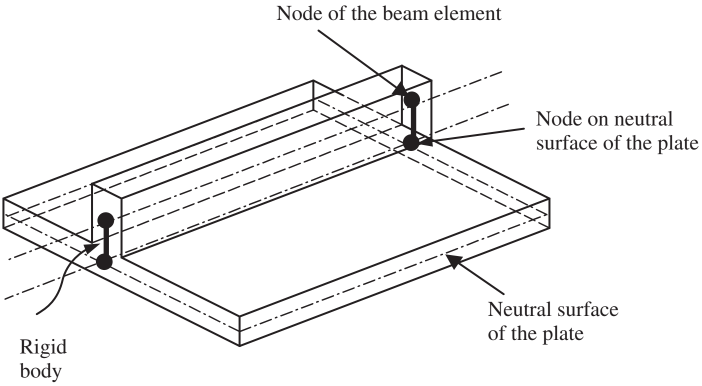

Node of the beam element
Node on neutral surface of the plate
Neutral surface of the plate
Rigid body

Figure 11.31. Stiffened plates with an offset in between the axis of a beam and mid-surface of a plate.

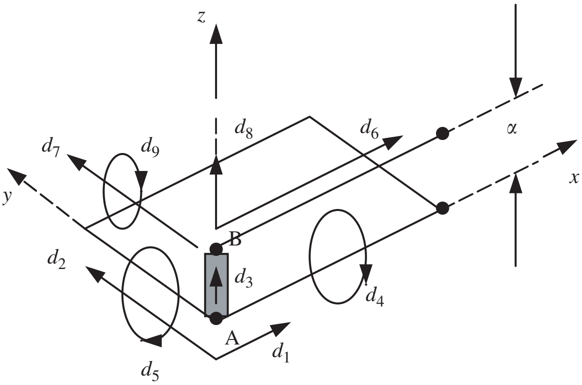


Figure 11.32. DOFs at nodes in the plate and beam.

offset should be modelled if accurate results are required. The offset can be modelled by assuming that the node on the beam is connected to the corresponding node at the plate by a rigid body A–B. For a node in the plate, there are five DOFs (three translational and two rotational about the two axes in the plane of the plate), as shown in Figure 11.32. For a node in the beam there are four DOFs (three translational and one rotational about the axis perpendicular to the bending plane). Note that, in this case, the beam has been defined with its bending plane as the x–z plane. Hence, there are a total of nine DOFs connected by the rigid body linking the node on the beam and the node on the plate. To simplify the process of deriving MPC equations, we let the rigid body follow the movement of node A in the plate. Therefore, the rigid body should have the same number of DOFs as the node in the plate, which is 5. The number of MPC equations should therefore be $9 - 5 = 4$, following Eq. (11.8). As the rigid body follows the movement of A in the plate, what should be done is to force the node belonging to the beam to follow the movement of B on the rigid body.

The resulting four constraint equations should be

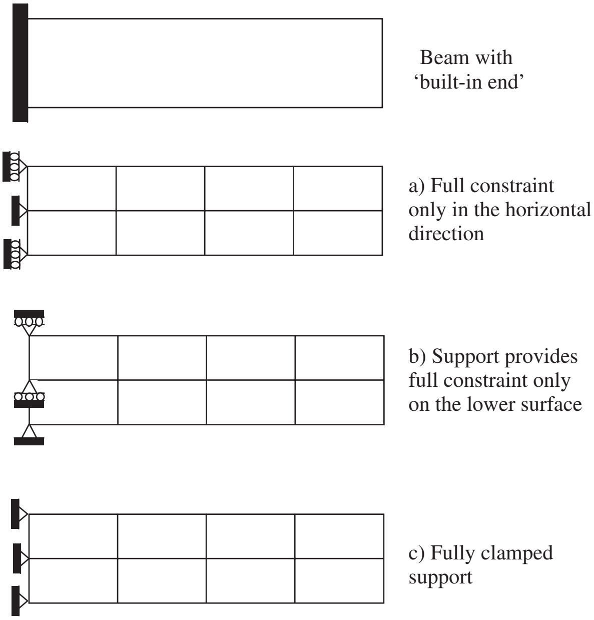
$$\begin{aligned} d_6 &= d_1 + \alpha d_5 \quad \text{or} \quad d_1 + \alpha d_5 - d_6 = 0 \\ d_7 &= d_2 - \alpha d_4 \quad \text{or} \quad d_2 - \alpha d_4 - d_7 = 0 \\ d_8 &= d_3 \quad \text{or} \quad d_3 - d_8 = 0 \\ d_9 &= d_5 \quad \text{or} \quad d_5 - d_9 = 0 \end{aligned} \tag{11.9}$$

The above equations must be specified as constraints for all pairs of nodes along the length of the beam attached on the plate. This ensures that the beam is perfectly attached onto the plate.

11.8 MODELLING OF SUPPORTS

Support of a structure is usually very complex. The FEM allows the constraint to be imposed differently at different nodes. Different methods can be used to simulate these supports. For example, in the analysis of thick beam structures using 2D plane stress elements, there are three possible ways to model the support conditions at the built-in end, as shown in Figure 11.33:

1. Full constraint is imposed at the boundary nodes only in the horizontal direction.
2. Partial full constraint to the nodes on the boundary.
3. Fully clamped boundary conditions are specified at all the boundary nodes.



Beam with 'built-in end'

a) Full constraint only in the horizontal direction

b) Support provides full constraint only on the lower surface

c) Fully clamped support

Figure 11.33. Modelling of built-in end support of beam.

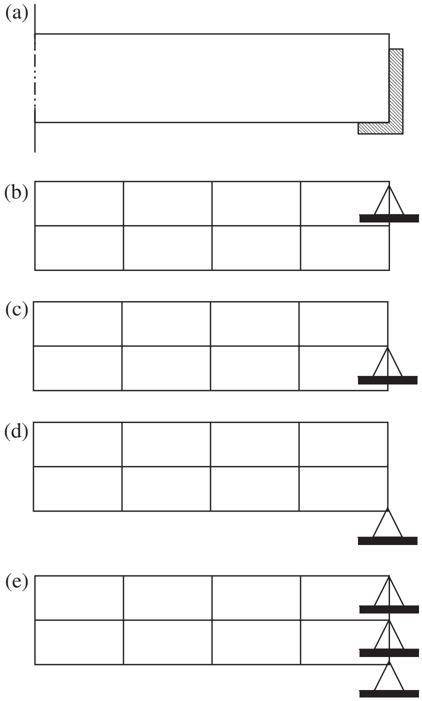


Figure 11.34. Modelling of prop support for a beam.

It is not possible to say which method is the best, because it is actually that which best simulates the situation. Therefore, good engineering judgment, experience and an understanding of the actual situation of support play important roles in creating a good finite element model. Trial and error, parametric studies, and even inverse analysis [Liu *et al.*, 2001] are often conducted when the situation is not very clear to the analyst. Similarly, for a prop support of a beam, there can be more than one way of modelling the support conditions shown in Figure 11.34.

For complex support on structures, it might be necessary to use finer meshes near the boundary. Denser nodes provide more flexibility to model the support, as the FEM allows specification of different constraints at different nodes.

11.9 MODELLING OF JOINTS

Care should be taken if the joints are between different element types. The complication comes from the difference in DOFs at a node in different elements. In many situations, a proper technique or a set of MPC equations is required to model the connection.

Consider a connection between the turbine blade and the turbine disc shown in Figure 11.35(a). A turbine blade is usually *perfectly* connected to a disc. We now need to build a FEM model to analyse the blade–disc system. If both the blade and disc are all

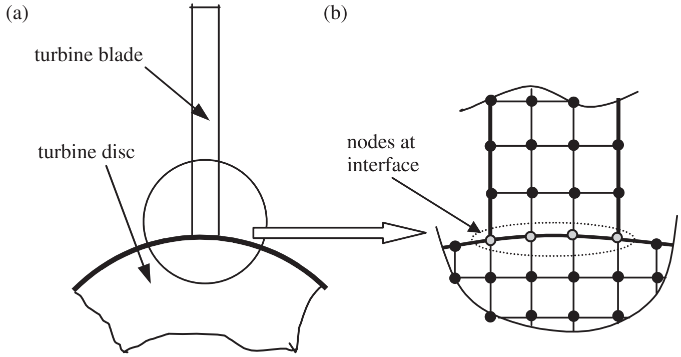


Figure 11.35. Modelling of turbine-blade and turbine-disc system. (a) Simplified diagram of a turbine blade connected to a turbine disc; (b) a magnified 2D solid element mesh at the interface.

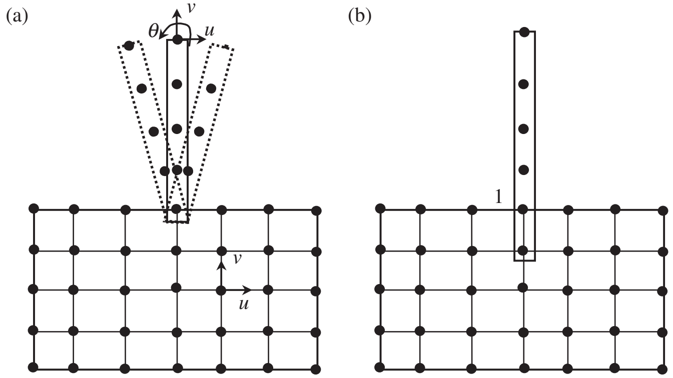


Figure 11.36. Joint between beam and 2D elements. (a) Beam is free to rotate in reference to the 2D solid; (b) perfect connection modelled by artificially extending the beam into a 2D element mesh.

modelled using 2D solid elements, as shown in Figure 11.35(b), the perfect connection is ensured as long as we perform a nodal equivalence operation so that at the interface the blade and disc share the same nodes.

Assume an FEM model where the blade is modelled using beam elements, and the disc is modelled using 2D elements, as shown in Figure 11.36. This model clearly reduces the number of nodes, and thus the DOFs. However, one needs to use a proper technique to model the connection properly, for the reasons given below.

First, we consider the connection of the beam element and 2D solid elements in the way shown in Figure 11.36(a). From Table 11.1, we know that at a node in a 2D solid element there are two DOFs: the translational displacement components u and v. At a node in a beam element, there should be three DOFs: two translational displacement components u and v, and one rotational DOF. Although the beam element and the 2D elements share the common node 1, which ensures that the translational displacements are the same at

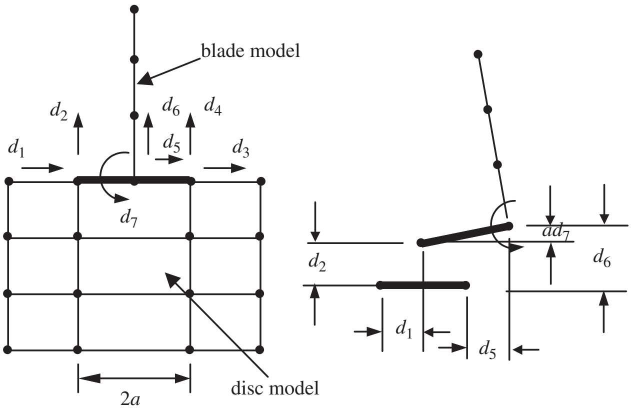


Figure 11.37. Modelling of the turbine blade connected to the disc using a rigid strip to create MPC equations.

the connection point for the beam and 2D solid, the rotation DOF at the beam element is, however, free. Therefore, the blade can rotate freely in reference to the 2D solid, which is not the perfect connection we wanted to model. The problem is the mismatch of the types of DOFs between the beam and 2D solid elements. The rotational DOF cannot be transmitted onto the node on a 2D solid element node, simply because it does not have rotational DOF. Modelling techniques are therefore required to fix this problem.

A simple method to fix the rotation of the beam on the 2D solid mesh is to extend the beam elements (artificially) into the disc for at least two nodes, as shown in Figure 11.36(b). This allows transmission of both translational and rotation deformation between the beam and 2D elements. The drawback of this simple method is the additional mass introduced near the joint area, which may affect the results of a dynamic analysis.

Another effective method is to use MPC equations to create a connection between the two types of elements. The detailed process is given as follows. First, assume that there is a very thin rigid strip connecting the beam and the disc, as shown in Figure 11.37. This strip connects three nodes together, one on the beam and two on the disc. These three nodes have to move together with the rigid strip. The MPC equations can then be established in a manner similar to that discussed for offsets.

Note that a node on the beam has three DOFs, whereas the two nodes on the disc have a total of four DOFs. The total number of DOFs connected to the rigid strip is thus seven. Since the DOF of the rigid strip is three, we should have four MPC equations following Eq. (11.8). The four MPC equations are given as:

$$\begin{aligned} d_1 &= d_5 \\ d_2 &= d_6 - ad_7 \\ d_3 &= d_5 \\ d_4 &= d_6 + ad_7 \end{aligned} \tag{11.10}$$

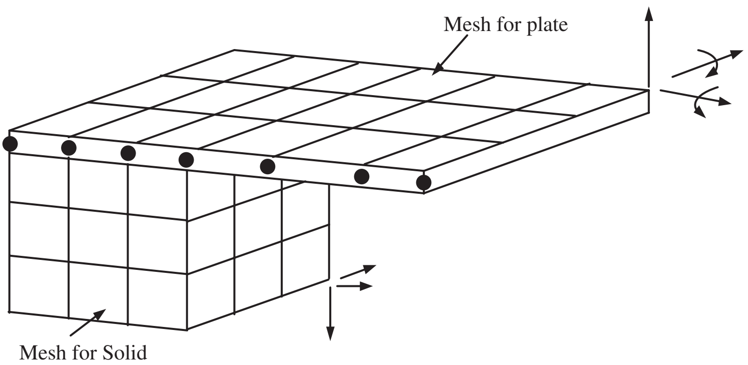


Figure 11.38. Joint between plate and 3D solids modelled by extending plate into a 3D solid element mesh.

These equations enforce full compatibility between the beam and the disc.

A similar problem can occur on the connection between the plate or shell elements that posses rotational DOFs with 3D solid elements that do not possess a rotational DOF. Such a connection between a plate and a 3D solid can be similarly modelled using this method, as shown in Figure 11.38.

11.10 OTHER APPLICATIONS OF MPC EQUATIONS

11.10.1 Modelling of Symmetric Boundary Conditions

When discussing the modelling of symmetric problems, it was mentioned that constraint (boundary) equations are required to ensure that symmetry is properly defined. Note that Eq. (11.2) was obtained when the axis of symmetry is parallel (or perpendicular) to an axis of the Cartesian coordinates. When the axis of symmetry is not parallel (or perpendicular) to the x- or y-axis, as shown in Figure 11.39, the displacement in the normal direction of the axis of symmetry should be zero, i.e.

$$d_n = 0 \tag{11.11}$$

which implies

$$u_i \cos\alpha + v_i \sin\alpha = 0 \quad \text{or} \quad u_i + v_i \tan\alpha = 0 \qquad \text{for } i = 1, 2, 3 \tag{11.12}$$

where u_i and v_i denote the x and y components of displacement at node i. Equation (11.12) is an MPC equation as it involves more than one DOF (u_i and v_i). When $\alpha = 0$, the axis of symmetry is parallel to the y-axis, and Eq. (11.12) will reduce to a SPC that is Eq. (11.2).

11.10.2 Enforcement of Mesh Compatibility

Section 11.5.1 discussed mesh compatibility issues, and it was mentioned that to ensure mesh compatibility on the interface of different types of elements, we can make use of MPC

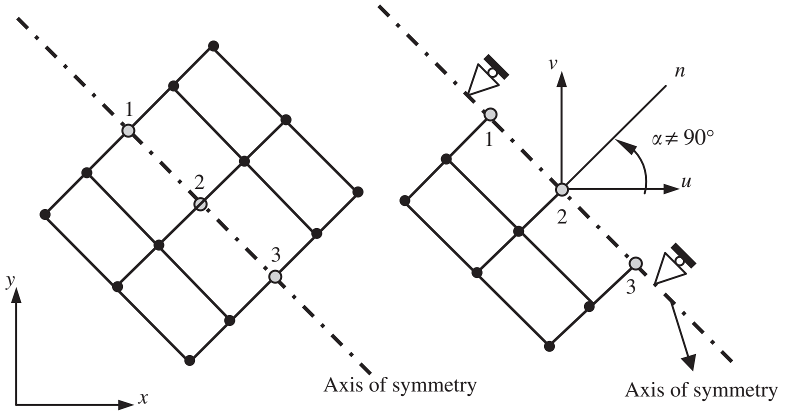


Figure 11.39. Imposing mirror symmetry using MPC when the axis of symmetry is not parallel (or perpendicular) to the *x*- or *y*-axes.

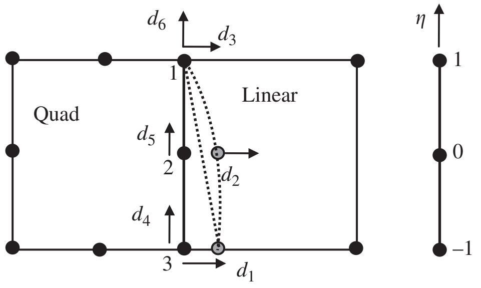


Figure 11.40. Enforcing compatibility of two elements of different orders using MPC.

equations. We now detail the procedure. With reference to Figure 11.40, the procedure of deriving the MPC equations is as follows:

- Use the lower order shape functions to interpolate the displacements on the edge of the element. In this example, the linear shape functions are used, which gives the displacement at any point on the edge 1–3 as

$$d_x = 0.5(1-\eta)d_1 + 0.5(1+\eta)d_3 \tag{11.13}$$

$$d_y = 0.5(1-\eta)d_4 + 0.5(1+\eta)d_6 \tag{11.14}$$

- Next, substitute the value of η at node 2 into the above two equations to get the constraint equations:

$$0.5d_1 - d_2 + 0.5d_3 = 0 \tag{11.15}$$

$$0.5d_4 - d_5 + 0.5d_6 = 0 \tag{11.16}$$

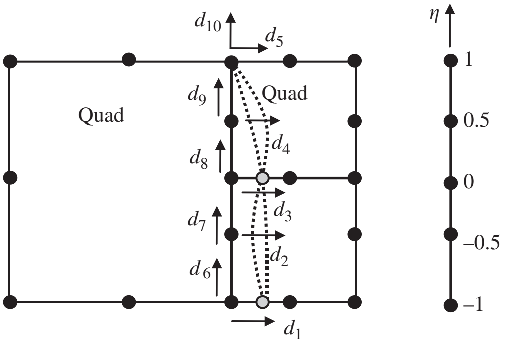


Figure 11.41. Enforcing the compatibility of two different numbers of elements on a common edge using MPC.

These are the two MPC equations for enforcing compatibility on the interface between two different types of elements.

When it is not possible to have gradual mesh density transition and a number of elements share a common edge, as shown in Figure 11.41, mesh compatibility can also be enforced by MPC equations. The procedure is given as follows:

- Use the shape functions of the longer element to interpolate the displacements. Then, for the DOF in the x direction, the quadratic shape function gives

$$d_x = -0.5\eta(1-\eta)d_1 + (1+\eta)(1-\eta)d_3 + 0.5\eta(1+\eta)d_5 \tag{11.17}$$

- Substituting the values of η for the two additional nodes of the elements with shorter edges yields

$$d_2 = 0.25 \times 1.5d_1 + 1.5 \times 0.5d_3 - 0.25 \times 0.5d_5 \tag{11.18}$$

$$d_4 = -0.25 \times 0.5d_1 + 0.5 \times 1.5d_3 + 0.25 \times 1.5d_5 \tag{11.19}$$

Which can be simplified to the following two constraint equations in the x-direction:

$$0.375d_1 - d_2 + 0.75d_3 - 0.125d_5 = 0 \tag{11.20}$$

$$-0.125d_1 + 0.75d_3 - d_4 + 0.375d_5 = 0 \tag{11.21}$$

Similarly, the constraint equations for the displacement in the y direction are obtained as

$$0.375d_6 - d_7 + 0.75d_8 - 0.125d_{10} = 0 \tag{11.22}$$

$$-0.125d_6 + 0.75d_8 - d_9 + 0.375d_{10} = 0 \tag{11.23}$$

Equations (11.20)–(11.23) are a set of MPC equations for enforcing compatibility of the interface between different numbers of elements.

11.10.3 Modelling of Constraints by Rigid Body Attachment

A rigid body attached on an elastic body will impose constraints on it. Figure 11.42 shows a rigid slab sitting on an elastic foundation. Assume that the rigid slab constrains only the vertical movement of the foundation, and it can move freely in the horizontal direction without affecting the foundation. To simulate such an effect of the rigid slab sitting on the foundation, MPC equations are required to enforce the four vertical DOFs on the foundation to follow the two DOFs on the slab.

First, the four equations are:

$$\begin{aligned} d_1 &= q_1 \\ d_2 &= q_1 + q_2 l_1 \\ d_3 &= q_1 + q_2 l_2 \\ d_4 &= q_1 + q_2 l_3 \end{aligned} \tag{11.24}$$

where q_1 and q_2 are, respectively, the translation and rotation of the rigid slab. Then, eliminating the DOFs of the rigid body q_1 and q_2 from the above four equations leads to two MPC equations:

$$(l_2/l_1 - 1)d_1 - (l_2/l_1)d_2 + d_3 = 0 \tag{11.25}$$

$$(l_3/l_1 - 1)d_1 - (l_3/l_1)d_2 + d_4 = 0 \tag{11.26}$$

In the above example, the DOF in the x-direction was not considered, because it is assumed that the slab can move freely in the horizontal direction, and hence it will not provide any constraint to the foundation. If the rigid slab is perfectly connected to the foundation, the constraints on the DOFs in the x-direction must also be considered.

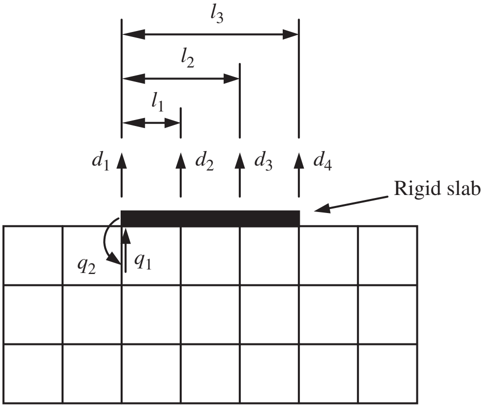


Figure 11.42. Modelling of a rigid slab on an elastic foundation using MPC.

11.11 IMPLEMENTATION OF MPC EQUATIONS

The previous sections have introduced some procedures for deriving MPC equations and some of its applications. However, how are such MPC equations implemented in the process of solving the global FEM system equation? Standard boundary condition SPC equations can be easily implemented as they are concerned with only a single DOF. Many examples of implementing such standard boundary conditions of SPC equations can be found in the examples and case studies of previous chapters. However, implementation of MPC equations normally requires more complex treatments of the global system equations.

Recall that the global system equation for a static system with a DOF of n can be written in the following matrix form:

$$\mathbf{KD} = \mathbf{F} \tag{11.27}$$

where $\mathbf{K}$ is the stiffness matrix of $n \times n$, $\mathbf{D}$ is the displacement vector of $n \times 1$, and $\mathbf{F}$ is the external force vector of $n \times 1$. If the system is constrained by m MPC equations, these equations can always be written in the following general matrix form:

$$\mathbf{CD} - \mathbf{Q} = \mathbf{0} \tag{11.28}$$

where $\mathbf{C}$ is a constant matrix of $m \times n$ $(m < n)$, and $\mathbf{Q}$ is a constant matrix of $m \times 1$. For example, let us assume that there is a system of 10 DOFs ($n = 10$); we can then write the displacement vector as

$$\mathbf{D}^T = \begin{Bmatrix} d_1 & d_2 & \cdots & d_{10} \end{Bmatrix} \tag{11.29}$$

If the two ($m = 2$) MPC equations for this system are given as

$$0.5d_1 - d_2 + 0.5d_3 = 0 \tag{11.30}$$

$$0.5d_4 - d_5 + 0.5d_6 = 0 \tag{11.31}$$

which have the same form as Eqs. (11.15) and (11.16). We then rewrite these MPCs in the form

$$\begin{aligned} &0.5d_1 - d_2 + 0.5d_3 + 0 \times d_4 + 0 \times d_5 + 0 \times d_6 + 0 \times d_7 + 0 \times d_8 + 0 \times d_9 \\ &\quad + 0 \times d_{10} = 0 \end{aligned} \tag{11.32}$$

$$\begin{aligned} &0 \times d_1 + 0 \times d_2 + 0 \times d_3 + 0.5d_4 - d_5 + 0.5d_6 + 0 \times d_7 + 0 \times d_8 + 0 \times d_9 \\ &\quad + 0 \times d_{10} = 0 \end{aligned} \tag{11.33}$$

From the above two equations, the matrix $\mathbf{C}$ is thus obtained as

$$\mathbf{C} = \begin{bmatrix} 0.5 & -1 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0.5 & -1 & 0.5 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{11.34}$$

and the vector $\mathbf{Q}$ is a null vector, given by

$$\mathbf{Q} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \tag{11.35}$$

A solution must thus be found for Eq. (11.27) subject to the constraint equations given by Eq. (11.28). The following are two methods often used to obtain such a constraint solution.

11.11.1 Lagrange Multiplier Method

First, the following m additional variables called *Lagrange multipliers* are introduced in this method as

$$\lambda = \begin{bmatrix} \lambda_1 & \lambda_2 & \cdots & \lambda_m \end{bmatrix}^T \tag{11.36}$$

corresponding to m MPC equations. Each equation in the matrix equation of Eq. (11.28) is then multiplied by the corresponding λ_i, which yields

$$\lambda^T\{\mathbf{CD} - \mathbf{Q}\} = 0 \tag{11.37}$$

The left-hand side of this equation is then added to the usual functional (refer to Eq. (3.2), Chapter 3) to obtain the *modified functional* for the constraint system:

$$\Pi_p = \tfrac{1}{2}\mathbf{D}^T\mathbf{KD} - \mathbf{D}^T\mathbf{F} + \lambda^T\{\mathbf{CD} - \mathbf{Q}\} \tag{11.38}$$

The solution for the constraint system is found at the stationary point of the modified functional. The stationary condition requires the derivatives of Π_{p} with respect to the $\mathbf{D}_i$ and λ_i to vanish, which yields

$$\begin{bmatrix} \mathbf{K} & \mathbf{C}^T \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \mathbf{D} \\ \boldsymbol{\lambda} \end{Bmatrix} = \begin{Bmatrix} \mathbf{F} \\ \mathbf{Q} \end{Bmatrix} \tag{11.39}$$

which is the set of algebraic equations for the constraint system. Equation (11.39) is then solved instead of Eq. (11.27) to obtain the solution that satisfies the MPC equations.

The advantage of this method is that the constraint equations are satisfied exactly. However, it can be seen that the total number of unknowns is increased. In addition, the expanded stiffness matrix in Eq. (11.39) is non-positive definite due to the presence of zero diagonal terms. Therefore, the efficiency of solving the system equations (11.39) is much lower than that of solving Eq. (11.27).

11.11.2 Penalty Method

The penalty method is a well-known and widely used method. In this method, Eq. (11.28) is written as

$$\mathbf{t} = \mathbf{CD} - \mathbf{Q} \tag{11.40}$$

so that $\mathbf{t} = 0$ implies full satisfaction of the constraints. The functional (refer to Eq. (3.2), Chapter 3) can then be modified as

$$\Pi_p = \tfrac{1}{2}\mathbf{D}^T\mathbf{KD} - \mathbf{D}^T\mathbf{F} + \tfrac{1}{2}\mathbf{t}^T\alpha\mathbf{t} \tag{11.41}$$

where $\alpha = \lceil \alpha_1 \quad \alpha_2 \quad \cdots \quad \alpha_m \rfloor$ is a diagonal matrix of 'penalty numbers' that are constants chosen by the analyst. The stationary condition of the modified functional requires the derivatives of Π_p with respect to the $\mathbf{D}_i$ to vanish, which yields

$$[\mathbf{K} + \mathbf{C}^T\alpha\mathbf{C}]\mathbf{D} = \mathbf{F} + \mathbf{C}^T\alpha\mathbf{Q} \tag{11.42}$$

where $\mathbf{C}^{\mathbf{T}}\alpha\mathbf{C}$ is called a 'penalty matrix'. From Eq. (11.42), it can be seen that if $\alpha = 0$, the constraints are ignored and the system equation reduces to the unconstrained equation

Eq. (11.27). As α_i becomes large, the penalty of violating constraints becomes large, so that constraints are closely satisfied.

Note that the choice of α_i can be a tricky task, as it depends upon many factors of the FEM model. Considering that the discretization errors can be of comparable magnitude to those of not satisfying the constraint, it has been suggested that [Zienkiewicz *et al.*, 2000]

$$\alpha = \text{constant}(1/h)^{p+1} \tag{11.43}$$

where h is the characteristic size of the elements, and p denotes the order of the element used. The following simple method for calculating the penalty factor works well for most problems:

$$\alpha = 1.0 \times 10^{4-6} \times \max\text{ (diagonal elements in the stiffness matrix)} \tag{11.44}$$

Also often used is

$$\alpha = 1.0 \times 10^{5-8} \times \text{Young's modulus} \tag{11.45}$$

Note that trials may be needed in choosing a proper penalty factor.

The advantage of this method is that the total number of unknowns is not changed, and the system equations generally behave well. Therefore, the efficiency of solving the system equations (11.42) is almost the same as that of solving Eq. (11.27). However, the constraint equations can only be satisfied approximately, and the right choice of α can be a problem for some cases.

11.12 REVIEW QUESTIONS

1. What are the conditions of the use of mirror symmetry for reducing the size of the finite element discretization of a problem? Give your answer with reference to geometry, material, boundary conditions and loading.
2. Indicate how the symmetric and antisymmetric conditions can be used on a half-model to obtain the natural frequencies of a clamped–clamped beam of length $2L$.

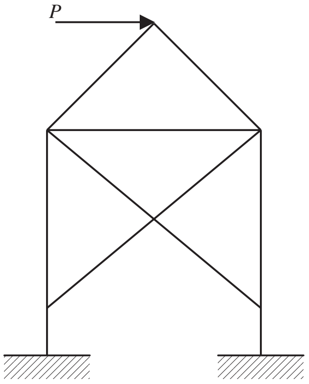


Figure 11.43. Frame structure subjected to horizontal load, *P*.

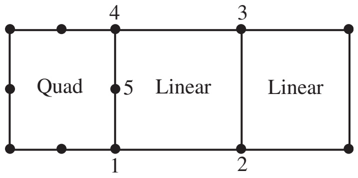


Figure 11.44. Mesh consisting of quadrilateral and linear elements.

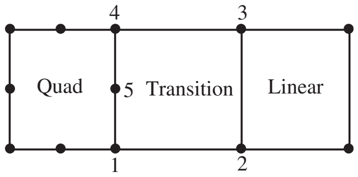


Figure 11.45. Mesh consisting of a transition element.

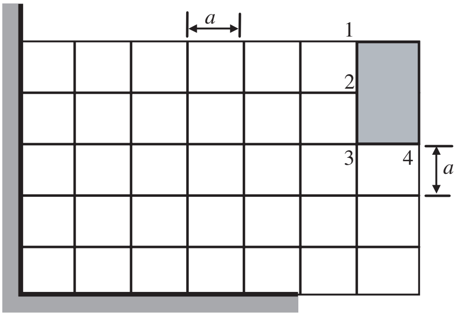


Figure 11.46. Mesh consisting of a rigid shaded portion.

3. Figure 11.43 shows a frame structure subjected to a horizontal force at the top. Explain the use of symmetry to solve the problem with the aid of diagrams.
4. Explain why a transition element or multipoint constraints are needed between the quadratic and linear elements shown in Figure 11.44.
5. Construct the shape functions in natural coordinates for the transition element shown in Figure 11.45. What is another way of solving the problem in Figure 11.44 if a transition element is not used?
6. Figure 11.46 shows a uniform mesh for a plane strain problem. The shaded block is rigid. Derive the multipoint constraint equations for nodes 1, 2, 3 and 4.
7. How are the constraint equations implemented using the penalty method in the system equation $\mathbf{Kd} = \mathbf{f}$? Here, $\mathbf{K}$ is the stiffness matrix for the global system, $\mathbf{d}$ is the displacement vector at all the nodes, and $\mathbf{f}$ is the external force vector acting at all the nodes. Assume that the constraint equations are given in the general form $\mathbf{Cd} = \mathbf{Q}$, where $\mathbf{C}$ and $\mathbf{Q}$ are given constant matrices.

12

FEM FOR HEAT TRANSFER PROBLEMS

12.1 FIELD PROBLEMS

This chapter deals with the use of the finite element method to solve the steady state heat transfer problem. Such a problem can be termed generally as one of the many field problems. Other field problems include torsional deformation of bars, irrotational flow and acoustic problems. This book makes use of the heat transfer problem to introduce concepts behind the solving of field problems using the FEM. Emphasis will be placed on one- and two-dimensional heat transfer problems. Three-dimensional problems can be solved in similar ways, except for the increase in DOFs due to the dependency of field variables on the third dimension. Many of the materials of this chapter are from the textbook by Segerlind (1984). The approach used to derive FE equations in heat transfer problems is a general approach to solve partial differential equations using the FEM. Therefore, the FE equations developed for heat transfer problems are directly applicable to all other types of field problems that are governed by a similar type of partial differential equation.

The general form of system equations of 2D linear steady state field problems can be given by the following general form of the *Helmholtz equation*:

$$D_x \frac{\partial^2 \phi}{\partial x^2} + D_y \frac{\partial^2 \phi}{\partial y^2} - g\phi + Q = 0 \tag{12.1}$$

where ϕ is the field variable, and D_x, D_y, g and Q are given constants whose physical meaning is different for different problems. For one-dimensional field problems, the general form of system equations can be written as

$$D \frac{\mathrm{d}^2 \phi}{\mathrm{d}x^2} - g\phi + Q = 0 \tag{12.2}$$

The following outlines different physical problems governed by Eqs. (12.1) or (12.2).

12.1.1 Heat Transfer in a Two-Dimensional Fin

Here we consider a problem of heat transfer in a two-dimensional fin, as shown in Figure 12.1. A 2D fin is mounted on a pipe. Heat conduction occurs in the x–y plane,

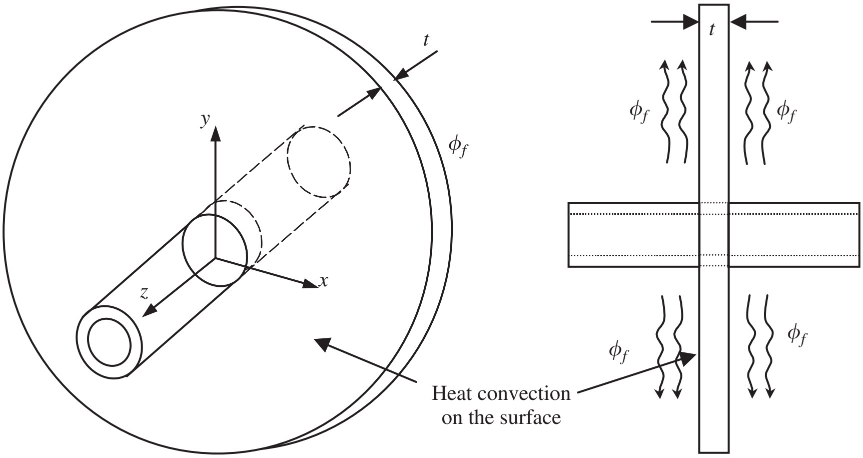


Figure 12.1. A 2D fin mounted on a pipe. Heat conduction occurs in the *x*–*y* plane, and heat convection occurs on the two surfaces and edges.

and heat convection occurs on the two surfaces and edges. Assume the fin is very thin, so that temperature does not vary significantly in the thickness (z) direction, therefore the temperature field is only a function of x and y. The governing equation for the temperature field in the fin, denoted by function $\phi(x, y)$, can be given by

$$\underbrace{-\left(k_x t\frac{\partial^2\phi}{\partial x^2} + k_y t\frac{\partial^2\phi}{\partial y^2}\right)}_{\text{Heat conduction}} + \underbrace{(2h\phi - 2h\phi_f)}_{\text{Heat convection}} = \underbrace{q}_{\text{Heat supply}} \tag{12.3}$$

where k_x, k_y are, respectively, the thermal conductivity coefficients in the x and y directions, h is the convection coefficient, t is the thickness of the fin, and ϕ_f is the ambient temperature of the surrounding fluid. The heat supply is denoted by q, which can be a function of x and y. The governing equation (12.3) can be derived simply using Fourier's laws of heat conduction and convection, as well as the conservation law of energy (heat). Equation (12.3) simply states that the heat loss due to heat conduction and heat convection should equal the heat supply at any point in the fin.

It can be seen that Eq. (12.3) takes on the general form of a field problem as in Eq. (12.1), with the substitution of

$$D_x = k_x t, \quad D_y = k_y t, \quad g = 2h, \quad Q = q + 2h\phi_f \tag{12.4}$$

12.1.2 Heat Transfer in a Long Two-Dimensional Body

If the domain is elongated in the z direction, and the geometry and temperature do not vary in the z direction, as illustrated in Figure 12.2, then a representative 2D 'slice' can be used

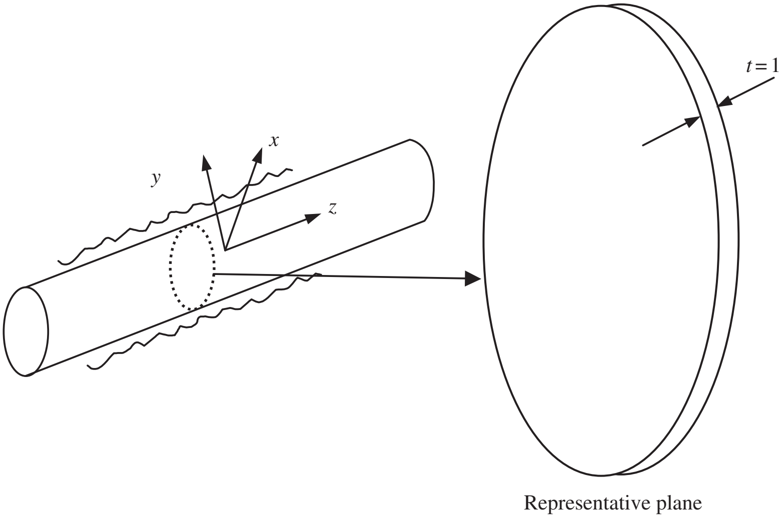


Figure 12.2. Heat transfer in a long body.

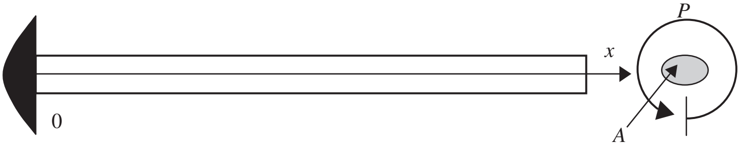


Figure 12.3. Heat conduction along a thin one-dimensional fin.

for modelling the problem. In this case, there is no heat convection on the two surfaces of the 2D slice, and the governing equation becomes

$$\underbrace{k_x \frac{\partial^2 \phi}{\partial x^2} + k_y \frac{\partial^2 \phi}{\partial y^2}}_{\text{Heat conduction}} + \underbrace{q}_{\text{Heat supply}} = 0 \tag{12.5}$$

which relates to the general form of the field equation with the following substitutions:

$$D_x = k_x, \quad D_y = k_y, \quad g = 0 \quad \text{and} \quad Q = q \tag{12.6}$$

12.1.3 Heat Transfer in a One-Dimensional Fin

Consider a one-dimensional fin, shown in Figure 12.3. Assume that the fin is very thin, and the dimensions of the cross-section of the fin are much smaller than the length of the fin, so temperature does not vary in the cross-section, and hence is only a function of x. Heat conduction occurs in the x direction, and heat convection occurs on the circumferential surface of the fin. The temperature field in the fin, denoted by function $\phi(x)$, is governed by the equation

$$\underbrace{kA\frac{\mathrm{d}^2\phi}{\mathrm{d}x^2}}_{\text{Heat conduction}}\ \underbrace{-hP\phi+hP\phi_f}_{\text{Heat convection}}+\ \underbrace{q}_{\text{Heat supply}}\ =0 \tag{12.7}$$

where k is the thermal conductivity, h is the convection coefficient, A is the cross-sectional area of the fin, and P is the perimeter of the fin, as shown in Figure 12.3. Equation (12.7) is created in the same way as its 2D counterpart of Eq. (12.3). Equation (12.7) can be written in the general form of Eq. (12.2), with the substitutions of

$$D=kA,\quad g=hP,\quad Q=q+hP\phi_f \tag{12.8}$$

12.1.4 Heat Transfer Across a Composite Wall

Many walls of industrial structures, or even simple appliances like a thermal flask, are composite in nature, i.e. they consist of more than one material. By utilizing the thermal conductivity properties of the material chosen, either thermal insulation or effective thermal heat transfer through the walls can be achieved.

Consider the heat transfer across a composite wall, as shown in Figure 12.4. The wall is assumed to be infinitely long in the y direction, and hence the heat source and any heat exchanges are also independent of y. In this case, the problem is one-dimensional. Since the wall is infinitely long, it is also not possible to have heat convection along the x-axis.

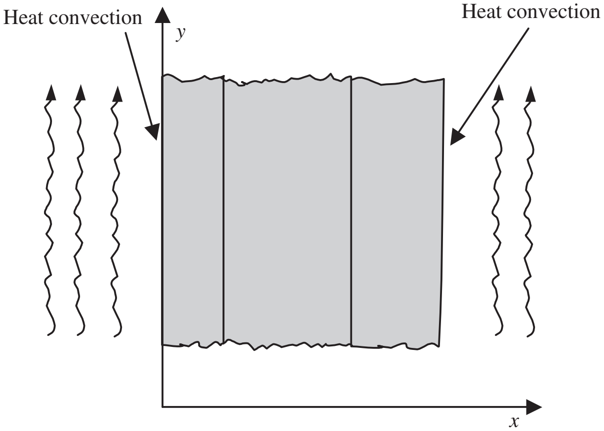


Figure 12.4. Heat transfer through a composite wall of three layers. Assume the temperature does not vary in the y direction.

Therefore, the system equation in this case is much simpler compared to that of the fin, and it governs only the heat conduction through the composite wall, which can be given by

$$\underbrace{kA\frac{\mathrm{d}^2\phi}{\mathrm{d}x^2}}_{\text{Heat conduction}} + \underbrace{q}_{\text{Heat supply}} = 0 \tag{12.9}$$

where ϕ is the known temperature. Equation (12.9) can be written in a general form of Eq. (12.2) with

$$D = kA, \quad g = 0, \quad Q = q \tag{12.10}$$

12.1.5 Torsional Deformation of a Bar

For problems of torsional deformation of a bar with noncircular sections, the field variable will be the stress function ϕ that is governed by a Poisson's equation,

$$\frac{1}{G}\frac{\partial^2\phi}{\partial x^2} + \frac{1}{G}\frac{\partial^2\phi}{\partial y^2} + 2\theta = 0 \tag{12.11}$$

where G is the shear modulus and θ is the given angle of twist. The stress function is defined by

$$\begin{aligned}\sigma_{xz} &= \frac{\partial\phi}{\partial y}\\ \sigma_{yz} &= -\frac{\partial\phi}{\partial x}\end{aligned} \tag{12.12}$$

Equation (12.11) can be easily derived in the following procedure. Consider pure torsional status where

$$\sigma_{xx} = \sigma_{yy} = \sigma_{zz} = \sigma_{xy} = 0 \tag{12.13}$$

Therefore, there are only two stress components, σ_{xz} and σ_{yz}. Using Hooke's law, we have

$$\begin{aligned}\varepsilon_{xz} &= \frac{\sigma_{xz}}{G}\\ \varepsilon_{yz} &= \frac{\sigma_{yz}}{G}\end{aligned} \tag{12.14}$$

The relationship between displacement and the given angle of twist θ can be given by [Fung, 1965]:

$$\begin{aligned}u &= -\theta yz\\ v &= \theta xz\end{aligned} \tag{12.15}$$

Next, using Eqs. (2.4), (12.15), (12.14) and (12.12), we obtain

$$\frac{\partial w}{\partial x} = -\frac{\partial u}{\partial z} + \varepsilon_{xz} = \theta y + \frac{1}{G}\frac{\partial\phi}{\partial y} \tag{12.16}$$

$$\frac{\partial w}{\partial y} = -\frac{\partial v}{\partial z} + \varepsilon_{yz} = -\theta x - \frac{1}{G}\frac{\partial\phi}{\partial x} \tag{12.17}$$

Differentiating Eq. (12.16) with respect to y and Eq. (12.17) with respect to x, and equating the two resultant equations, leads to Eq. (12.11).

It can be seen that Eq. (12.11) takes on the general form of Eq. (12.1), with

$$D_x = 1/G, \quad D_y = 1/G, \quad g = 0, \quad Q = 2\theta \tag{12.18}$$

12.1.6 Ideal Irrotational Fluid Flow

For problems of ideal, irrotational fluid flow, the field variables are the streamline, ψ, and potential, ϕ, functions that are governed by Laplace's equations

$$\frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} = 0 \tag{12.19}$$

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = 0 \tag{12.20}$$

Derivation of the above equations can be found in any textbook on fluid flows (e.g. see Daily and Harleman [1966]).

It can be seen that both Eqs. (12.19) and (12.20) take on the form of Eq. (12.1), with

$$D_x = D_y = 1, \quad g = Q = 0 \tag{12.21}$$

12.1.7 Acoustic Problems

For problems of vibrating fluid in a closed volume, as in the case of the vibration of air particles in a room, the field variable, P, is the pressure above the ambient pressure, and is governed by Poisson's equation

$$\frac{\partial^2 P}{\partial x^2} + \frac{\partial^2 P}{\partial y^2} + \frac{w^2}{c^2} P = 0 \tag{12.22}$$

where w is the wave frequency and c is the wave velocity in the medium. The derivation of the above equations can be found in any book on acoustics (e.g. see Crocker [1998]). It can be seen that Eq. (12.22) takes on the form of Eq. (12.1), with substitution of

$$g = -\frac{w^2}{c^2}, \quad D_x = D_y = 1, \quad Q = 0 \tag{12.23}$$

The above examples show that the Helmholtz equation (12.1) governs many different physical phenomena. Therefore, a productive way to solve all these problems using the FEM is to derive FE equations for solving the general form of Eqs. (12.1) or (12.2). The following sections thus introduce the FEM as a general numerical tool for solving partial differential equations in the form of Eqs. (12.1) or (12.2). In applying the FEM equations, we focus mainly on heat transfer problems.

12.2 WEIGHTED RESIDUAL APPROACH FOR FEM

In Chapter 3, Hamilton's principle is utilized to formulate the finite element equations. In that case, one needs to know the functional for the physical problem. For many engineering problems, one does not know the functional or it is not known as intuitively as in mechanics problems. Instead, the governing equation for the problem would be known. What we want to do now is establish FEM equations based on the governing equation, but without knowing the functional. In this case, it is convenient to use the *weighted residual* approach to formulate the FEM system equations.

The general form of Eq. (12.1) can be rewritten in the form

$$f(\phi(x, y)) = 0 \tag{12.24}$$

where the function f is given as

$$f(\phi(x, y)) = D_x \frac{\partial^2 \phi}{\partial x^2} + D_y \frac{\partial^2 \phi}{\partial y^2} - g\phi + Q \tag{12.25}$$

In general, it is difficult to obtain the *exact* solution of $\phi(x, y)$ which satisfies Eq. (12.24). Therefore, an approximated solution of $\phi(x, y)$ is sought for, which satisfies Eq. (12.24) in a weighted integral sense, i.e.

$$\int_A W f(\phi(x, y))\, \mathrm{d}x\, \mathrm{d}y = 0 \tag{12.26}$$

where W is the weight function. We hope that the solution of $\phi(x, y)$ that satisfies Eq. (12.26) can be a good approximation of the exact results. This is the basic idea behind the weighted residual approach. This approach is very simple, and can be used in the finite element method to establish the discretized system equations as described below.

To ensure a good approximation, the problem domain is divided into smaller subdomains (elements), as we have done in previous chapters, as shown in Figure 12.5. In each

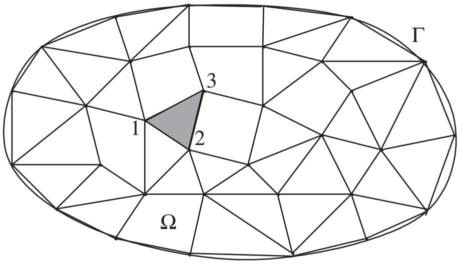


Figure 12.5. Division of problem domain Ω bounded by Γ into elements.

element, it is assumed that

$$\phi^h(x, y) = \mathbf{N}(x, y)\mathbf{\Phi}^{(e)} \tag{12.27}$$

where the superscript h indicates that the field variable is approximated, and

$$\mathbf{N} = \begin{bmatrix} N_1 & N_2 & \cdots & N_{n_d} \end{bmatrix} \tag{12.28}$$

in which N_i is the shape function of x and y, and n_d is the number of the nodes of the element. For the triangular element shown in Figure 12.5, $n_d = 3$. In Eq. (12.27), $\mathbf{\Phi}^{(e)}$ is the field variable at the nodes of the element. There are a number of ways in which the weight function, W, can be chosen when developing the element equations. When the shape functions are used as the weight function, the method is called the Galerkin method, which is one of the most popular methods for developing the FE equation. The shape function N_i in $\mathbf{N}$ is usually developed in exactly the same manner as discussed in Chapter 3 and other previous chapters. The functions developed should in general satisfy the sufficient conditions discussed in Chapter 3: the delta function property, the partition of unity property and the linear reproduction property.

Using the Galerkin method, the residual calculated at all the nodes for an element is then evaluated by the equation

$$\mathbf{R}^{(e)} = \int_{A_e} \mathbf{N}^T f(\phi(x, y))\, \mathrm{d}x\, \mathrm{d}y \tag{12.29}$$

Finally, the total residual at each of all the nodes in the problem domain is then assembled and enforced to zero to establish the system equation for the whole system. FEM equations will be developed in the following sections for one- and two-dimensional field problems, using a heat transfer problem as an example.

12.3 1D HEAT TRANSFER PROBLEM

12.3.1 One-Dimensional Fin

Consider the one-dimensional fin shown in Figure 12.6. The governing differential equation for a steady-state heat transfer problem for the fin is given by Eq. (12.7). The boundary conditions associated with Eq. (12.7) usually consist of a specified temperature at $x = 0$

$$\phi(0) = \phi_0 \tag{12.30}$$

and convection heat loss at the free end, where $x = H$

$$-kA\frac{\mathrm{d}\phi}{\mathrm{d}x} = hA(\phi_b - \phi_f) \quad \text{at } x = H \tag{12.31}$$

where ϕ_b is the temperature at the end of the fin and is not known prior to the solution of the problem. Note that the convective heat transfer coefficient in Eq. (12.31) may or may not be the same as that in Eq. (12.7).

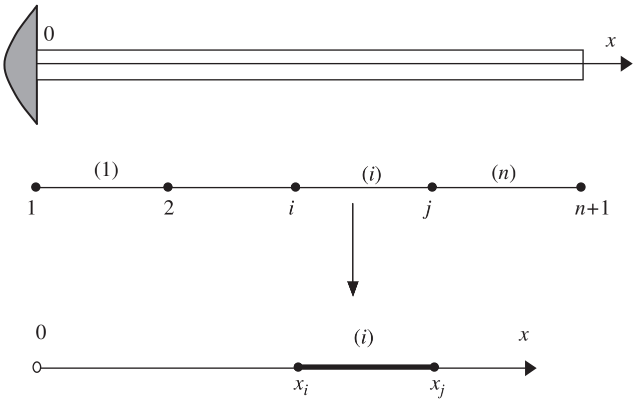


Figure 12.6. One-dimensional problem: heat transfer in a thin fin that is divided into *n* elements.

Using the same finite element technique, the fin is divided into elements as shown in Figure 12.6. In one element, the residual equation, $\mathbf{R}^{(e)}$ can be obtained using the Galerkin approach as in Eq. (12.29):

$$\begin{aligned}\mathbf{R}^{(e)} &= -\int_{x_i}^{x_j} \mathbf{N}^T \left(D\frac{\mathrm{d}^2\phi^h}{\mathrm{d}x^2} - g\phi^h + Q \right)\mathrm{d}x \\ &= -\int_{x_i}^{x_j} \mathbf{N}^T \left(D\frac{\mathrm{d}^2\phi^h}{\mathrm{d}x^2} + Q \right)\mathrm{d}x + \int_{x_i}^{x_j} g\mathbf{N}^T\phi^h\,\mathrm{d}x \end{aligned} \tag{12.32}$$

where $D = kA$, $g = hP$ and $Q = q + hP\phi_f$ for heat transfer in thin fins. Note that the minus sign is added to the residual mainly for convenience. *Integration by parts* is then performed on the first term of the right-hand side of Eq. (12.32), leading to

$$\mathbf{R}^{(e)} = -\mathbf{N}^T D\left.\frac{\mathrm{d}\phi^h}{\mathrm{d}x}\right|_{x_i}^{x_j} + \int_{x_i}^{x_j} \frac{\mathrm{d}\mathbf{N}^T}{\mathrm{d}x} D \frac{\mathrm{d}\phi^h}{\mathrm{d}x}\mathrm{d}x - \int_{x_i}^{x_j} Q\mathbf{N}^T\,\mathrm{d}x + \int_{x_i}^{x_j} g\mathbf{N}^T\phi^h\,\mathrm{d}x \tag{12.33}$$

Using the usual interpolation of the field variable, ϕ^h, by the shape functions in the 1D case,

$$\phi^h(x) = \mathbf{N}(x)\mathbf{\Phi}^{(e)} \tag{12.34}$$

and substituting Eq. (12.34) into Eq. (12.33), gives

$$\begin{aligned}\mathbf{R}^{(e)} = &\underbrace{-\mathbf{N}^T D\left.\frac{\mathrm{d}\phi^h}{\mathrm{d}x}\right|_{x_i}^{x_j}}_{\mathbf{b}^{(e)}} + \underbrace{\left(\int_{x_i}^{x_j} \frac{\mathrm{d}\mathbf{N}^T}{\mathrm{d}x} D \frac{\mathrm{d}\mathbf{N}}{\mathrm{d}x}\mathrm{d}x\right)}_{\mathbf{k}_D^{(e)}}\mathbf{\Phi}^{(e)} \\ &- \underbrace{\left(\int_{x_i}^{x_j} Q\mathbf{N}^T\,\mathrm{d}x\right)}_{\mathbf{f}_Q^{(e)}} + \underbrace{\left(\int_{x_i}^{x_j} g\mathbf{N}^T\mathbf{N}\,\mathrm{d}x\right)}_{\mathbf{k}_g^{(e)}}\mathbf{\Phi}^{(e)}\end{aligned} \tag{12.35}$$

or

$$\mathbf{R}^{(e)} = \mathbf{b}^{(e)} + \left[\mathbf{k}_D^{(e)} + \mathbf{k}_g^{(e)}\right] \mathbf{\Phi}^{(e)} - \mathbf{f}_Q^{(e)} \tag{12.36}$$

where

$$\mathbf{k}_D^{(e)} = \int_{x_i}^{x_j} \frac{\mathrm{d}\mathbf{N}^T}{\mathrm{d}x} D \frac{\mathrm{d}\mathbf{N}}{\mathrm{d}x} \mathrm{d}x = \int_{x_i}^{x_j} \mathbf{B}^T D \mathbf{B} \, \mathrm{d}x \tag{12.37}$$

is the element matrix of thermal conduction. Matrix $\mathbf{k}_g^{(e)}$ in Eq. (12.36) is defined by

$$\mathbf{k}_g^{(e)} = \int_{x_i}^{x_j} g\mathbf{N}^T \mathbf{N} \, \mathrm{d}x \tag{12.38}$$

which is the matrix of thermal convection on the circumference of the element. Vector $\mathbf{f}_Q^{(e)}$ is associated with the external heat applied on the element, defined as

$$\mathbf{f}_Q^{(e)} = \int_{x_i}^{x_j} Q\mathbf{N}^T \, \mathrm{d}x \tag{12.39}$$

Finally, $\mathbf{b}^{(e)}$ is defined by

$$\mathbf{b}^{(e)} = -\mathbf{N}^T D \left. \frac{\mathrm{d}\phi^h}{\mathrm{d}x} \right|_{x_i}^{x_j} \tag{12.40}$$

which associates with the gradient of the temperature (or heat flux) at the two ends of the element.

In Eq. (12.37),

$$\mathbf{B} = \frac{\mathrm{d}\mathbf{N}}{\mathrm{d}x} \tag{12.41}$$

is the same as the *strain matrix* in the case of mechanics problems. For linear elements, the shape functions are as follows:

$$\mathbf{N}(x) = \begin{bmatrix} N_i & N_j \end{bmatrix} = \begin{bmatrix} \frac{x_j - x}{l} & \frac{x - x_i}{l} \end{bmatrix} \tag{12.42}$$

and

$$\mathbf{B} = \frac{\mathrm{d}\mathbf{N}}{\mathrm{d}x} = \frac{\mathrm{d}}{\mathrm{d}x} \begin{bmatrix} \frac{x_j - x}{l} & \frac{x - x_i}{l} \end{bmatrix} = \begin{bmatrix} \frac{-1}{l} & \frac{1}{l} \end{bmatrix} \tag{12.43}$$

Substituting the above equation into Eq. (12.37), the heat conduction $\mathbf{k}_D^{(e)}$ matrix is obtained as

$$\mathbf{k}_D^{(e)} = \int_{x_i}^{x_j} \begin{bmatrix} -1/l \\ 1/l \end{bmatrix} D \begin{bmatrix} -1/l & 1/l \end{bmatrix} \mathrm{d}x = \frac{kA}{l} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{12.44}$$

In deriving the above equation, $D = kA$ has been used. Compared with Eq. (4.15), it can be seen that $\mathbf{k}_D^{(e)}$ is in fact the analogy of the stiffness matrix of a truss structure. The tensile stiffness coefficient AE/l has been replaced by the heat conductivity coefficient of kA/l.

Similarly, to obtain the convection matrix $\mathbf{k}_g^{(e)}$ corresponding to the heat convection, substitute Eq. (12.42) into Eq. (12.38),

$$\mathbf{k}_g^{(e)} = \int_{x_i}^{x_j} g \begin{bmatrix} (x_j - x)/l \\ (x - x_i)/l \end{bmatrix} \begin{bmatrix} (x_j - x)/l & (x - x_i)/l \end{bmatrix} \mathrm{d}x = \frac{hPl}{6} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \tag{12.45}$$

In deriving the above equation, $g = hP$ has been used. Compared with Eq. (4.16), it can be seen that $\mathbf{k}_g^{(e)}$ is in fact the analogy of the mass matrix of a truss structure. The total mass of the truss element $A\rho l$ corresponds to the total heat convection rate of hPl.

The nodal heat vector $\mathbf{f}_Q^{(e)}$ is obtained when Eq. (12.42) is substituted into Eq. (12.39), giving

$$\mathbf{f}_Q^{(e)} = \int_{x_i}^{x_j} Q \begin{bmatrix} (x_j - x)/l \\ (x - x_i)/l \end{bmatrix} \mathrm{d}x = \frac{Ql}{2} \begin{Bmatrix} 1 \\ 1 \end{Bmatrix} = \left(\underbrace{q}_{\text{Heat supply}} + \underbrace{hP\phi_f}_{\text{Heat convection}} \right) \frac{l}{2} \begin{Bmatrix} 1 \\ 1 \end{Bmatrix} \tag{12.46}$$

In deriving the above equation, $Q = q + hP\phi_f$ (see Eq. (12.8)) has been used. The nodal heat vector consists of the heat supply and the convectional heat input to the fin. The nodal heat vector is the analogy of the nodal force vector for truss elements.

Finally, let us analyse the vector $\mathbf{b}^{(e)}$ defined by Eq. (12.40), which is associated with the thermal conditions on the boundaries of the element:

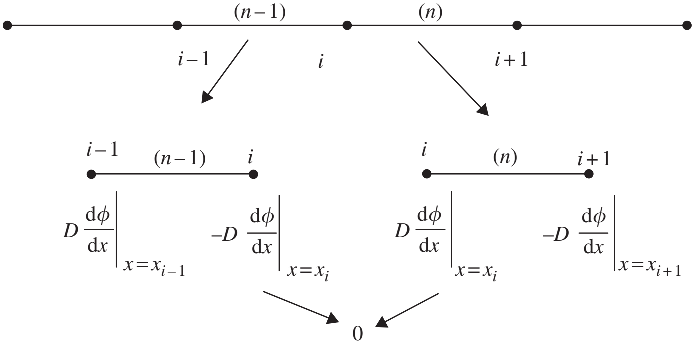
$$\mathbf{b}^{(e)} = -\mathbf{N}^T D \left.\frac{\mathrm{d}\phi}{\mathrm{d}x}\right|_{x_i}^{x_j} = \begin{Bmatrix} kA \left.\dfrac{\mathrm{d}\phi}{\mathrm{d}x}\right|_{x=x_i} \\ -kA \left.\dfrac{\mathrm{d}\phi}{\mathrm{d}x}\right|_{x=x_j} \end{Bmatrix} = \underbrace{\begin{Bmatrix} kA \left.\dfrac{\mathrm{d}\phi}{\mathrm{d}x}\right|_{x=x_i} \\ 0 \end{Bmatrix}}_{\mathbf{b}_L^{(e)}} + \underbrace{\begin{Bmatrix} 0 \\ -kA \left.\dfrac{\mathrm{d}\phi}{\mathrm{d}x}\right|_{x=x_j} \end{Bmatrix}}_{\mathbf{b}_R^{(e)}} \tag{12.47}$$

or

$$\mathbf{b}^{(e)} = \mathbf{b}_L^{(e)} + \mathbf{b}_R^{(e)} \tag{12.48}$$

where the subscripts 'L' and 'R' stand for the left and right ends of the element. It can be easily proven that at the internal nodes of the fin, $\mathbf{b}_L^{(e)}$ and $\mathbf{b}_R^{(e)}$ vanish when the elements are assembled, as illustrated in Figure 12.7. As a result, $\mathbf{b}$ needs to be determined only for the nodes on the boundary by using the conditions prescribed. At boundaries where the temperature is prescribed, as shown in Eq. (12.30), $\mathbf{b}_L^{(e)}$ or $\mathbf{b}_R^{(e)}$ are yet to know. In fact, there is no need to know $\mathbf{b}_L^{(e)}$ or $\mathbf{b}_R^{(e)}$ in the stage of solving the system equation, as the temperature at the node is already known. The situation is very much the same at the prescribed displacement boundary, where the reaction force is usually unknown at the stage of solving system equations.

However, when there is heat convection at the ends of the fin, as prescribed in Eq. (12.31), $\mathbf{b}_L^{(e)}$ or $\mathbf{b}_R^{(e)}$ are obtained using the boundary conditions, since the heat flux there can be



(n−1)
(n)
i−1
i
i+1
i−1
(n−1)
i
i
(n)
i+1
$D \left.\frac{d\phi}{dx}\right|_{x=x_{i-1}}$
$-D \left.\frac{d\phi}{dx}\right|_{x=x_i}$
$D \left.\frac{d\phi}{dx}\right|_{x=x_i}$
$-D \left.\frac{d\phi}{dx}\right|_{x=x_{i+1}}$
0

Figure 12.7. Vector $\boldsymbol{b}^{(e)}$ vanishes at internal points after assembly.

calculated. For example, for boundary conditions defined by Eq. (12.31), we have

$$\mathbf{b}_R^{(e)} = \begin{Bmatrix} 0 \\ hA(\phi_b - \phi_f) \end{Bmatrix} = \begin{Bmatrix} 0 \\ hA\phi_j \end{Bmatrix} - \begin{Bmatrix} 0 \\ hA\phi_f \end{Bmatrix} \tag{12.49}$$

Since ϕ_b is the temperature of the fin at the boundary point, we have $\phi_b = \phi_j$, which is an unknown. Equation (12.49) can be rewritten as

$$\mathbf{b}_R^{(e)} = \underbrace{\begin{bmatrix} 0 & 0 \\ 0 & hA \end{bmatrix}}_{\mathbf{k}_M^{(e)}} \begin{Bmatrix} \phi_i \\ \phi_j \end{Bmatrix} - \underbrace{\begin{Bmatrix} 0 \\ hA\phi_f \end{Bmatrix}}_{\mathbf{f}_s^{(e)}} \tag{12.50}$$

or

$$\mathbf{b}_R^{(e)} = \mathbf{k}_M^{(e)} \boldsymbol{\Phi}^{(e)} - \mathbf{f}_s^{(e)} \tag{12.51}$$

where

$$\mathbf{k}_M^{(e)} = \begin{bmatrix} 0 & 0 \\ 0 & hA \end{bmatrix} \tag{12.52}$$

and

$$\mathbf{f}_s^{(e)} = \begin{Bmatrix} 0 \\ hA\phi_f \end{Bmatrix} \tag{12.53}$$

Note that Eqs. (12.51)–(12.53) are derived assuming that the convective boundary is on node j, which is the right side of the element. If the convective boundary is on the left side of the element, we then have

$$\mathbf{b}_L^{(e)} = \mathbf{k}_M^{(e)} \boldsymbol{\Phi}^{(e)} - \mathbf{f}_s^{(e)} \tag{12.54}$$

where

$$\mathbf{k}_M^{(e)} = \begin{bmatrix} hA & 0 \\ 0 & 0 \end{bmatrix} \tag{12.55}$$

and

$$\mathbf{f}_s^{(e)} = \begin{Bmatrix} hA\phi_f \\ 0 \end{Bmatrix} \tag{12.56}$$

Substituting the expressions for $\mathbf{b}^{(e)}$ back into Eq. (12.36), we obtain

$$\mathbf{R}^{(e)} = \underbrace{\left[\mathbf{k}_D^{(e)} + \mathbf{k}_g^{(e)} + \mathbf{k}_M^{(e)}\right]}_{\mathbf{k}^{(e)}} \mathbf{\Phi}^{(e)} - \underbrace{\left\{\mathbf{f}_Q^{(e)} + \mathbf{f}_S^{(e)}\right\}}_{\mathbf{f}^{(e)}} \tag{12.57}$$

or in a simplified form of

$$\mathbf{R}^{(e)} = \mathbf{k}^{(e)}\mathbf{\Phi}^{(e)} - \mathbf{f}^{(e)} \tag{12.58}$$

where

$$\mathbf{k}^{(e)} = \mathbf{k}_D^{(e)} + \mathbf{k}_g^{(e)} + \mathbf{k}_M^{(e)} \tag{12.59}$$

and

$$\mathbf{f}^{(e)} = \mathbf{f}_Q^{(e)} + \mathbf{f}_S^{(e)} \tag{12.60}$$

Note that $\mathbf{k}_M^{(e)}$ and $\mathbf{f}_S^{(e)}$ exist only if the node is on the convective boundary, and they are given by Eqs. (12.52) and (12.53) or Eqs. (12.55) and (12.56), depending on the location of the node. If the boundary is insulated, meaning there is no heat exchange occurring there, both $\mathbf{k}_M^{(e)}$ and $\mathbf{f}_S^{(e)}$ vanish, because $\mathrm{d}\phi/\mathrm{d}x = 0$ at such a boundary. After obtaining the element matrices, the residual defined by Eq. (12.58) is assembled, and enforced to equate to zero, which will lead to the following global system equation:

$$\mathbf{KD} = \mathbf{F} \tag{12.61}$$

The above equation has the same form as that for a static mechanics problem. The detailed assembly process is described in the examples that follow.

12.3.2 Direct Assembly Procedure

Consider an element equation of residual in the form

$$\mathbf{R}^{(e)} = \mathbf{k}^{(e)}\mathbf{\Phi}^{(e)} - \mathbf{f}^{(e)} \tag{12.62}$$

or in the expanded form

$$\begin{Bmatrix} R_1^{(e)} \\ R_2^{(e)} \end{Bmatrix} = \begin{bmatrix} k_{11}^{(e)} & k_{12}^{(e)} \\ k_{21}^{(e)} & k_{22}^{(e)} \end{bmatrix} \begin{Bmatrix} \phi_1^{(e)} \\ \phi_2^{(e)} \end{Bmatrix} - \begin{Bmatrix} f_1^{(e)} \\ f_2^{(e)} \end{Bmatrix} \tag{12.63}$$

Consider now two linear one-dimensional elements, as shown in Figure 12.8. We have for element 1,

$$R_1^{(1)} = k_{11}^{(1)}\phi_1 + k_{12}^{(1)}\phi_2 - f_1^{(1)} \tag{12.64}$$

$$R_2^{(1)} = k_{21}^{(1)}\phi_1 + k_{22}^{(1)}\phi_2 - f_2^{(1)} \tag{12.65}$$

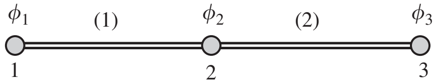


Figure 12.8. Assembling of two elements demonstrating the direct assembly principle.

and for element 2,

$$R_1^{(2)} = k_{11}^{(2)}\phi_2 + k_{12}^{(2)}\phi_3 - f_1^{(2)} \tag{12.66}$$

$$R_2^{(2)} = k_{21}^{(2)}\phi_2 + k_{22}^{(2)}\phi_3 - f_2^{(2)} \tag{12.67}$$

The total residual at a node should be obtained by summarizing all the residuals contributed from all the elements that are connected to the node, and the total residual should vanish to ensure the best satisfaction of the system equations. We then have at node 1,

$$R_1^{(1)} = 0: \quad k_{11}^{(1)}\phi_1 + k_{12}^{(1)}\phi_2 - f_1^{(1)} = 0 \tag{12.68}$$

at node 2,

$$R_1^{(2)} + R_2^{(1)} = 0: \quad k_{21}^{(1)}\phi_1 + \left(k_{22}^{(1)} + k_{11}^{(2)}\right)\phi_2 + k_{12}^{(2)}\phi_3 - \left(f_2^{(1)} + f_1^{(2)}\right) = 0 \tag{12.69}$$

and at node 3,

$$R_2^{(2)} = 0: \quad k_{21}^{(2)}\phi_2 + k_{22}^{(2)}\phi_3 - f_2^{(2)} = 0 \tag{12.70}$$

Writing Eqs. (12.68), (12.69) and (12.70) in matrix form gives

$$\begin{bmatrix} k_{11}^{(1)} & k_{12}^{(1)} & 0 \\ k_{21}^{(1)} & k_{22}^{(1)} + k_{11}^{(2)} & k_{12}^{(2)} \\ 0 & k_{21}^{(2)} & k_{22}^{(2)} \end{bmatrix} \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{Bmatrix} = \begin{Bmatrix} f_1^{(1)} \\ f_2^{(1)} + f_1^{(2)} \\ f_2^{(2)} \end{Bmatrix} \tag{12.71}$$

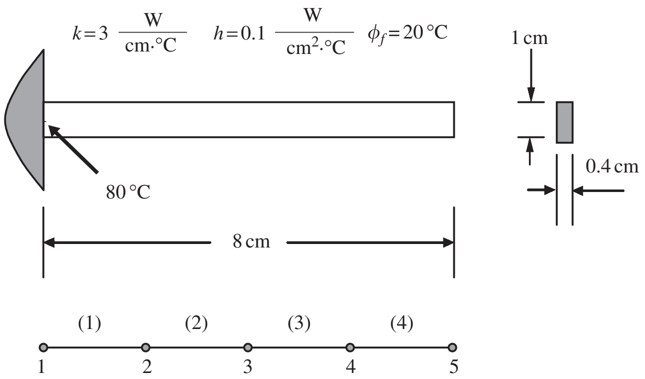
which is the same as the assembly procedure introduced in Section 3.4.7 and Example 4.2.

12.3.3 Worked Example

Example 12.1: Heat transfer along 1D fin of rectangular cross-section

The temperature distribution in the fin, as shown in Figure 12.9, is to be calculated using the finite element method. The fin is rectangular in shape, 8 cm long, 0.4 cm wide, and 1 cm thick. Assume that convection heat loss occurs from the right end of the fin.

Analysis of the problem. The fin is divided uniformly into four elements with a total of five nodes. Each element is with a length of $l = 2$ cm. The system equation should be 5×5. At node 1 the temperature is specified, therefore there is no need to calculate $\mathbf{k}_M^{(e)}$ and $\mathbf{f}_S^{(e)}$ for element 1 as only the temperature is requested. As nodes 2, 3 and 4 are internal, there is also no need to calculate $\mathbf{k}_M^{(e)}$ and $\mathbf{f}_S^{(e)}$. Since heat convection is occurring on node 5, we have to calculate $\mathbf{k}_M^{(e)}$ and $\mathbf{f}_S^{(e)}$ using Eqs. (12.52) and (12.53).



$k=3\ \frac{\text{W}}{\text{cm}\cdot{}^\circ\text{C}}$
$h=0.1\ \frac{\text{W}}{\text{cm}^2\cdot{}^\circ\text{C}}$
$\phi_f=20\,^\circ\text{C}$
1 cm
0.4 cm
80 °C
8 cm
(1)
(2)
(3)
(4)
1
2
3
4
5

Figure 12.9. A one-dimensional fin of a rectangular cross-section.

Data preparation

$$\frac{kA}{l} = \frac{3(0.4)}{2} = 0.6\,\frac{\text{W}}{{}^\circ\text{C}} \tag{12.72}$$

$$\frac{hPl}{6} = \frac{0.1(2.8)2}{6} = 0.093\,\frac{\text{W}}{{}^\circ\text{C}} \tag{12.73}$$

$$hA = 0.1(0.4) = 0.04\,\frac{\text{W}}{{}^\circ\text{C}} \tag{12.74}$$

$$\frac{hPl\phi_f}{2} = \frac{0.1(2.8)(20)(2)}{2} = 5.6\,\text{W} \tag{12.75}$$

$$hA\phi_f = 0.1(0.4)(20) = 0.8\,\text{W} \tag{12.76}$$

Solution. The general forms of the element matrices for elements 1, 2 and 3 are

$$\mathbf{k}^{(e)} = \frac{kA}{l}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + \frac{hPl}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \quad \text{and} \tag{12.77}$$

$$\mathbf{f}^{(e)} = \frac{hPl\phi_f}{2}\begin{Bmatrix} 1 \\ 1 \end{Bmatrix} \tag{12.78}$$

Substituting the data into the above two equations, we have

$$\mathbf{k}^{(1,2,3)} = \begin{bmatrix} 0.786 & -0.507 \\ -0.507 & 0.786 \end{bmatrix} \quad \text{and} \quad \mathbf{f}^{(1,2,3)} = \begin{Bmatrix} 5.6 \\ 5.6 \end{Bmatrix} \tag{12.79}$$

The general forms of the element matrices for element 4 are

$$\mathbf{k}^{(e)} = \frac{kA}{l}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + \frac{hPl}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & hA \end{bmatrix} \quad \text{and} \tag{12.80}$$

$$\mathbf{f}^{(e)} = \frac{hPl\phi_f}{2}\begin{Bmatrix} 1 \\ 1 \end{Bmatrix} + \begin{Bmatrix} 0 \\ hA\phi_f \end{Bmatrix} \tag{12.81}$$

Substituting the data into the above equations, we have

$$\mathbf{k}^{(4)} = \begin{bmatrix} 0.786 & -0.507 \\ -0.507 & 0.826 \end{bmatrix} \quad \text{and} \quad \mathbf{f}^{(4)} = \begin{Bmatrix} 5.6 \\ 6.4 \end{Bmatrix} \tag{12.82}$$

The next step is to assemble the element matrices together to form the global system equations. The assembly is carried out using the direct assembly procedure described in the previous section. We obtain

$$\begin{bmatrix} 0.786 & -0.507 & 0 & 0 & 0 \\ -0.507 & 1.572 & -0.507 & 0 & 0 \\ 0 & -0.507 & 1.572 & -0.507 & 0 \\ 0 & 0 & -0.507 & 1.572 & -0.507 \\ 0 & 0 & 0 & -0.507 & 0.826 \end{bmatrix} \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \end{Bmatrix} = \begin{Bmatrix} 5.6 \\ 11.2 \\ 11.2 \\ 11.2 \\ 6.4 \end{Bmatrix} + \begin{Bmatrix} Q^* \\ 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix} \tag{12.83}$$

Note that at node 1 the temperature is to be fixed at a particular value. This requires a heat exchanger (or heat source) there for it to happen. The heat source is unknown at this stage, and we do not need to know it actually in computing the temperature distribution. However, it is required in balancing the equation. We therefore simply note it as Q^*. Since the temperature at node 1 is given as 80°C, we actually have four unknown temperatures in a 5×5 matrix equation. This implies that we can actually eliminate Q^* in the first system equation as indicated below, and still solve the four unknowns with the remaining 4×4 matrix equation. Note that the term in the second row, first column (circled term) must also be accounted for in the four remaining system equations:

$$\begin{bmatrix} \cancel{0.786} & \cancel{0.507} & \cancel{0} & \cancel{0} & \cancel{0} \\ \boxed{\cancel{-0.507}} & 1.572 & -0.507 & 0 & 0 \\ \cancel{0} & -0.507 & 1.572 & -0.507 & 0 \\ \cancel{0} & 0 & -0.507 & 1.572 & -0.507 \\ \cancel{0} & 0 & 0 & -0.507 & 0.826 \end{bmatrix} \begin{Bmatrix} \cancel{\phi_1 = 80} \\ \phi_2 \\ \phi_3 \\ \phi_4 \\ \phi_5 \end{Bmatrix} = \begin{Bmatrix} \cancel{5.6 + Q^*} \\ 11.2 + 80 \times 0.507 \\ 11.2 \\ 11.2 \\ 6.4 \end{Bmatrix} \tag{12.84}$$

Solving the 4×4 equation gives

$$\mathbf{\Phi}^T = \{80.0 \quad 42.0 \quad 28.2 \quad 23.3 \quad 22.1\} \tag{12.85}$$

12.3.4 Remarks

Formulations and examples given in this section have clearly shown that the heat transfer problem is very similar to the mechanics problem in terms of FEM treatment. The displacement and force correspond to the temperature and heat flux, respectively. We also showed the analogies of some of the element matrices and vectors between heat transfer problems and mechanics problems. However, we did not mention the mechanics counterparts of matrix $\mathbf{k}_M^{(e)}$ and vector $\mathbf{f}_S^{(e)}$ that come from the heat convection boundary. Since the heat flux on the boundary depends upon the unknown field variable (temperature), the resulting term ($\mathbf{k}_M^{(e)}$) needs to be combined with the heat convection matrix. In fact, in mechanics problems, we can also have a similar situation when the structure is supported by elastic supports such as springs. The reaction force on the boundary depends upon the unknown field variable (displacement) at the boundary. For such a mechanics system, we will have an additional matrix corresponding to $\mathbf{k}_M^{(e)}$ and a vector corresponding to $\mathbf{f}_S^{(e)}$. However, in mechanics problems, it is often more convenient to formulate such an elastic support using an additional element. The stiffness of the support is then assembled to the global stiffness in the same systematic direct assembly procedure. The reaction force of the elastic support is found after the global equations system is solved for the displacements.

We conclude that the Galerkin residual formulation gives exactly the same FE equations as those we obtained using the energy principles discussed in previous chapters.

12.3.5 Composite Wall

Consider heat transfer through a composite wall, as shown in Figure 12.10. The governing equation is given by Eq. (12.9). If there is no heat source or sink in a layer ($q = 0$ within the layer), one linear element is enough for modelling the entire layer (why?)[1]. For the case shown in Figure 12.10, a total of three elements, one for each layer, should be used if there is no heat source within these layers. This argument holds even if there is no heat source or sink in the interface between the layers.

As for the boundary conditions, at any one or both of the outer surfaces, the temperature or heat convection or heat insulation could be specified. The convective boundary conditions are given as

$$kA\frac{\mathrm{d}\phi}{\mathrm{d}x} = hA(\phi_b - \phi_f) \quad \text{at } x = 0 \tag{12.86}$$

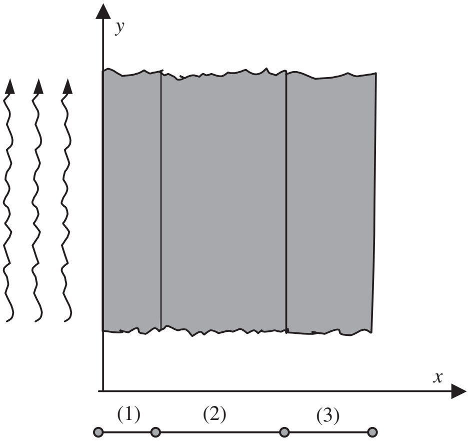


Figure 12.10. Heat transfer through a composite wall of three layers. One element is sufficient to model a layer if there is no heat supply/sink in the layers.

[1] Hint: see Example 4.1.

and

$$-kA\frac{\mathrm{d}\phi}{\mathrm{d}x} = hA(\phi_b - \phi_f) \quad \text{at } x = H \tag{12.87}$$

Therefore, all the elements developed in the section for the 1D fin are also valid for the case of the composite wall, except that $\mathbf{k}_G^{(e)}$ and $\mathbf{f}_Q^{(e)}$ do not exist because the g and Q vanish in the case of a composite wall (see Eq. (12.10)).

The general form of the element stiffness matrix is given by

$$\mathbf{k}^{(e)} = \frac{kA}{l}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} + \mathbf{k}_M^{(e)} \tag{12.88}$$

where $\mathbf{k}_M^{(e)}$ is obtained either by Eqs. (12.52) or (12.55), if there is heat convection occurring on the surface, or else $\mathbf{k}_M^{(e)} = 0$ if the surface is insulated. As for a surface with a specified temperature, there is no need to calculate $\mathbf{k}_M^{(e)}$. For the force vector, $\mathbf{f}_S^{(e)}$, it is obtained either by Eqs. (12.53) or (12.56) if there is heat convection occurring on the surface, or zero if the surface is insulated. In the case where the surface has a specific temperature, it is not necessary to have $\mathbf{f}_S^{(e)}$ for calculation of the temperature. It can always be calculated after the temperature field has been found.

12.3.6 Worked Example

Example 12.2: Heat transfer through a composite wall of two layers

Figure 12.11 shows a composite wall consisting of two layers of different materials. Various heat transfer parameters are shown in the figure. The temperature distribution through the composite wall is to be calculated by the finite element method.

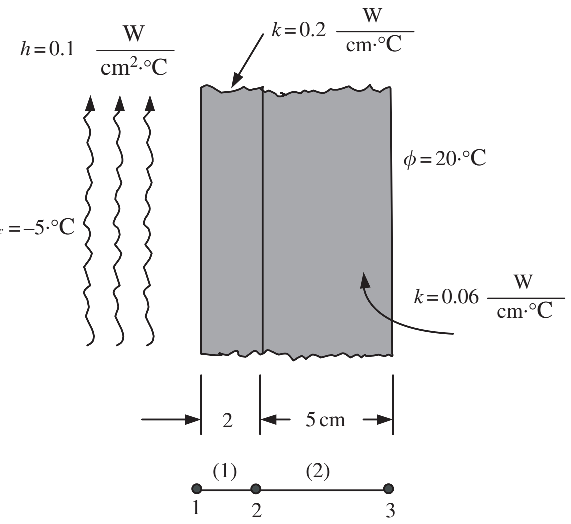

$h=0.1\ \frac{\mathrm{W}}{\mathrm{cm}^2\cdot{}^\circ\mathrm{C}}$
$k=0.2\ \frac{\mathrm{W}}{\mathrm{cm}\cdot{}^\circ\mathrm{C}}$
$\phi=20\cdot{}^\circ\mathrm{C}$
$\phi_f=-5\cdot{}^\circ\mathrm{C}$
$k=0.06\ \frac{\mathrm{W}}{\mathrm{cm}\cdot{}^\circ\mathrm{C}}$
2
5 cm
(1)
(2)
1
2
3

Figure 12.11. Heat transfer through a composite wall of two layers.

Analysis of problem. The wall is divided into two elements with a total of three nodes. Hence, the system equation should be 3 × 3. At node 3, the temperature is specified, therefore there is no need to calculate $\mathbf{k}_M^{(e)}$ and $\mathbf{f}_S^{(e)}$ for element 2. Since the heat convection is occurring on node 1, $\mathbf{k}_M^{(e)}$ and $\mathbf{f}_S^{(e)}$ have to be calculated using Eqs. (12.55) and (12.56), respectively.

Data preparation. For element (1),

$$\frac{kA}{L} = \frac{0.2(1)}{2} = 0.1\,\frac{\mathrm{W}}{°\mathrm{C}} \tag{12.89}$$

$$hA = 0.1(1) = 0.1\,\frac{\mathrm{W}}{°\mathrm{C}} \tag{12.90}$$

$$hA\phi_f = 0.1(1)(-5) = -0.5\,\mathrm{W} \tag{12.91}$$

For element (2),

$$\frac{kA}{L} = \frac{0.06(1)}{5} = 0.012\,\frac{\mathrm{W}}{°\mathrm{C}} \tag{12.92}$$

Solution. The element matrices for element (1) are

$$\mathbf{k}^{(1)} = \begin{bmatrix} 0.2 & -0.1 \\ -0.1 & 0.1 \end{bmatrix} \quad \text{and} \quad \mathbf{f}_S^{(1)} = \begin{Bmatrix} -0.5 \\ 0 \end{Bmatrix} \tag{12.93}$$

The element matrix for element (2) is

$$\mathbf{k}^{(2)} = \begin{bmatrix} 0.012 & -0.012 \\ -0.012 & 0.012 \end{bmatrix} \tag{12.94}$$

The next step is to assemble the element matrices of the two elements together to form the global system equations, which leads to

$$\begin{bmatrix} 0.20 & -0.10 & 0 \\ -0.10 & 0.112 & -0.012 \\ 0 & -0.012 & 0.012 \end{bmatrix} \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3(=20) \end{Bmatrix} = \begin{Bmatrix} -0.5 \\ 0 \\ 0 \end{Bmatrix} + \begin{Bmatrix} 0 \\ 0 \\ Q^* \end{Bmatrix} \tag{12.95}$$

Note again that Q^* is as yet unknown, but is required to balance the equation, as the temperature at node 3 is fixed. Having only two unknowns of temperature, the first two equations in the above give

$$\begin{bmatrix} 0.20 & -0.10 \\ -0.10 & 0.112 \end{bmatrix} \begin{Bmatrix} \phi_1 \\ \phi_2 \end{Bmatrix} = \begin{Bmatrix} -0.5 \\ 0.24(= 20 \times 0.012) \end{Bmatrix} \tag{12.96}$$

The above matrix equation (actually consisting of two simultaneous equations) is solved, and the solution is given as

$$\mathbf{\Phi}^T = \{-2.5806 \quad -0.1613 \quad 20\} \tag{12.97}$$

Example 12.3: Heat transfer through layers of thin films

Figure 12.12 shows the process of producing thin film layers of different materials on a substrate using physical deposition techniques. A heat supply is provided on the upper surface of the glass substrate. At the stage shown in Figure 12.12, a layer of iron and a layer of platinum have been formed. The thickness of these layers and the thermal conductivities for these materials of these layers are also shown in the figure. Convection heat loss occurs on the lower platinum surface, and the ambient temperature is 150°C. A heat supply is provided to maintain the temperature on the upper surface of the glass substrate at 300°C. The temperature distribution through the thickness of the layers of the thin film system is to be calculated by the finite element method.

Analysis of problem. This problem is actually similar to the previous example on the composite wall. Hence, 1D elements can be used for this purpose. The layers are divided into three elements with a total of four nodes, as shown in Figure 12.12. Since the temperature is specified at node 1, there is no need to calculate $\mathbf{k}_M^{(e)}$ and $\mathbf{f}_S^{(e)}$ for element 1. $\mathbf{k}_M^{(e)}$ and $\mathbf{f}_S^{(e)}$ for element 2 is zero, since there is no heat convection occurring at either nodes 2 or 3. Since there is heat convection occurring at node 4, $\mathbf{k}_M^{(e)}$ and $\mathbf{f}_S^{(e)}$ have to be calculated using Eqs. (12.52) and (12.53), respectively.

Data preparation. For element (1),

$$\frac{kA}{L} = \frac{(0.1)(1)}{0.2} = 0.5\frac{\text{W}}{°\text{C}} \tag{12.98}$$

For element (2),

$$\frac{kA}{L} = \frac{(0.5)(1)}{0.02} = 25\frac{\text{W}}{°\text{C}} \tag{12.99}$$

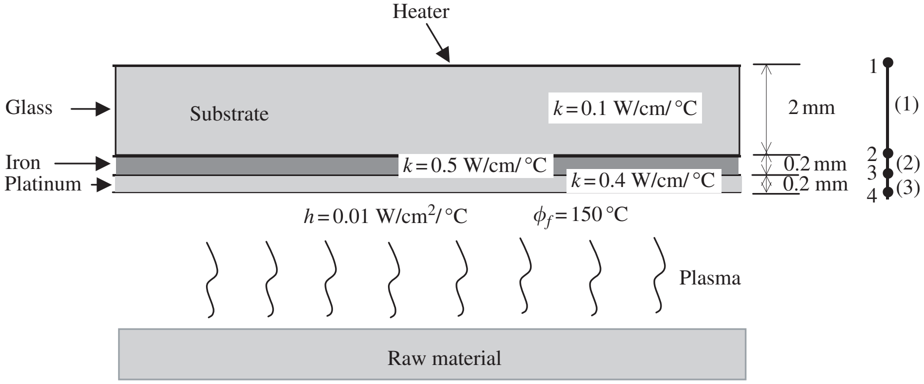


Figure 12.12. Heat transfer during a thin film deposition process.

For element (3),

$$\frac{kA}{L} = \frac{0.4(1)}{0.02} = 20\,\frac{\mathrm{W}}{^\circ\mathrm{C}} \tag{12.100}$$

$$hA = 0.01(1) = 0.01\,\frac{\mathrm{W}}{^\circ\mathrm{C}} \tag{12.101}$$

$$hA\phi_f = 0.01(1)(150) = 1.5\,\mathrm{W} \tag{12.102}$$

Solution. The element matrix for element (1) is

$$\mathbf{k}^{(1)} = \begin{bmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{bmatrix} \tag{12.103}$$

The element matrix for element (2) is

$$\mathbf{k}^{(2)} = \begin{bmatrix} 25 & -25 \\ -25 & 25 \end{bmatrix} \tag{12.104}$$

And finally, the element matrices for element (3) are

$$\mathbf{k}^{(3)} = \begin{bmatrix} 20 & -20 \\ -20 & 20.01 \end{bmatrix} \quad \text{and} \quad \mathbf{f}_S^{(3)} = \begin{Bmatrix} 0 \\ 1.5 \end{Bmatrix} \tag{12.105}$$

The next step is to assemble the element matrices of the two elements together to form the global system equations, which leads to

$$\begin{bmatrix} 0.5 & -0.5 & 0 & 0 \\ -0.5 & 25.5 & -25 & 0 \\ 0 & -25 & 45 & -20 \\ 0 & 0 & -20 & 20.01 \end{bmatrix} \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{Bmatrix} = \begin{Bmatrix} Q^* \\ 0 \\ 0 \\ 1.5 \end{Bmatrix} \tag{12.106}$$

Since ϕ_1 is given to be 300°C, we can therefore reduce the above equation to a 3×3 matrix to solve for the remaining three unknown temperatures:

$$\begin{bmatrix} 25.5 & -25 & 0 \\ -25 & 45 & -20 \\ 0 & -20 & 20.01 \end{bmatrix} \begin{Bmatrix} \phi_2 \\ \phi_3 \\ \phi_4 \end{Bmatrix} = \begin{Bmatrix} 150 \\ 0 \\ 1.5 \end{Bmatrix} \tag{12.107}$$

The above matrix equation is solved, and the solution is given as

$$\mathbf{\Phi}^T = \begin{bmatrix} 300.0 & 297.1 & 297.0 & 296.9 \end{bmatrix} \tag{12.108}$$

To calculate the heat flux, Q^* on the top of the glass substrate, we can now substitute the temperatures into the first equation of the matrix equation in Eq. (12.106) to obtain

$$Q^* = 0.5 \times 300 - 0.5 \times 297.1 = 1.45\,\mathrm{W/cm^2} \tag{12.109}$$

12.4 2D HEAT TRANSFER PROBLEM

12.4.1 Element Equations

This section deals with heat transfer problems in two-dimensions that is governed by Eq. (12.1). The procedure for obtaining the FEM equations for 2D heat transfer problems is the same as that for the 1D problems described in the previous sections, except that the mathematical manipulation is more tedious due to the additional dimension.

Let us assume that the problem domain is divided into elements, as shown in Figure 12.5. For one element in general, the residual can be obtained by the Galerkin method as

$$\mathbf{R}^{(e)} = -\int_{A_e} \mathbf{N}^T \left(D_x \frac{\partial^2 \phi^h}{\partial x^2} + D_y \frac{\partial^2 \phi^h}{\partial y^2} - g\phi^h + Q \right) \mathrm{d}A \tag{12.110}$$

Note that the minus sign is added to the residual mainly for convenience. The integration in Eq. (12.110) for the residual must be evaluated so as to obtain the element matrices, but in this case, the integration is much more complex than the 1D case because the integration is performed over the area of the element. Recall that in the 1D case, the integral is evaluated by parts, but in this 2D case we need to use Gauss's divergence theorem instead.

Using the product rule for differentiation first, the following expression can be obtained:

$$\frac{\partial}{\partial x}\left(\mathbf{N}^T \frac{\partial \phi}{\partial x}\right) = \mathbf{N}^T \frac{\partial^2 \phi}{\partial x^2} + \frac{\partial \mathbf{N}^T}{\partial x}\frac{\partial \phi}{\partial x} \tag{12.111}$$

The first integral in Eq. (12.110) can then be obtained by

$$-\int_{A_e} \mathbf{N}^T D_x \frac{\partial^2 \phi}{\partial x^2}\,\mathrm{d}A = -\int_{A_e} D_x \frac{\partial}{\partial x}\left(\mathbf{N}^T \frac{\partial \phi}{\partial x}\right)\mathrm{d}A + \int_{A_e} D_x \frac{\partial \mathbf{N}^T}{\partial x}\frac{\partial \phi}{\partial x}\,\mathrm{d}A \tag{12.112}$$

where A_e is the area of the element. Gauss's divergence theorem can be stated mathematically for this case as

$$\int_{A_e} \frac{\partial}{\partial x}\left(\mathbf{N}^T \frac{\partial \phi}{\partial x}\right)\mathrm{d}A = \int_{\Gamma_e} \mathbf{N}^T \frac{\partial \phi}{\partial x}\cos\theta\,\mathrm{d}\Gamma \tag{12.113}$$

where θ is the angle of the outwards normal on the boundary Γ_e of the element with respect to the x-axis. Equation (12.113) is thus substituted into Eq. (12.112) to obtain

$$-\int_{A} \mathbf{N}^T D_x \frac{\partial^2 \phi}{\partial x^2}\,\mathrm{d}A = -\int_{\Gamma_e} D_x \mathbf{N}^T \frac{\partial \phi}{\partial x}\cos\theta\,\mathrm{d}\Gamma + \int_{A} D_x \frac{\partial \mathbf{N}^T}{\partial x}\frac{\partial \phi}{\partial x}\,\mathrm{d}A \tag{12.114}$$

In a similar way, the second integral in Eq. (12.110) can be evaluated to obtain

$$-\int_{A} \mathbf{N}^T D_y \frac{\partial^2 \phi}{\partial y^2}\,\mathrm{d}A = -\int_{\Gamma_e} D_y \mathbf{N}^T \frac{\partial \phi}{\partial y}\sin\theta\,\mathrm{d}\Gamma + \int_{A} D_y \frac{\partial \mathbf{N}^T}{\partial y}\frac{\partial \phi}{\partial y}\,\mathrm{d}A \tag{12.115}$$

The two integrals in Eqs. (12.114) and (12.115) are substituted back into the residual in Eq. (12.110) to give

$$\begin{aligned}\mathbf{R}^{(e)} &= -\int_{\Gamma_e} \mathbf{N}^T \left(D_x \frac{\partial \phi^h}{\partial x} \cos\theta + D_y \frac{\partial \phi^h}{\partial y} \sin\theta \right) \mathrm{d}\Gamma \\ &+ \int_{A_e} \left(D_x \frac{\partial \mathbf{N}^T}{\partial x} \frac{\partial \phi^h}{\partial x} + D_y \frac{\partial \mathbf{N}^T}{\partial y} \frac{\partial \phi^h}{\partial y} \right) \mathrm{d}A \\ &+ \int_{A_e} g \mathbf{N}^T \phi^h \, \mathrm{d}A - \int_{A_e} Q \mathbf{N}^T \, \mathrm{d}A \end{aligned} \tag{12.116}$$

The field variable ϕ is now interpolated from the nodal variables by shape functions as in Eq. (12.34), which is then substituted into Eq. (12.116) to give

$$\begin{aligned}\mathbf{R}^{(e)} &= \underbrace{-\int_{\Gamma_e} \mathbf{N}^T \left(D_x \frac{\partial \phi^h}{\partial x} \cos\theta + D_y \frac{\partial \phi^h}{\partial y} \sin\theta \right) \mathrm{d}\Gamma}_{\mathbf{b}^{(e)}} \\ &+ \underbrace{\left(\int_{A_e} \left(D_x \frac{\partial \mathbf{N}^T}{\partial x} \frac{\partial \mathbf{N}}{\partial x} + D_y \frac{\partial \mathbf{N}^T}{\partial y} \frac{\partial \mathbf{N}}{\partial y} \right) \mathrm{d}A \right)}_{\mathbf{k}_D^{(e)}} \mathbf{\Phi}^{(e)} \\ &+ \underbrace{\left(\int_{A_e} g \mathbf{N}^T \mathbf{N} \, \mathrm{d}A \right)}_{\mathbf{k}_g^{(e)}} \mathbf{\Phi}^{(e)} - \underbrace{\int_{A_e} Q \mathbf{N}^T \, \mathrm{d}A}_{\mathbf{f}_Q^{(e)}} \end{aligned} \tag{12.117}$$

or in matrix form,

$$\mathbf{R}^{(e)} = \mathbf{b}^{(e)} + \left[\mathbf{k}_D^{(e)} + \mathbf{k}_g^{(e)} \right] \mathbf{\Phi}^{(e)} - \mathbf{f}_Q^{(e)} \tag{12.118}$$

where

$$\mathbf{b}^{(e)} = -\int_{\Gamma_e} \mathbf{N}^T \left(D_x \frac{\partial \phi^h}{\partial x} \cos\theta + D_y \frac{\partial \phi^h}{\partial y} \sin\theta \right) \mathrm{d}\Gamma \tag{12.119}$$

$$\mathbf{k}_D^{(e)} = \int_{A_e} \left(\frac{\partial \mathbf{N}^T}{\partial x} D_x \frac{\partial \mathbf{N}}{\partial x} + \frac{\partial \mathbf{N}^T}{\partial y} D_y \frac{\partial \mathbf{N}}{\partial y} \right) \mathrm{d}A \tag{12.120}$$

$$\mathbf{k}_g^{(e)} = \int_{A_e} g \mathbf{N}^T \mathbf{N} \, \mathrm{d}A \tag{12.121}$$

$$\mathbf{f}_Q^{(e)} = \int_{A_e} Q \mathbf{N}^T \, \mathrm{d}A \tag{12.122}$$

As in the 1D case, the vector $\mathbf{b}^{(e)}$ is related to the derivatives of temperature (heat flux) on the boundaries of the element. It will be evaluated in the next section in detail. For now, Eqs. (12.120)–(12.122) will be evaluated and analysed.

The integral in Eq. (12.120) can be rewritten in the matrix form by defining

$$\mathbf{D} = \begin{bmatrix} D_x & 0 \\ 0 & D_y \end{bmatrix} \tag{12.123}$$

and the gradient vector as

$$\nabla\phi = \begin{Bmatrix} \dfrac{\partial\phi}{\partial x} \\ \dfrac{\partial\phi}{\partial y} \end{Bmatrix} = \begin{bmatrix} \dfrac{\partial\mathbf{N}}{\partial x} \\ \dfrac{\partial\mathbf{N}}{\partial y} \end{bmatrix} \boldsymbol{\Phi}^{(e)} = \mathbf{B}\boldsymbol{\Phi}^{(e)} \tag{12.124}$$

where **B** is the strain matrix given by

$$\mathbf{B} = \begin{bmatrix} \dfrac{\partial\mathbf{N}}{\partial x} \\ \dfrac{\partial\mathbf{N}}{\partial y} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial N_1}{\partial x} & \dfrac{\partial N_2}{\partial x} & \cdots & \dfrac{\partial N_{n_d}}{\partial x} \\ \dfrac{\partial N_1}{\partial y} & \dfrac{\partial N_2}{\partial y} & \cdots & \dfrac{\partial N_{n_d}}{\partial y} \end{bmatrix} \tag{12.125}$$

Note that to obtain the above equation, the usual shape function given by Eq. (12.28) is utilized. Using Eqs. (12.123)–(12.125), it can be easily verified that

$$\mathbf{B}^T\mathbf{D}\mathbf{B} = D_x \frac{\partial\mathbf{N}^T}{\partial x}\frac{\partial\mathbf{N}}{\partial x} + D_y \frac{\partial\mathbf{N}^T}{\partial y}\frac{\partial\mathbf{N}}{\partial y} \tag{12.126}$$

Therefore, the general element ‘stiffness’ matrix for 2D elements given by Eq. (12.120) becomes

$$\mathbf{k}_D^{(e)} = \int_{A_e} \mathbf{B}^T\mathbf{D}\mathbf{B}\,\mathrm{d}A \tag{12.127}$$

which is exactly the same as Eq. (3.71) that is obtained using Hamilton’s principle, except that the matrix of material elasticity is replaced by the matrix of heat conductivity. Note also that Eq. (12.121) $\mathbf{k}_g^{(e)}$ is the same as the matrix given by Eq. (3.75) for mechanics problems. We observe again that the Galerkin weighted residual formulation produces the same set of FE equations as those produced by the energy principle.

12.4.2 Triangular Elements

Using shape functions of a triangular element, the field function of temperature, ϕ, can be interpolated as follows:

$$\phi^{(e)} = \mathbf{N}\boldsymbol{\Phi}^{(e)} = [N_1 \quad N_2 \quad N_3] \begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \end{Bmatrix} \tag{12.128}$$

where N_i $(i = 1, 2, 3)$ are the three shape functions given by Eq. (7.22), and ϕ_i $(i = 1, 2, 3)$ are the nodal values of temperature at the three-nodes of the triangular element shown in Figure 12.13.

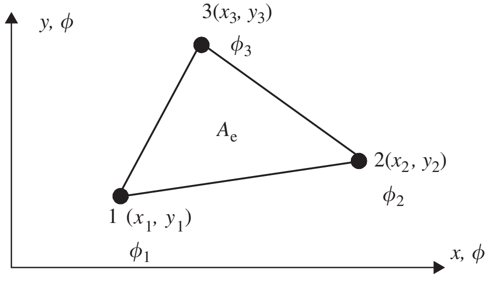


Figure 12.13. Linear triangular element.

Note that the strain matrix **B** is constant for triangular elements, and is given by Eq. (7.38). Using Eq. (12.127), $\mathbf{k}_D^{(e)}$ can be evaluated easily as the integrand is a constant matrix if the material constants D_x and D_y do not change within the element:

$$\mathbf{k}_D^{(e)} = \int_{A_e} \mathbf{B}^T\mathbf{D}\mathbf{B}\,\mathrm{d}A = \mathbf{B}^T\mathbf{D}\mathbf{B}\int_{A_e}\mathrm{d}A = \mathbf{B}^T\mathbf{D}\mathbf{B}A_e \tag{12.129}$$

Expanding the matrix product yields

$$\mathbf{k}_D^{(e)} = \frac{D_x}{4A}\begin{bmatrix} b_i^2 & b_ib_j & b_ib_k \\ b_ib_j & b_j^2 & b_jb_k \\ b_ib_k & b_jb_k & b_k^2 \end{bmatrix} + \frac{D_y}{4A}\begin{bmatrix} c_i^2 & c_ic_j & c_ic_k \\ c_ic_j & c_j^2 & c_jc_k \\ c_ic_k & c_jc_k & c_k^2 \end{bmatrix} \tag{12.130}$$

It is noted that the stiffness matrix is symmetrical.

The matrix, $\mathbf{k}_g^{(e)}$ defined by Eq. (12.121) can be evaluated as

$$\mathbf{k}_g^{(e)} = \int_{A_e} g\mathbf{N}^T\mathbf{N}\,\mathrm{d}A = g\int_{A_e}\begin{Bmatrix} N_1 \\ N_2 \\ N_3 \end{Bmatrix}[N_1 \quad N_2 \quad N_3]\,\mathrm{d}A$$

$$= g\int_{A_e}\begin{bmatrix} N_1^2 & N_1N_2 & N_1N_3 \\ N_1N_2 & N_2^2 & N_2N_3 \\ N_1N_3 & N_2N_3 & N_3^2 \end{bmatrix}\mathrm{d}A \tag{12.131}$$

The above integral is carried out using the following factorial formula Eq. (7.43), and the fact that the area coordinates are the same as the shape functions, just as we have done for the mass matrix, Eq. (7.44), in Chapter 7. For example,

$$\int_{A_e} N_1N_2\,\mathrm{d}A = \int_{A_e} L_1^1L_2^1L_3^0\,\mathrm{d}A = \frac{1!1!0!}{(1+1+0+2)!}2A = \frac{1}{4\times 3\times 2\times 1}2A = \frac{A}{12} \tag{12.132}$$

Using the area coordinates and the factorial formula in Eq. (7.43), the matrix $\mathbf{k}_g^{(e)}$ is found as

$$\mathbf{k}_g^{(e)} = \frac{gA}{12}\begin{bmatrix} 2 & 1 & 1 \\ 1 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix} \tag{12.133}$$

The element force vector $\mathbf{f}_Q^{(e)}$ defined in Eq. (12.122) also involves the integration of shape functions, and can also be obtained using the factorial formula in Eq. (7.43):

$$\mathbf{f}_Q^{(e)} = \int_{A_e} Q\mathbf{N}^T \,\mathrm{d}A = Q\int_{A_e} \begin{Bmatrix} N_i \\ N_j \\ N_k \end{Bmatrix} \mathrm{d}A = Q\int_{A_e} \begin{Bmatrix} L_1 \\ L_2 \\ L_3 \end{Bmatrix} \mathrm{d}A = \frac{QA}{3}\begin{Bmatrix} 1 \\ 1 \\ 1 \end{Bmatrix} \tag{12.134}$$

It is assumed that Q is a constant within the element. The value of QA is equally shared by the three nodes of the rectangular element.

12.4.3 Rectangular Elements

Consider now a four-node, rectangular element as shown in Figure 7.8. The field function ϕ is interpolated over the element as follows:

$$\phi^{(e)} = \mathbf{N}\mathbf{\Phi}^{(e)} = [N_1 \quad N_2 \quad N_3 \quad N_4]\begin{Bmatrix} \phi_1 \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{Bmatrix} \tag{12.135}$$

Note that for rectangular elements, the natural coordinate system is again adopted, as in mechanics problems (Figure 7.8). The shape functions are given by Eq. (7.51), and are known as bilinear shape functions. The strain matrix for the rectangular element is given by Eq. (7.55). Note that for bilinear elements, the strain matrix is no longer constant. Using Eqs. (12.127), (7.51) and (7.55), $\mathbf{k}_D^{(e)}$ can be evaluated as

$$\begin{aligned}\mathbf{k}_D^{(e)} &= \int_{A_e} \mathbf{B}^T\mathbf{D}\mathbf{B}\,\mathrm{d}A = \int_{-1}^{+1}\int_{-1}^{+1} ab\mathbf{B}^T\mathbf{D}\mathbf{B}\,\mathrm{d}\xi\,\mathrm{d}\eta \\ &= \frac{D_x b}{6a}\begin{bmatrix} 2 & -2 & -1 & 1 \\ -2 & 2 & 1 & -1 \\ -1 & 1 & 2 & -2 \\ 1 & -1 & -2 & 2 \end{bmatrix} + \frac{D_y a}{6b}\begin{bmatrix} 2 & 1 & -1 & -2 \\ 1 & 2 & -2 & -1 \\ -1 & -2 & 2 & 1 \\ 1 & -1 & 1 & 2 \end{bmatrix}\end{aligned} \tag{12.136}$$

The matrix $\mathbf{k}_g^{(e)}$ defined by Eq. (12.121) can be evaluated as

$$\begin{aligned}\mathbf{k}_g^{(e)} &= \int_{A_e} g\mathbf{N}^T\mathbf{N}\,\mathrm{d}A = \int_{-1}^{+1}\int_{-1}^{+1} abg\mathbf{N}^T\mathbf{N}\,\mathrm{d}\xi\,\mathrm{d}\eta \\ &= abg\int_{A_e}\begin{bmatrix} N_1^2 & N_1N_2 & N_1N_3 & N_1N_4 \\ N_1N_2 & N_2^2 & N_2N_3 & N_2N_3 \\ N_1N_3 & N_2N_3 & N_3^2 & N_3N_4 \\ N_1N_4 & N_2N_4 & N_3N_4 & N_4^2 \end{bmatrix}\mathrm{d}\xi\,\mathrm{d}\eta\end{aligned} \tag{12.137}$$

which results in

$$\mathbf{k}_g^{(e)} = \frac{gA}{36}\begin{bmatrix} 4 & 2 & 1 & 2 \\ 2 & 4 & 2 & 1 \\ 1 & 2 & 4 & 2 \\ 2 & 1 & 2 & 4 \end{bmatrix} \tag{12.138}$$

The element force vector, $\mathbf{f}_Q^{(e)}$, defined in Eq. (12.122) also involves the integration of the shape functions, and with substitution of the shape functions, Eq. (7.51), it can be obtained as

$$\mathbf{f}_Q^{(e)} = \int_{A_e} QN^T \, \mathrm{d}A = Q \int_{A_e} \begin{Bmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \end{Bmatrix} \mathrm{d}A = \frac{QA}{4} \begin{Bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{Bmatrix} \tag{12.139}$$

Note that in the above, Q is assumed to be constant within the element. The value of QA is equally shared by the four nodes of the rectangular element.

In this subsection, matrices $\mathbf{k}_D^{(e)}$, $\mathbf{k}_g^{(e)}$, $\mathbf{f}_Q^{(e)}$ have been evaluated exactly and explicitly for rectangular elements. In engineering practice, however, it is very rare to use rectangular elements unless the geometry of the problem domain is also a rectangular one. Very often, quadrilateral elements with four nodes and four non-parallel sides are used for the complex geometry of the problem domain. Formulating FEM equations for quadrilateral elements has been detailed in Section 7.4. Note that with quadrilateral elements, it is difficult to work out the exact explicit form of the element matrices. Therefore, the integrals are carried out in most commercial software packages using numerical integral schemes such as the Gauss integration scheme discussed in Chapter 7 for 2D solid elements.

12.4.4 Boundary Conditions and Vector $\mathbf{b}^{(e)}$

Previously, it is mentioned that the vector, $\mathbf{b}^{(e)}$, for the 2D element as defined by Eq. (12.119) is associated with the derivatives of temperature (or heat flux) on the boundaries of the element. In this section, the relationship of vector, $\mathbf{b}^{(e)}$, with the boundaries of the element and hence the boundaries of the problem domain, will be studied in detail.

The vector $\mathbf{b}^{(e)}$ defined in Eq. (12.119) is first split into two parts:

$$\mathbf{b}^{(e)} = \mathbf{b}_I^{(e)} + \mathbf{b}_B^{(e)} \tag{12.140}$$

where $\mathbf{b}_I^{(e)}$ comes from integration of the element boundaries lying inside the problem domain, and $\mathbf{b}_B^{(e)}$ is that which lies on the boundary of the problem domain. It can then be proven that $\mathbf{b}_I^{(e)}$ should vanish, which we have seen for the one-dimensional case.

Figure 12.14 shows two adjacent elements numbered, for example, 1 and 2. In evaluating the vector, $\mathbf{b}^{(e)}$ as defined in Eq. (12.119), the integration needs to be done on all the edges of these elements. As Eq. (12.119) involves a line integral, the results will be direction-dependent. The direction of integration has to be consistent for all the elements in the system, either clockwise or counter-clockwise. For elements 1 and 2, their directions of integration are assumed to be counter-clockwise, as shown by arrows in Figure 12.14. Note that on their common edge j–k, the value of $\mathbf{b}_I^{(e)}$ obtained for element 2 is the same as that obtained for element 1, except that their signs are opposite because the direction of integration on this edge of both elements are opposite. Therefore, when these elements are assembled together, values of $\mathbf{b}_I^{(e)}$ will cancel each other out and vanish. This happens for all other edges of all the elements in the interior of the problem domain. Therefore, when the edge lies on the boundary of the problem domain, $\mathbf{b}_B^{(e)}$ has to be evaluated.

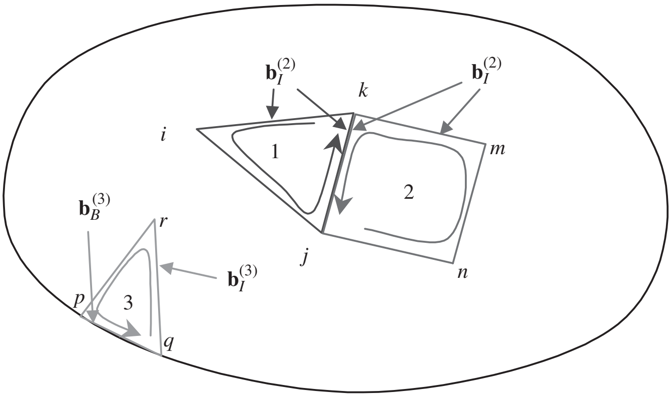


Figure 12.14. Direction of integration path for evaluating $\boldsymbol{b}^{(e)}$. For element edges that are located in the interior of the problem domain, $\boldsymbol{b}^{(e)}$ vanishes after assembly of the elements, because the values of $\boldsymbol{b}^{(e)}$ obtained for the same edge of the two adjacent elements possess opposite signs.

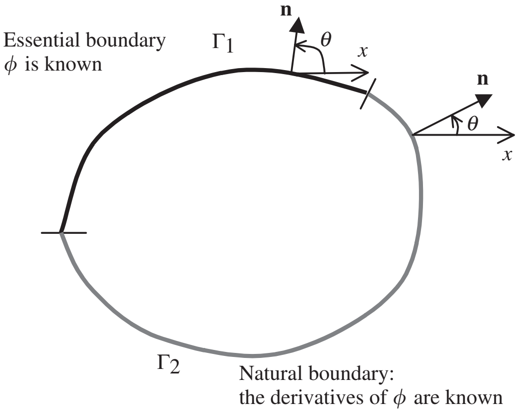


Figure 12.15. Types of boundary conditions. Γ_1: essential boundary where the temperature is known; Γ_2: natural boundary where the heat flux (derivative of temperature) is known.

The boundary of the problem domain can be divided broadly into two categories. One is the boundary where the field variable temperature ϕ is specified, as noted by Γ_1 in Figure 12.15, which is known as the *essential boundary condition*. The other is the boundary where the derivatives of the field variable of temperature (heat flux) are specified, as shown in Figure 12.15. This second type of boundary condition is known as the *natural boundary condition*. For the essential boundary, we need not evaluate $\mathbf{b}_B^{(e)}$ at the stage of formulating and solving the FEM equations, as the temperature is already known, and the corresponding columns and rows will be removed from the global FEM equations. We have seen such a treatment in examples such as Example 12.1. Because $\mathbf{b}^{(e)}$ is derived *naturally* from the weighted residual weak form, it relates to the *natural boundary condition*. Therefore, our concern is only for elements that are on the natural boundaries, where the derivatives of the field variable are specified, and special methods of evaluating the integral are required, as in the 1D case (Example 12.2).

In heat transfer problems, the natural boundary often refers to a boundary where heat convection occurs. The integrand in Eq. (12.119) can be generally rewritten in the form

$$D_x \frac{\partial \phi^h}{\partial x} \cos\theta + D_y \frac{\partial \phi^h}{\partial y} \sin\theta = -M\phi_b + S \quad \text{on natural boundary } \Gamma_2 \tag{12.141}$$

where θ is the angle of the outwards normal on the boundary with respect to the x-axis, and M and S are given constants depending on the type of the natural boundaries, and ϕ_b is the unknown temperature on the boundary. Note that the left-hand side of Eq. (12.141) is in fact the heat flux across the boundary; it can therefore be re-written as

$$D_x \frac{\partial \phi^h}{\partial x} \cos\theta + D_y \frac{\partial \phi^h}{\partial y} \sin\theta = k\frac{\partial \phi^h}{\partial n} = -M\phi_b + S \quad \text{on } \Gamma_2 \tag{12.142}$$

where k is the heat conductivity at the boundary point in the direction of the boundary normal. For heat transfer problems, there are the following types of boundary conditions:

- ***Heat insulation boundary:*** on the boundary where the heat is insulated from heat exchange, there will be no heat flux across the boundary and the derivatives of temperature there will be zero. In such cases, we have $M = S = 0$, and the value of $\mathbf{b}_B^{(e)}$ is simply zero.
- ***Convective boundary condition:*** Figure 12.16 shows the situation whereby there are exchanges of heat via convection. Following the Fourier's heat convection flow, the heat flux across the boundary due to the heat conduction can be given by

$$q_k = -k\frac{\partial \phi}{\partial n} \tag{12.143}$$

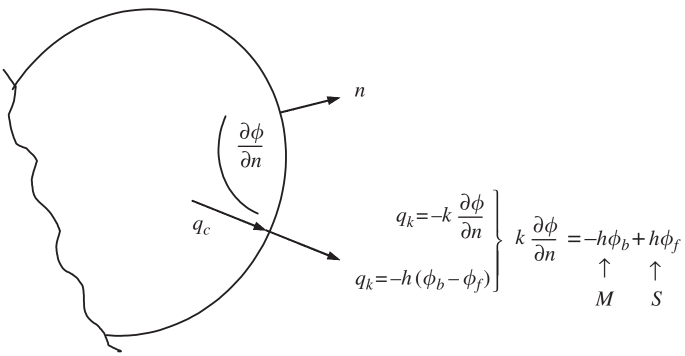

n
$\frac{\partial \phi}{\partial n}$
q_c
$q_k = -k\frac{\partial \phi}{\partial n}$
$q_k = -h(\phi_b - \phi_f)$
$k\frac{\partial \phi}{\partial n} = -h\phi_b + h\phi_f$
↑ ↑
M S

Figure 12.16. Heat convection on the boundary.

where k is the heat conductivity at the boundary point in the direction of the boundary normal. On the other hand, following the Fourier's heat convection law, the heat flux across the boundary due to the heat convection can be given by

$$q_h = h(\phi_b - \phi_f) \tag{12.144}$$

where h is the heat convection coefficient at the boundary point in the direction of the boundary normal. At the same boundary point the heat flux by conduction should be the same as that by convection, i.e. $q_k = q_h$, which leads to

$$k\frac{\partial \phi}{\partial n} = -\underbrace{h\phi_b}_{M} + \underbrace{h\phi_f}_{S} \tag{12.145}$$

The values of M and S for the heat convection boundary are then found to be

$$M = h, \quad S = h\phi_f \tag{12.146}$$

- ***Specified heat flux on boundary:*** when there is a heat flux specified on the boundary, as shown in Figure 12.17. The heat flux across the boundary due to the heat conduction can be given by Eq. (12.143). The heat flux by conduction should be the same as the specified heat flux, i.e. $q_k = q_s$, which leads to

$$k\frac{\partial \phi}{\partial n} = \underbrace{0\times}_{M}\ \phi_b \underbrace{-q_s}_{S} \tag{12.147}$$

The values of M and S for the heat convection boundary are then found to be

$$M = 0, \quad S = -q_s \tag{12.148}$$

From Figure 12.17, it can be seen that

$$S = \begin{cases} \text{Positive} & \text{if heat flows into the boundary} \\ \text{Negative} & \text{if heat flows out of the boundary} \\ 0 & \text{insulated} \end{cases} \tag{12.149}$$

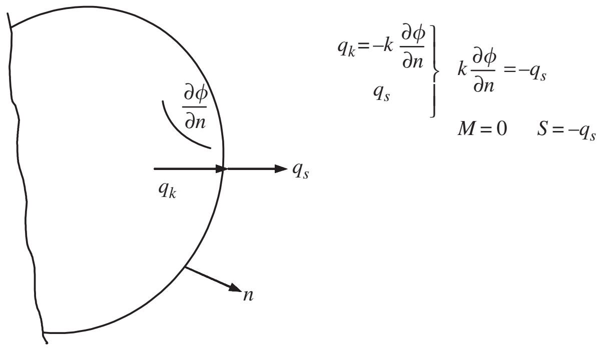

$\frac{\partial \phi}{\partial n}$
q_k
q_s
n
$q_k = -k\frac{\partial \phi}{\partial n}$
q_s
$k\frac{\partial \phi}{\partial n} = -q_s$
$M = 0 \quad S = -q_s$

Figure 12.17. Specified heat flux applied on the boundary.

For other cases where M and/or S is not zero, $\mathbf{b}_B^{(e)}$ can be given by

$$\begin{aligned}\mathbf{b}_B^{(e)} &= -\int_{\Gamma_2} \mathbf{N}^T \left(D_x \frac{\partial \phi^h}{\partial x} \cos\theta + D_y \frac{\partial \phi^h}{\partial y} \sin\theta \right) \mathrm{d}\Gamma \\ &= -\int_{\Gamma_2} \mathbf{N}^T (M\phi_b + S)\, \mathrm{d}\Gamma\end{aligned} \tag{12.150}$$

where ϕ_b can be expressed using shape function as follows:

$$\phi_b^{(e)} = \mathbf{N\Phi} \tag{12.151}$$

Substituting Eq. (12.151) back into Eq. (12.150) leads to

$$\begin{aligned}\mathbf{b}_B^{(e)} &= -\int_{\Gamma_2} \mathbf{N}^T (-M\mathbf{N\Phi}^{(e)} + S)\, \mathrm{d}\Gamma \\ &= \underbrace{\left(\int_{\Gamma_2} \mathbf{N}^T M \mathbf{N}\, \mathrm{d}\Gamma \right)}_{\mathbf{k}_M^{(e)}} \mathbf{\Phi}^{(e)} - \underbrace{\int_{\Gamma_2} \mathbf{N}^T S\, \mathrm{d}\Gamma}_{\mathbf{f}_S^{(e)}}\end{aligned} \tag{12.152}$$

or

$$\mathbf{b}_B^{(e)} = \mathbf{k}_M^{(e)} \mathbf{\Phi}^{(e)} - \mathbf{f}_S^{(e)} \tag{12.153}$$

in which

$$\mathbf{k}_M^{(e)} = \int_{\Gamma_2} \mathbf{N}^T M \mathbf{N}\, \mathrm{d}\Gamma \tag{12.154}$$

is the contribution by the natural boundaries to the ‘stiffness’ matrix, and

$$\mathbf{f}_S^{(e)} = \int_{\Gamma_2} \mathbf{N}^T S\, \mathrm{d}\Gamma \tag{12.155}$$

is the force vector contribution from the natural boundaries.

Let us now calculate the force vector $\mathbf{f}_S^{(e)}$ for a rectangular element shown in Figure 7.8. Assuming that S is specified over side 1–2,

$$\mathbf{f}_S^{(e)} = \int_{\Gamma_{1-2}} S\mathbf{N}^T\, \mathrm{d}\Gamma = \int_{-1}^{1} S \begin{Bmatrix} N_1 \\ N_2 \\ N_3 \\ N_4 \end{Bmatrix} a\, \mathrm{d}\xi \tag{12.156}$$

where the shape functions are given by Eq. (7.51) in the natural coordinate system. Note, however, that $N_3 = N_4 = 0$ along edge 1–2. Substituting the non-zero shape functions into

Eq. (12.156), we obtain

$$\mathbf{f}_S^{(e)} = \int_{-1}^{1} \frac{Sa}{2} \begin{Bmatrix} (1-\xi) \\ (1+\xi) \\ 0 \\ 0 \end{Bmatrix} \mathrm{d}\xi = Sa \begin{Bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{Bmatrix} \tag{12.157}$$

The above equation implies that the quantity of $(2aS)$ is shared equally between nodes 1 and 2 on the edge. This even distribution among the nodes on the edge is valid for all the elements with a linear shape function. Therefore, if the natural boundary is on the other three edges of the rectangular element, the force vector can be simply written as follows:

$$\mathbf{f}_{S,2-3}^{(e)} = Sb \begin{Bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{Bmatrix}, \quad \mathbf{f}_{S,3-4}^{(e)} = Sa \begin{Bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{Bmatrix}, \quad \mathbf{f}_{S,1-4}^{(e)} = Sb \begin{Bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{Bmatrix} \tag{12.158}$$

Note that if S is specified on more than one side of an element, the values for $\{f_S^{(e)}\}$ for the appropriate sides are added together.

The same principle of equal sharing can be applied to the linear triangular element shown in Figure 12.13. The expression for the force vectors on the three edges can be simply written as

$$\mathbf{f}_{S,1-2}^{(e)} = \frac{SL_{12}}{2} \begin{Bmatrix} 1 \\ 1 \\ 0 \end{Bmatrix}, \quad \mathbf{f}_{S,2-3}^{(e)} = \frac{SL_{23}}{2} \begin{Bmatrix} 0 \\ 1 \\ 1 \end{Bmatrix}, \quad \mathbf{f}_{S,1-3}^{(e)} = \frac{SL_{13}}{2} \begin{Bmatrix} 1 \\ 0 \\ 1 \end{Bmatrix} \tag{12.159}$$

The quantities L_{12}, L_{23} and L_{13} are the lengths of the respective edges of the triangular element.

To derive $\mathbf{k}_M^{(e)}$ for the rectangular element shown in Figure 7.8 using Eq. (12.154), we have

$$\mathbf{k}_M^{(e)} = \int_{\Gamma_2} M \begin{bmatrix} N_1^2 & N_1N_2 & N_1N_3 & N_1N_4 \\ N_1N_2 & N_2^2 & N_2N_3 & N_2N_4 \\ N_1N_3 & N_2N_3 & N_3^2 & N_3N_4 \\ N_1N_4 & N_2N_4 & N_3N_4 & N_4^2 \end{bmatrix} \mathrm{d}\Gamma \tag{12.160}$$

Note that the line integration is performed round the edge of the rectangular element. If we assume that M is specified over edge 1–2, then $N_3 = N_4 = 0$, and the above equation becomes

$$\mathbf{k}_{M,1-2}^{(e)} = aM \int_{-1}^{1} \begin{bmatrix} N_1^2 & N_1N_2 & 0 & 0 \\ N_2N_1 & N_2^2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \mathrm{d}\xi \tag{12.161}$$

Evaluation of the individual coefficients after noting $\eta = -1$ gives

$$\begin{aligned}
\int_{-1}^{1} N_1^2 \,\mathrm{d}\xi &= \int_{-1}^{1} \frac{(1-\xi)^2}{4} \mathrm{d}\xi = \frac{2}{3} \\
\int_{-1}^{1} N_1 N_2 \,\mathrm{d}\xi &= \int_{-1}^{1} \frac{(1-\xi)(1+\xi)}{4} \mathrm{d}\xi = \frac{2}{6} \\
\int_{-1}^{1} N_2^2 \,\mathrm{d}\xi &= \int_{-1}^{1} \frac{(1+\xi)^2}{4} \mathrm{d}\xi = \frac{2}{3}
\end{aligned} \tag{12.162}$$

Equation (12.161) thus becomes

$$\mathbf{k}_{M,1-2}^{(e)} = \frac{2aM}{6} \begin{bmatrix} 2 & 1 & 0 & 0 \\ 1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} = (2aM) \begin{bmatrix} \frac{2}{6} & \frac{1}{6} & 0 & 0 \\ \frac{1}{6} & \frac{2}{6} & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{12.163}$$

It is observed that the amount of $(2aM)$ is shared by four components k_{11}, k_{12}, k_{21} and k_{22} in ratios of $\frac{2}{6}$, $\frac{1}{6}$, $\frac{1}{6}$ and $\frac{2}{6}$. This sharing principle can be used to directly obtain the matrices $\mathbf{k}_M^{(e)}$ for a situation where M is specified on the other three edges. They are

$$\begin{gathered}
\mathbf{k}_{M,2-3}^{(e)} = \frac{M2b}{6} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{k}_{M,3-4}^{(e)} = \frac{M2a}{6} \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 1 \\ 0 & 0 & 1 & 2 \end{bmatrix}, \\
\mathbf{k}_{M,1-4}^{(e)} = \frac{M2b}{6} \begin{bmatrix} 2 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 2 \end{bmatrix}
\end{gathered} \tag{12.164}$$

This sharing principle can also be applied to linear triangular elements, since the shape functions are also linear. We therefore obtain

$$\begin{gathered}
\mathbf{k}_{M,i-j}^{(e)} = \frac{ML_{ij}}{6} \begin{bmatrix} 2 & 1 & 0 \\ 1 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \mathbf{k}_{M,j-k}^{(e)} = \frac{ML_{jk}}{6} \begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 1 & 2 \end{bmatrix}, \\
\mathbf{k}_{M,i-k}^{(e)} = \frac{ML_{ik}}{6} \begin{bmatrix} 2 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 2 \end{bmatrix}
\end{gathered} \tag{12.165}$$

12.4.5 Point Heat Source or Sink

If there is a heat source or sink in the domain of the problem, it is best recommended that in the modelling, a node is placed at the point where the source or sink is located, so that the source or sink can be directly added into the force vector, as shown in Figure 12.18. If, for some reason, this cannot be done, then we have to distribute the source or sink into the nodes of the element, in which the source or sink is located. To do this, we have to go back to Eq. (12.122), which is once again rewritten:

$$\mathbf{f}_Q^{(e)} = \int_{A_e} Q\mathbf{N}^T \, \mathrm{d}A \tag{12.166}$$

Consider a point source or sink in a triangular element, shown in Figure 12.19. The source or sink can be mathematically expressed using the delta function

$$Q = Q^*\delta(x - X_0)\delta(y - Y_0) \tag{12.167}$$

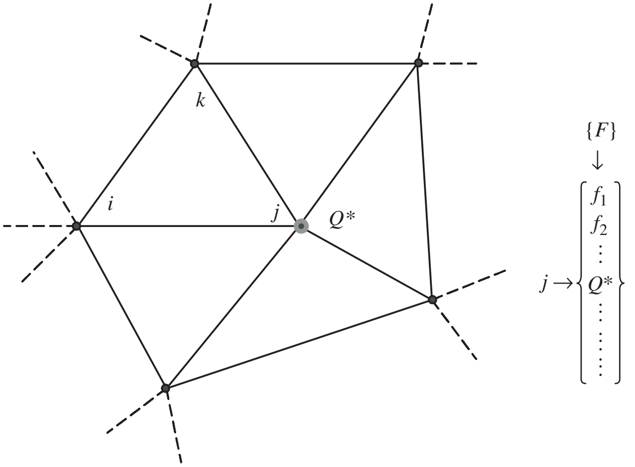


Figure 12.18. A heat source or sink at a node of the FE model.

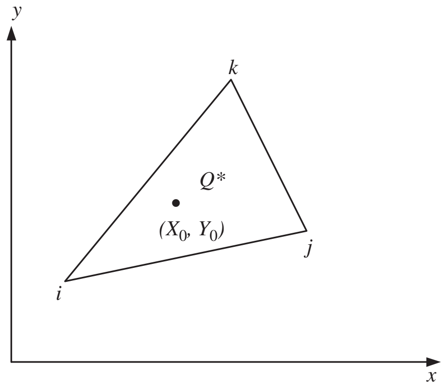


Figure 12.19. A heat source or sink in a triangular element.

where Q^* represents the strength of the source or sink, and (X_0, Y_0) is the location of the source or sink. Substitute Eq. (12.167) into Eq. (12.166), and we have

$$\mathbf{f}_Q^{(e)} = Q^* \int_{A_e} \begin{Bmatrix} N_i \\ N_j \\ N_k \end{Bmatrix} \delta(x - X_0)\delta(y - Y_0)\, \mathrm{d}x\, \mathrm{d}y \tag{12.168}$$

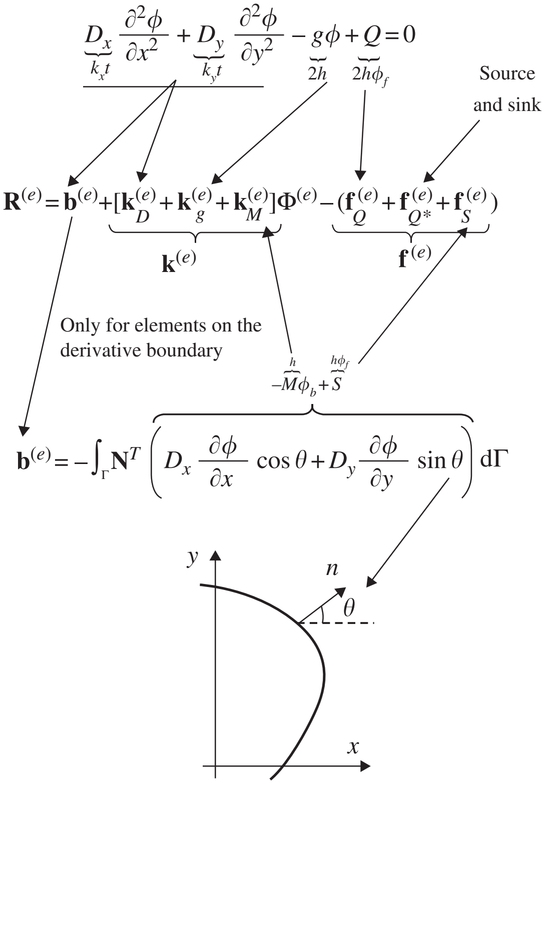
which becomes

$$\mathbf{f}_Q^{(e)} = Q^* \begin{Bmatrix} N_i(X_0, Y_0) \\ N_j(X_0, Y_0) \\ N_k(X_0, Y_0) \end{Bmatrix} \tag{12.169}$$

This implies that the source or sink is shared by the nodes of the elements in the ratios of shape functions evaluated at the location of the source or sink. This sharing principle can be applied to any type of elements, and also other types of physical problems. For example, for a concentrated force applied in the middle of a 2D element.

12.5 SUMMARY

Finite element formulation for field problems governed by the general form of a *Helmholtz equation* can be summarized as follows.



$\underbrace{D_x}_{k_x t}\dfrac{\partial^2\phi}{\partial x^2} + \underbrace{D_y}_{k_y t}\dfrac{\partial^2\phi}{\partial y^2} - \underbrace{g}_{2h}\phi + \underbrace{Q}_{2h\phi_f} = 0$
Source and sink
$\mathbf{R}^{(e)} = \mathbf{b}^{(e)} + \underbrace{[\mathbf{k}_D^{(e)} + \mathbf{k}_g^{(e)} + \mathbf{k}_M^{(e)}]}_{\mathbf{k}^{(e)}}\Phi^{(e)} - \underbrace{(\mathbf{f}_Q^{(e)} + \mathbf{f}_{Q^*}^{(e)} + \mathbf{f}_S^{(e)})}_{\mathbf{f}^{(e)}}$
Only for elements on the derivative boundary
$-\overset{h}{\overbrace{M}}\phi_b + \overset{h\phi_f}{\overbrace{S}}$
$\mathbf{b}^{(e)} = -\int_\Gamma \mathbf{N}^T \left(D_x \dfrac{\partial\phi}{\partial x}\cos\theta + D_y \dfrac{\partial\phi}{\partial y}\sin\theta \right) \mathrm{d}\Gamma$
y
n
θ
x

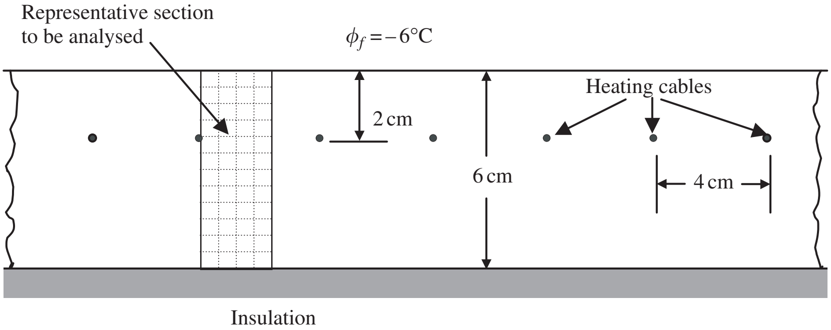


Figure 12.20. Cross-section of a road with heating cables.

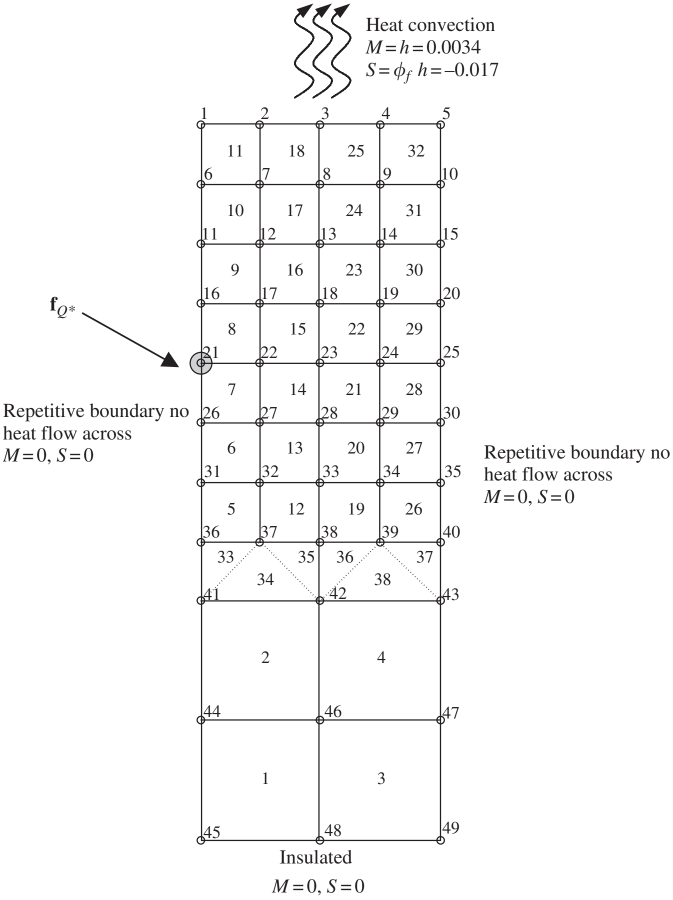


Figure 12.21. 2D finite element mesh with boundary conditions.

12.6 CASE STUDY: TEMPERATURE DISTRIBUTION OF HEATED ROAD SURFACE

Figure 12.20 shows a cross-section of a road with heating cables to prevent the surface of the road from freezing. The cables are 4 cm apart and 2 cm below the surface of the road. The slab rests on a thick layer of insulation, and the heat loss from the bottom can be neglected. The conductivity coefficients are $k_x = k_y = 0.018\,\mathrm{W/cm°C}$ and the surface convection coefficient is $h = 0.0034\,\mathrm{W/cm°C}$. The latter corresponds to about a 30–35 km/hr of wind velocity. The surface temperature of the road is to be determined when the cable produces 0.080 W/cm of heat, and the air temperature is -6°C.

12.6.1 Modelling

Since the road is very long in the horizontal direction, a representative section shown in Figure 12.20 can be used to model the whole problem domain. The FE mesh is shown in Figure 12.21, together with boundary conditions specified.

The mesh shown in Figure 12.21 demonstrates mesh transition from an area consisting of a sparse mesh to an area of denser mesh. The analyst has chosen to mesh it this way, since the temperature distribution at the bottom of the model is not that critical. Hence, computational time is reduced as a result. The transition is done with the use of triangular elements in between larger rectangular elements and smaller rectangular elements. Note that all the elements used are linear elements, and hence the mixture of elements here is still compatible.

12.6.2 ABAQUS Input File

Part of the ABAQUS input file is shown here:

```
*HEADING
Calculation of 2D heat transfer
**
*NODE
1, 0., 6.
2, 0.5, 6.
3, 1., 6.
4, 1.5, 6.
5, 2., 6.

⋮

46, 1., 1.
47, 2., 1.
48, 1., 0.
49, 2., 0.
**
```

Nodal cards

Node ID, *x*-coordinate, *y*-coordinate.

```
*ELEMENT, TYPE=DC2D4, ELSET=QUAD
1,45,48,46,44
2,48,49,47,46

⋮

31,14,15,10,9
32,9,10,5,4
**
```

Element (connectivity) cards

Element type here is DC2D4 which represents a 2D, four-node quadrilateral, heat transfer element. (Element ID, node 1, node 2, node 3, node 4)

```
*ELEMENT, TYPE=DC2D3, ELSET=TRI
33,41,37,36
34,41,42,37

⋮

37,39,43,40
38,42,43,39
**
**
```

Element (connectivity) cards

Element type here is DC2D3 which represents a 2D, three nodal triangular, heat transfer element. (Element ID, node 1, node 2, node 3)

```
*SOLID SECTION, ELSET=QUAD, MATERIAL=ROAD
1.,
**
*SOLID SECTION, ELSET=TRI, MATERIAL=ROAD
1.,
**
**
```

Property cards

Define properties to the elements of sets “QUAD” and “TRI”. They will have the material properties defined under “ROAD”.

```
*MATERIAL, NAME=ROAD
**
*CONDUCTIVITY, TYPE=ISO
0.018,
**
**
```

Material cards

Define material properties under the name “ROAD”. Thermal conductivity coefficient is being defined. TYPE=ISO represents isotropic properties.

```
*STEP
**
*HEAT TRANSFER, STEADY STATE
0.1, 1.
**
```

Control cards

Indicate the steady state, heat transfer analysis.

```
*ELSET, ELSET=SURFACE
11, 18, 25, 32
*ELSET, ELSET=LEFT_QUAD, GENERATE
1, 2, 1
5, 11, 1
*ELSET, ELSET=RIGHT_QUAD, GENERATE
3, 4, 1
26, 32, 1
*ELSET, ELSET=BASE
1, 3
*ELSET, ELSET=LEFT_TRI
33
```

Element sets

Group elements into sets to be referenced when defining boundary conditions.

```
*ELSET, ELSET=RIGHT_TRI
37
*NSET, NSET=SOURCE
21
**
*FILM, OP=NEW
SURFACE, F3, -6., 0.0034
**
** insulated edges
**
*DFLUX, OP=NEW
LEFT_QUAD, S4, 0.
RIGHT_QUAD, S2, 0.
BASE, S1, 0.
LEFT_TRI, S3, 0.
RIGHT_TRI, S2, 0.
```

BC cards

The keyword *FILM is used to define the heat convection properties. In the data line, the first input refers to the element set SURFACE, the second refers to the surface or edge the convection is occurring, the fourth is the sink temperature, and lastly, the convection coefficient.

*DFLUX is for specifying distributed heat flux. In this case, the left, right and bottom edges are all insulated (= 0).

```
**
** heat source
**
*CFLUX, OP=NEW
SOURCE, 11, 0.08
**
```

Load cards

The load here is a concentrated heat flux or source defined by *CFLUX and applied on node 21 or node set SOURCE. Note that in this case the DOF for the temperature is defined by the number 11.

```
*NODE PRINT, FREQ=1
NT,
*NODE FILE, FREQ=1
NT,
**
*END STEP
```

Output control cards

Define the requested output. In this case, NT is the nodal temperature.

The information provided in the above input file is used by the software in similar ways as discussed in case studies in previous chapters.

12.6.3 Result and Discussion

Running the above problem in ABAQUS, the nodal temperatures can be calculated. Figure 12.22 shows a fringe plot of the distribution of the temperatures in the model. It can be seen clearly how the temperature varies from a maximum at the heat source (the heating cables) to other parts of the road cross-section.

In the analysis, the temperatures at all the nodes are calculated. For this problem, we would be interested in only the temperature of the road surface. Table 12.1 shows the nodal temperature on the surface of the road. It can be seen here how the presence of the heating cables under the road is able to keep the road surface at a temperature above the freezing point of water (0°C), as shown in Table 12.1. This would prevent the build up of ice on the road surface during winter, which makes it safer for drivers on the road. The usefulness of the finite element method is demonstrated here, as there are actually many parameters involved when it comes to designing such a system. For example, how deep should the cables

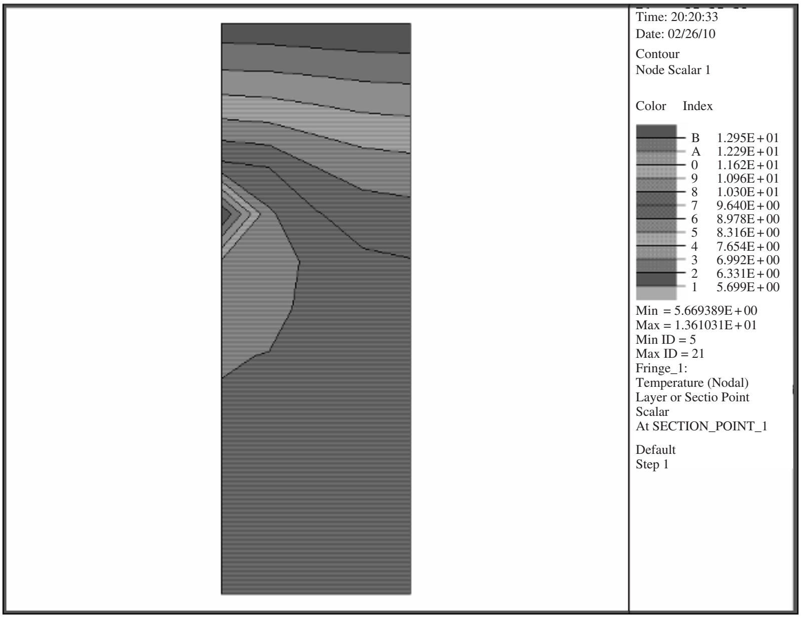


Figure 12.22. Temperature distribution of the cross-section of a road.

Table 12.1. Nodal temperatures of road surface

Node	Temperature (°C)
1	5.861
2	5.832
3	5.764
4	5.697
5	5.669

be buried underground; what should be the distance between cables; what amount of heat generated by the heating cables is sufficient for the purpose, and so on. The finite element method used here can effectively aid the engineer in deciding upon all these parameters.

12.7 REVIEW QUESTIONS

1. (a) A fin with a length of L has a uniform cross-sectional area A and thermal conductivity k, as shown in Figure 12.23. A linearly distributed heat supply is applied on the fin. The temperature at the left end is fixed at T_0, and the heat flux at the right

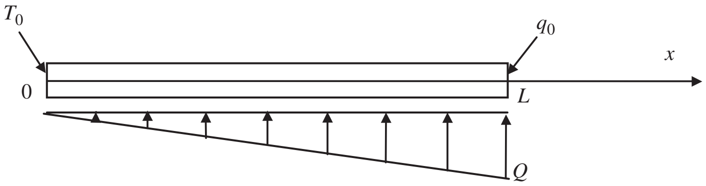


Figure 12.23. 1D fin with linearly distributed heat supply.

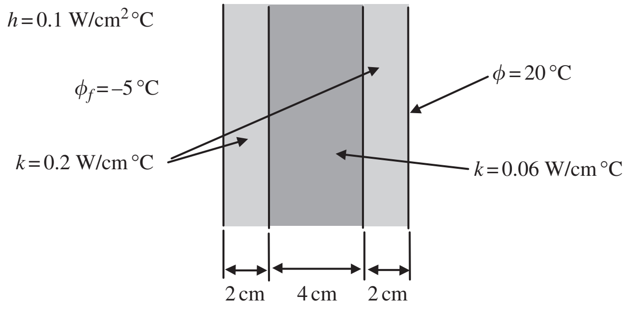


Figure 12.24. Sandwiched composite wall.

end is q_0. The governing equation for the fin is given by

$$Ak\frac{\partial^2\phi}{\partial x^2}+Q=0, \quad 0\le x\le L$$

Develop the finite element equation for a two-node element.

(b) If $L=8\,\text{m}, A=1\,\text{m}^2, k=5\,\text{J/°C ms}, Q=100\,\text{J/sm}, T_0=0$ and $q_0=15\,\text{J/m}^2\text{s}$, determine the temperatures at the nodes by using two linear elements.

2. Figure 12.24 shows a sandwiched composite wall. Convection heat loss occurs on the left surface, and the temperature on the right surface is constant. Considering a unit area, and with the parameters given in Figure 12.24, use three linear elements (one for each layer) to

   (a) determine the temperature distribution through the composite wall, and
   (b) calculate the flux on the right surface of the wall.

3. Consider a soldering situation where the tip raises the temperature at point A of a copper wire to 100°C (Figure 12.25). The wire is 15 cm long and 0.02 cm in diameter. The temperature at both ends is 20°C. The thermal conductivity k of the copper wire is 26 J/°C ms. Assume the circumferential surface of the wire is adiabatic. Using three linear elements of equal length,

   (a) determine the heat flux into the wire at point A, and
   (b) determine the heat flux at both ends of the wire.

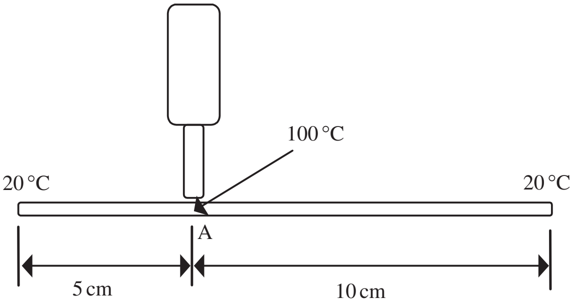


Figure 12.25. Soldering of copper wire.

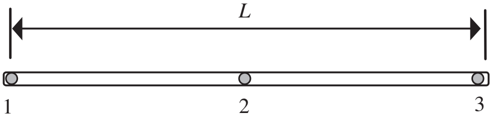


Figure 12.26. Quadratic 1D element.

(c) explain whether three elements are really needed for this problem, and
(d) repeat (a) and (b), for giving a heat flux of 4×10^{-3} J/s instead of the temperature.

4. Consider a quadratic heat-conduction line element with three equally spaced nodes as shown in Figure 12.26:

(a) Using the quadratic element, determine the heat conduction matrix.
(b) Using one linear element for the left portion, and one quadratic element for the right portion of the wire shown in Figure 12.25, determine the heat flux at point A and both ends of the wire.
(c) Comment on the results by comparing with the results of Question 3.

13

USING ABAQUS©

13.1 INTRODUCTION

Realistic finite element problems might consist of up to hundreds of thousands, and even several millions, of elements and nodes, and therefore they are usually solved in practice using commercially available software packages. There is currently a large number of commercial software packages available for solving a wide range of problems: solid and structural mechanics, heat and mass transfer, fluid mechanics, acoustics and multi-physics, which might be static, dynamic, linear and nonlinear. Most of these software packages use the finite element method, or are used in combination with other numerical methods. All these software packages are developed based on similar methodology described in this book, with many detailed and fine tuned techniques and schemes. Table 13.1 lists some of the commercially available software packages that use the FEM, Finite Volume Method

Table 13.1. Commercially available software packages

Software packages	Methods used	Application problems
ABAQUS	**FEM (implicit, explicit)**	**Structural analysis, acoustics, thermal analysis, etc.**
I-deas	FEM (implicit)	Structural analysis, acoustics, thermal analysis, etc.
LS-DYNA	FEM (explicit)	Structural dynamics, computational fluid dynamics, Fluid-structural interaction, etc.
Sysnoise	FEM/BEM	Acoustics (frequency domain)
NASTRAN	FEM (implicit)	Structural analysis, acoustics, thermal analysis, etc.
MARC	FEM (implicit)	Structural analysis, acoustics, thermal analysis, etc.
MSC-DYTRAN	FEM + FVM (explicit)	Structural dynamics, computational fluid dynamics, Fluid-structural interaction, etc.
ANSYS	FEM (implicit)	Structural analysis, acoustics, thermal analysis, multi-physics, etc.
ADINA DIANA	FEM (implicit)	Structural analysis, computational fluid dynamics, Fluid-structural interaction, etc.

(FVM) and Boundary Element Method (BEM). This chapter introduces the use of ABAQUS, developed by Hibbitt, Karlsson & Sorenson, Inc., due to its strong capabilities in dealing with nonlinear problems.

With the development of convenient user interfaces, most finite element software can be used as a ‘black box’, and is used by many users without proper knowledge of the FEM. The authors have seen many cases of misusing FEM packages, which results in simulations that can be described as garbage in and garbage out. The danger is that the garbage output is often covered by beautiful pictures and animations that can lead to harmful decisions in designing an engineering system.

Understanding of the materials covered in the previous chapters should shed some light into the black box, leading to the proper use of most software packages. This chapter uses ABAQUS as an example to describe proper use of commercial software packages from the user’s point of view. Chapters 1 to 12 have highlighted the various concepts in the finite element method, and the case studies actually relate these concepts to examples of using the ABAQUS software. Throughout the case studies in this book, the analyses are carried out using ABAQUS/Standard finite element software (version 6.1). There are other modules in the ABAQUS finite element package, including ABAQUS/Explicit, ABAQUS/CAE and ABAQUS/Viewer. ABAQUS/Explicit is mainly used for explicit dynamic analysis. ABAQUS/CAE is an interactive preprocessor that can be used to create finite element models and the associated input file for ABAQUS. ABAQUS/Viewer is a menu-driven interactive post-processor for viewing the results obtained from ABAQUS/Standard and ABAQUS/Explicit. In this chapter, however, the focus will be on the writing of the ABAQUS/Standard input file, and ABAQUS/Standard will from now on just be called ABAQUS. This book cannot and will not try in any way to replace the extensive and excellent manuals provided together with the ABAQUS software. This chapter will just serve as a general guide especially suited for beginner users to have a quick start in using ABAQUS, without going through the details of thick manuals. It is hoping that readers, after reading this chapter, will have an even better understanding of the finite element concepts being implemented in the finite element software. It should be noted that though only ABAQUS is introduced in this book, the use of other software packages is actually similar in many ways, except for the detailed format of inputs and outputs.

13.2 BASIC BUILDING BLOCK: KEYWORDS AND DATA LINES

The first step to writing an ABAQUS input file is to know the way in which data is included in the input file. In ABAQUS, the data definitions are expressed in what are termed ‘option blocks’ or ‘groups of cards’. Basically, it is thus called because the user has the option to choose particular data blocks that are relevant for the model to be defined. Each option block can be considered to be a basic building block that builds up the entire input file. The option block is introduced by a *keyword* line, and if the option block requires *data lines*, these will follow directly below the keyword line. The general layout of a particular option block is shown in Case 1 with the definition of beam elements as an example.

Case 1

```
*ELEMENT, TYPE=B23, ELSET=BEAM }  Keyword line  begin with *
1, 1, 2                        }
2, 2, 3                        }  Data lines
```

Keyword lines begin with an asterisk, *, followed by the name of the block. In this case, to define elements in the element block identified with the keyword ELEMENT. Subsequent information on the keyword line provides additional parameters associated with the block being used. In this case, it is necessary to tell ABAQUS what type of elements are being used (B23 – 2 node, 1D Euler–Bernoulli beam element), and the elements are grouped up into a particular set with the arbitrary given name ‘BEAM’. This grouping of entities into sets is a very convenient tool, which the analyst will use very often. It enables the analyst to make references to the set when defining certain option blocks.

The data lines basically provide data, if required, that is associated with the option block used. In the example in Case 1, element identification (i.d.) and the nodes that make up the element are the necessary data required. Note that the information provided in the data lines would vary with some of the parameters defined in the keyword line. For example, if the element type being used is a 2D plane stress element, then the data lines would require different data, as shown in Case 2.

Case 2

```
*ELEMENT, TYPE=CPS4, ELSET=PLSTRESS
1, 1, 3, 4, 2 }
2, 3, 5, 6, 4 }  Element i.d.,  1st node, 2nd node, 3rd node, 4th node
```

Note that, in this case, the element type CPS4 which represents four-nodal, 2D solid elements is being used, and correspondingly, the data lines must include the element i.d. (as before), and four nodes that make up each element instead of two for the case of the beam element previously.

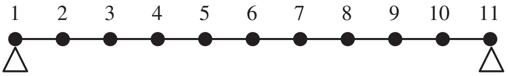
13.3 USING SETS

In the last section, it was seen in Case 1 that elements can be grouped into a *set* for future reference by other option blocks. A set can be a grouping of nodes or a grouping of elements. The analyst will usually provide a name for the set that contains between 1–80 characters. For example, in Case 1, ‘BEAM’ is the name of the set containing elements 1 and 2; and in Case 2, ‘PLSTRESS’ is the name of the set containing elements 1 and 2. In both examples, the sets are defined together with the definition of the elements themselves in the element block. However, sets can also be defined as a separate block on their own. In Case 3, the pinned support of the 1D beam is to be defined. Nodes 1 and 11 (provided in the data line for the ‘NSET’ block) are first grouped in the node set called ‘SUPPORT’. Then using the

'BOUNDARY' option block, the node set, 'SUPPORT' is referenced to constrain the DOFs, 1 and 2 (x translation and y translation) to zero. In other words, rather than having four data lines for the two nodes 1 and 11 (each node having to constrain two DOFs), we now have only two lines with the reference to the node set 'SUPPORT'. Another thing to note in this example is the use of comment lines. Comment lines exist in ABAQUS too, just as in most programming languages. Comment lines begin with two asterisks, `**`, and whatever follows in that line after that will not be read by ABAQUS as input information defining the model.

Case 3

```
** NODE SET FOR SUPPORTS
*NSET, NSET=SUPPORT
1, 11
**
** PIN SUPPORT
*BOUNDARY
SUPPORT, 1,,0.
SUPPORT, 2,,0.
```



Node set name or node i.d.,1 st DOF to be constrained, last DOF to be constrained, magnitude of prescribed condition

This is, of course, a very simple case, and the reduction in the number of data lines from four to two may not seem very significant. However, imagine if the model is a huge 3D model, and one whole surface containing about 100 nodes is to be prescribed a boundary condition. If the nodes on this surface were not grouped into node sets, then the user would end up with 100 × (*No. of DOFs to be constrained*) data lines just to prescribe a boundary condition. A more efficient way, of course, is to group the nodes in this particular surface in a node set, and then like Case 3, write down the data lines referencing the node set to be constrained. In this way, the number of data lines for the 'BOUNDARY' block would be equal to the number of DOFs to be constrained. Similar use of sets can be applied to elements as well. One common use of element sets (ELSET) is the referencing of element properties to the elements in a particular set (see the example in Section 13.5.3). Sets are thus the basic referencing tool in ABAQUS.

13.4 ABAQUS INPUT SYNTAX RULES

The previous section introduced the way in which data are organized in the ABAQUS input file. Like most programming languages, there are certain rules which the entries into the input file must follow. A violation of these rules would generally result in syntax error when ABAQUS is run, and most of the time the analysis will not be carried out. So far, it has been shown that ABAQUS generally has three types of entries in the input file: the keyword lines, the data lines and the comment lines. Comment lines generally do not need many rules, except that it must begin with two asterisks (`**`) in columns 1 and 2. This section therefore describes the rules that apply to all keywords and data lines.

13.4.1 Keyword Lines

The following rules apply when entering a keyword line:

- The first non-blank character of each keyword line must be an asterisk (*).
- The keyword must be followed by a comma (,), if any further parameters are to be included in the line.
- Parameters are separated by commas.
- Blanks on a keyword line are ignored.
- A line can include no more than 256 characters including blanks.
- Keywords and parameters are not case sensitive.
- Parameter values are not usually case-sensitive. The only exceptions to this rule are those imposed externally on ABAQUS, such as file names on case-sensitive operating systems.
- If a parameter has a value, the equal sign (=) is used. The value can be an integer, a floating point number, or a character string, depending on the context. For example,

```
*EL FILE, POS=INTEG, FREQ=1
```

- Continuation of a keyword line is sometimes necessary, especially when there is a large number of parameters. If the last character of a keyword line is a comma, the next line is treated as a continuation line. For example, the example stated in the previous rule can also be written as

```
*EL FILE, POS=INTEG,
FREQ=1
```

13.4.2 Data Lines

The data lines must immediately follow a keyword line if they are required. The following rules apply when entering a data line:

- A data line can include no more than 256 characters including blanks.
- All data items are separated by commas. An empty data field is specified by omitting data between commas. ABAQUS will use values of zeros for any required numeric data that are omitted, unless there is a default value allocated. If a data line contains only a single data item, the data item should be followed by a comma.
- A line must contain only the number of items specified.
- Empty data fields at the end of a line can be ignored.
- Floating point numbers can occupy a maximum of 20 spaces including the sign, decimal point, and any exponential notation.
- Floating point numbers can be given with or without an exponent. Any exponent, if input, must be preceded by E or D and an optional (−) or (+) to indicate the sign of the exponent.
- Integer data items can occupy a maximum of 10 digits.

- Character strings can be up to 80 characters long and are not case-sensitive.
- Continuation lines are allowed in specific instances, such as when defining elements with a large number of nodes. If allowed, such lines are indicated by a comma as the last character of the preceding line.

13.4.3 Labels

Examples of labels are set names, surface names and material names, and they are case-sensitive, unlike the other entries in the keyword line. Labels can be up to 80 characters long. All spaces within a label are ignored unless the label is enclosed in quotation marks, in which case all spaces within the label are maintained. A label that is not enclosed within quotation marks may not include a period (.), and should not contain characters such as commas and equal signs. If a label is defined using quotation marks, the quotation marks are stored as part of the label. Any subsequent reference or use of the label should also include the quotation marks. Labels cannot begin and end with a double underscore (e.g. __ALU__). This label format is reserved for use internally within ABAQUS.

13.5 DEFINING A FINITE ELEMENT MODEL IN ABAQUS

Though the use of a preprocessor like ABAQUS/CAE or PATRAN can be helpful in creating the finite element model and generating the input file for complex models, the analyst will still often find that the preprocessor cannot automatically generate many functions available in ABAQUS, which are required in the input file. In a way, today's preprocessors mainly cater for the most common problems. A specific analysis will often require more than just the usual analysis steps, and this is when an analyst will find that knowing the basic concepts of writing an input file will enable him or her to either write a whole input file for the specific problem, or to modify the existing input file that is generated by the preprocessor.

An ABAQUS input file is an ASCII data file which can be created or edited using any text editor. The input file contains two main sets of data: *model data* and *history data*. The model data consists of data defining the finite element model. This part of the input file defines the elements, nodes, element properties, material properties and any other data that specifies the model itself. Looking at the input files provided for the case studies included in previous chapters, it should be noted that all the data provided before the '*STEP' line is considered as the model data. The history data on the other hand defines what happens to the finite element model. It tells ABAQUS the events the model has gone through, the loadings the model has, the type of response that should be sought for, and so on. In ABAQUS, the history data is made up of one or more *steps*. Each step defines the analysis procedures by providing the required parameters. It is possible and in fact quite common to have multiple steps to define a whole analysis procedure. For example, to obtain a steady-state dynamic response due to a harmonic excitation at a given frequency by modal analysis, one must first obtain the eigenvalues and eigenvectors. This can be defined in a step defined by *FREQUENCY, which calls for the eigenvector extraction analysis procedure. Following that, another step defined by *STEADY STATE DYNAMICS is necessary that calls for the modal analysis procedure to solve for the response under a harmonic excitation. As such, the history data can be said to make up of series of steps, which in a way tells the history of the analysis procedure. This section will describe in more detail how a basic model can be defined in ABAQUS. Input files defining most problems have the same basic structure:

1. An input file must begin with the *HEADING option block, which is used to define a title for the analysis. Any number of data lines can be used to give the title.
2. After the heading lines, the input file would usually consist of the model data, which is the node definition, element definition, material definition, initial conditions and so on.
3. Finally, the input file would consist of the history data, in which is defined the analysis type, loading, output requests, and so on. Usually, the *STEP option is the dividing point in the input file between the model and history data. Everything appearing before the *STEP option will be considered as model data, and everything after will be considered as the history data.

The following outlines some of the options and data that can be included in the model and history data. This book will not elaborate on all the available options, and if required, the user is recommended to refer to the software's user manual. Elaboration will be done on some of the necessary options for a basic finite element model later.

13.5.1 Model Data

Some of the data that must be included in the model data are as follows:

- Geometry of the model: The geometry of the model is described by its elements and nodes.
- Material definitions, which are usually associated with parts of the geometry.

Other optional data in the model data section are:

- Parts and an assembly: the geometry can be divided into parts, which are positioned relative to one another in an assembly.
- Initial conditions: non-zero initial conditions such as initial stresses, temperatures or velocities can be specified.
- Boundary conditions: zero-valued boundary conditions (including symmetry conditions) can be imposed on the model.
- Constraints: linear constraint equations or multi-point constraints can be defined.
- Contact interactions: contact conditions between surfaces or parts can be defined.
- Amplitude definitions: amplitude curves for which the loads or boundary conditions are to follow can be defined.
- Output control: options for controlling model definition output to the data file can be included.
- Environment properties: environment properties such as the attributes of a fluid surrounding the model can be defined.
- User subroutines: user-defined subroutines, which allow the user to customize ABAQUS, can be defined.
- Analysis continuation: it is possible to write restart data or to use the results from a previous analysis and continue the analysis with new model or history data.

13.5.2 History Data

As mentioned, in the history data, the entries are divided into steps. Each step begins with the *STEP option and ends with the *END STEP option. There are generally two kinds of step that can be defined in ABAQUS – the general response analysis steps (can be linear or nonlinear); and the linear perturbation steps. A general analysis step is one in which the effects of any nonlinearities present in the model can be included. The starting condition for each general analysis step is the ending condition from the last general analysis step. The response of each general analysis step contributes to the overall history of the response of the model. A linear perturbation analysis step, on the other hand, is used to calculate the linear perturbation response from the *base state*. The base state is the present state of the model at the end of the last general analysis response. For the perturbation step, the response does not contribute to the history of the overall response, and hence can be called for at any time in between general steps. For cases where the general step or the linear perturbation step is the first step, then the initial conditions defined will define the starting condition or the base state, respectively. The following is a list of the analysis types that uses linear perturbation procedures:

- *BUCKLE (Eigenvalue buckling prediction)
- *FREQUENCY (Natural frequency extraction)
- *MODAL DYNAMIC (Transient modal dynamic analysis)
- *STEADY STATE DYNAMICS (Modal steady-state dynamic analysis)
- *STEADY STATE DYNAMICS, DIRECT (Direct steady-state analysis)
- *RESPONSE SPECTRUM (Response spectrum analysis)
- *RANDOM RESPONSE (Random response analysis)

Except for the above analysis types and for the *STATIC (where both general and perturbation steps can be used), all other analysis types are general analysis steps.

Some of the data that must be included in the history data or within a step are:

- Analysis type: an option to define the analysis procedure type which ABAQUS will perform. This must appear immediately after the *STEP option.

Other optional data include:

- Loading: some form of external loading can be defined. Loadings can be in the form of concentrated loads, distributed loads, thermal loads, and so on. Loadings can also be prescribed as a function of time following the amplitude curve defined in the model data. If an amplitude curve is not defined, ABAQUS will assume that the loading varies linearly over the step (ramp loading), or that the loading is applied instantaneously at the beginning of the step (step loading).
- Boundary conditions: zero-valued or non-zero boundary conditions can be added, modified or removed. Note that if defined in the model data, only zero-valued and symmetrical boundary conditions can be included.
- Output control: controls the requested output from the analysis. Output variables depend upon the type of analysis and the type of elements used.

- Auxiliary controls: options are provided to allow the user to overwrite the solution controls that are built into ABAQUS.
- Element and surface removal/reactivation: portions of the model can be removed or reactivated from step to step.

13.5.3 Example of Cantilever Beam Problem

One of the best ways of learning about and understanding the ABAQUS input file is to follow an example. We shall illustrate with a simple example of modelling a cantilever beam subjected to a downward force, as shown in Figure 13.1. The above problem can be modelled using 1D beam elements, and the finite element mesh will be as follows:

As mentioned, the first thing to include in the input file would be the *HEADING option. The data line after the *HEADING keyword line briefly describes the problem.

```
*HEADING
Model of a cantilever beam with a downward force
```

Next will be writing the model data. First, the nodes of the problem must be defined, since elements must be made up of nodes, and both nodes and elements make up the geometry of the problem.

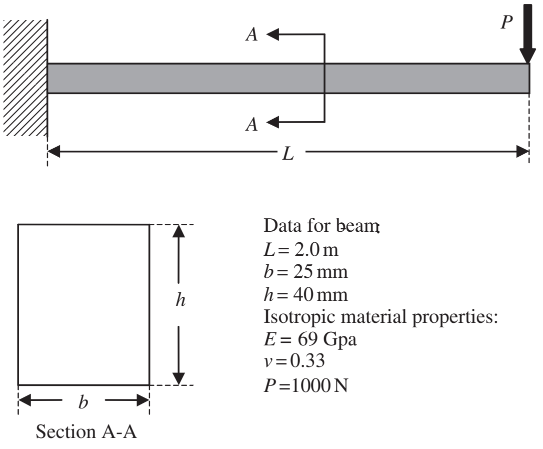


Figure 13.1. Cantilever beam under downward force.

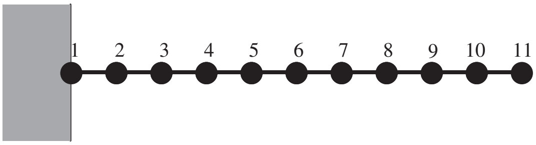


Figure 13.2. Cantilever beam meshed with 1D, two-nodal, beam elements.

```
*NODE
1, 0.
11, 2.0
*NGEN
1, 11
```

Using the *NODE option, the nodes at the end are first defined. We then use the option *NGEN to generate evenly distributed nodes between the first and last nodes. *NGEN is one of the several mesh generation capabilities provided by ABAQUS. We could also define all 11 nodes individually by specifying their coordinates, but using *NGEN would be more efficient for large problems. So we have now defined 11 nodes uniformly along the length of the beam. Next, the elements would be defined:

```
*ELEMENT, TYPE=B23
1, 1, 2
*ELGEN, ELSET=RECT_BEAM
1, 10
```

Here, the *ELEMENT option is used to define the first element that consists of nodes 1 and 2. The TYPE parameter is included to specify what type of element is being defined. In this case, B23 refers to a 1D beam elment in a plane with cubic interpolation. Users can refer to the ABAQUS manual for the element library to check the codes for other element types. Similar to the definition of nodes, *ELGEN is used to generate 1–10 elements subsequently. The elements are then grouped into a node set called RECT_BEAM. This will make the referencing of element properties much easier later. So we have now defined 11 nodes and 10 elements as shown in Figure 13.2.

The next step will be to define the element properties:

```
*BEAM SECTION, ELSET=RECT_BEAM, SECTION=RECT, MATERIAL=ALU
0.025, 0.040
0., 0., -1.0
```

In the *BEAM SECTION keyword line, the element set RECT_BEAM defined earlier is now referenced, meaning that the elements grouped under RECT_BEAM will all have the properties defined in this option block. We also provide the information that the beam has a rectangular (RECT) cross-section. There are other cross-sections available in ABAQUS, such as circular cross-sections (CIRC), trapezoidal cross-sections (TRAPEZOID), closed thin-walled sections (BOX, HEX and PIPE) and open thin-walled sections (I-section and L-section). ABAQUS also provides for a ‘general’ cross-section by specifying geometrical quantitites necessary to define the section. The material associated with the elements is also defined as ALU, where the properties will be defined later. It is a good time to note that, unlike most programming languages, the ABAQUS input file need not follow a top-down approach when ABAQUS is assessing the file. For example, the material ALU is already referenced at this point under the *BEAM SECTION option block, though its material properties are actually defined further down the input file. There will not be any error stating that the material ALU is invalid regardless of where the material is defined unless it is not defined at all throughout the input file. This is true for all other entries into the input file. Let us now look at the data lines. The first data line in the *BEAM SECTION basically defines the dimensions of the cross-section (0.025 × 0.04 m). Note that the dimensions here are converted to metres to be consistent with the coordinates of the nodes. The second data line basically defines the direction cosines indicating the local beam axis. What is given is the default values, and this line can actually be omitted in this case.

The next entry in the model data would be the material properties definition:

```
*MATERIAL, NAME=ALU
*ELASTIC, TYPE=ISOTROPIC
69.E9, 0.33
```

The material for our example is aluminium, and we name it ALU for short. All the properties option block will follow after the *MATERIAL option block, which does not require any data lines by itself. The *ELASTIC option defines elastic properties, and TYPE=ISOTROPIC defines the material as an isotropic material, i.e. the material properties are the same in all directions. The data line for the *ELASTIC option includes the values for the Young's modulus and Poisson's ratio. Depending upon the type of analysis carried out, or the type of material being defined, other properties may need to be defined. For example, if a dynamic analysis is required, then the *DENSITY option would also need to be included; or when the material exhibits viscoelastic behaviour, then the *VISCOELASTIC option would be required.

At this point, we have almost completed describing the model in the model data. What is left are the boundary conditions. Note that the boundary conditions can also be defined in the history data. What can be defined in the model data is only the zero valued conditions.

```
*BOUNDARY
1, 1, 6, 0.
```

There is actually more than one way of defining a *BOUNDARY. What is shown is the most direct way. The first entry into the data line is the node i.d. or the name of the node set, if one is defined. In this case, since it is only one single node, there is no need for a node set. But many times, a problem might involve a whole set of nodes where the same boundary conditions are applied. It would thus be more convenient to group these nodes into a set and referenced in the data line. The second entry is the first DOF of the node to be constrained, while the third entry is the last DOF to be constrained. In ABAQUS, for displacement DOFs, the number 1, 2 and 3 would represent the translational displacements in the x, y and z-directions, respectively, while the numbers 4, 5 and 6 would represent the rotations about the x, y and z-axes, respectively. Of course, depending on the type of element used and the type of analysis carried out, there may be other DOFs represented by other numbers (refer to the ABAQUS manual). For example, if a piezoelastic analysis is carried out using piezoelastic elements, there is an additional DOF (other than the displacement

DOFs) number 9 representing the electric potential of the node. In this case, all the DOFs from 1 to 6 will be constrained to zero (fourth entry in the data line). Strictly speaking, the DOFs available for the 1D planar, beam element in ABAQUS are only 1, 2 and 6 since the others are considered out of plane displacements. Since we constrained all six DOFs, ABAQUS will just give a warning during analysis that the constraints on DOFs 3, 4 and 5 will be ignored, since they do not exist in this context.

There are numerous parameters that can actually be included in the keyword line of the *BOUNDARY option if they are required (refer to the ABAQUS manuals for details). For example, the boundary condition can be made to follow an amplitude curve by including AMPLITUDE = *Name of amplitude curve definition* in the keyword line. ABAQUS also provides for certain standard types of zero-valued boundary conditions. For example, the above boundary condition can also be written as

```
*BOUNDARY
1, ENCASTRE
```

The word ENCASTRE used in the data line represents a fully built-in condition, which also means that DOFs 1–6 are constrained to zero. Other standard boundary conditions are listed in Table 13.2. After defining the boundary condition, we have now completed what is required for the model data of the input file.

We now need to define the history data. As mentioned, the history data would begin with the *STEP option. In this example, we would be required to obtain the displacements of the beam as well as the stress along the beam due to the downward force. One step would be sufficient here and the loading will be static:

```
*STEP, PERTURBATION
*STATIC
```

The perturbation parameter following the *STEP option basically tells ABAQUS that only a linear response should be considered. The *STATIC option specifies that a static analysis

Table 13.2. Standard boundary condition types in ABAQUS

Boundary condition type	Description
XSYMM	Symmetry about a plane x = constant (DOFs 1, 5, 6 = 0)
YSYMM	Symmetry about a plane y = constant (DOFs 2, 4, 6 = 0)
ZSYMM	Symmetry about a plane z = constant (DOFs 3, 4, 5 = 0)
ENCASTRE	Fully clamped (DOFs 1 to 6 = 0)
PINNED	Pinned joint (DOFs 1, 2, 3 = 0)
XASYMM	Anti-symmetry about a plane x = constant (DOFs 2, 3, 4 = 0)
YASYMM	Anti-symmetry about a plane y = constant (DOFs 1, 3, 5 = 0)
ZASYMM	Anti-symmetry about a plane z = constant (DOFs 1, 2, 6 = 0)

is required. The next thing to include will be the loading conditions:

```
*CLOAD
11, 2, -1000.
```

ABAQUS offers many types of loading. *CLOAD represents concentrated loading, which is the case for our problem. Other types of loading include *DLOAD for distributed loading, *DFLUX for distributed thermal flux in thermal-stress analysis, and *CECHARGE for concentrated electric charge for nodes of piezoelectric elements. The first entry in the data line is the node i.d. or the name of the node set if defined, the second is the DOF the load is applied to, and the third is the value of the load. In our case, since the force is acting downward, it is acting on DOF 2 with a negative sign following the convention in ABAQUS. Most loadings can also follow an amplitude curve varying with time by including AMPLITUDE = *Name of amplitude curve definition* in the keyword line. This is especially so if transient, dynamic analysis is required.

For this problem, there is not much more data to include in the history data other than the output requests. The user can request the type of outputs he or she wants by indicating as follows:

```
*NODE PRINT, FREQ=1
U,
*NODE FILE, FREQ=1
U,
*ELEMENT PRINT, FREQ=1
S,
E
*ELEMENT FILE, FREQ=1
S,
E
```

From what we learned from the finite element method, we can actually deduce that certain output variables are direct nodal variables like displacements, while others like stress and strain are actually determined as a distribution in the element using the shape functions. In ABAQUS, this difference is categorized into nodal output variables and element output variables. *NODE PRINT outputs the results of the required nodal variables in an ASCII text file (.dat file), while the *NODE FILE ouputs the results in a binary format (.fil file). The binary format can be read by post-processors in which the results can be displayed. Similarly, *ELEMENT PRINT outputs element variables in ASCII format, while *ELEMENT FILE outputs them in binary format. A list of the different output variables can be obtained in the ABAQUS manuals. For our case, U in the data lines for *NODE PRINT and *NODE FILE will output all the components of the nodal displacements. S and E represent all the components of stress and strain, respectively. So if the analysis is run, there will be altogether three tables: one showing the nodal displacements, one showing the stresses in the elements, and the last one showing the strains in the elements. The last thing to do now is end the step by including *ENDSTEP. If multiple steps are present, this would separate the different steps in the history data.

Running the analysis

So the whole input file defining the problem of the cantilever beam is shown below:

```
*HEADING
Model of a cantilever beam with a downward force
**
*NODE
1, 0.
11, 2.0
*NGEN
1, 11
**
*ELEMENT, TYPE=B21
1, 1, 2
*ELGEN, ELSET=RECT_BEAM
1, 10
**
*BEAM SECTION, ELSET=RECT_BEAM, SECTION=RECT,MATERIAL=ALU
0.025, 0.040
0., 0., -1.0
**
*MATERIAL, NAME=ALU
*ELASTIC, TYPE=ISOTROPIC
69.E9, 0.33
**
*BOUNDARY
1, 1, 6, 0.
**
*STEP, PERTURBATION
*STATIC
**
*CLOAD
11, 2, -1000.
**
*NODE PRINT, FREQ=1
U,
*NODE FILE, FREQ=1
U,
*ELEMENT PRINT, FREQ=1
S,
E
*ELEMENT FILE, FREQ=1
S,
E
**
*ENDSTEP
```

ABAQUS input files end with the extension .inp. So if we call this file beam.inp, we can run this example in ABAQUS using the following command at the command prompt (note that, to-date, ABAQUS is usually run on a unix platform):

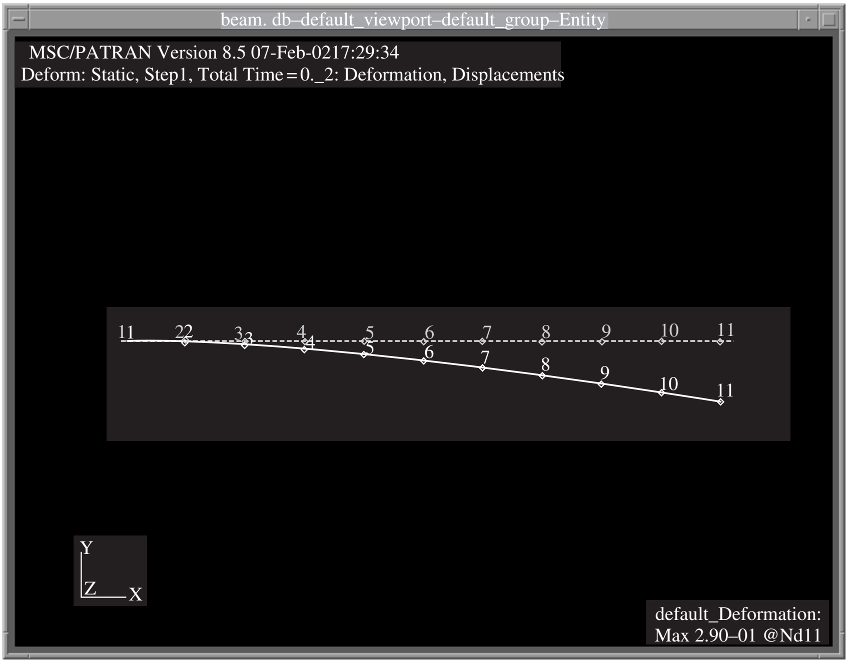
```
abaqus job = beam
```

Users can check the full syntax of the ABAQUS execution command in the manuals.

Results

After executing the analysis, there could be several results files generated. In ASCII text format would be the beam.dat file. ABAQUS ouputs its results in ASCII format in the file ending with the extension .dat. As mentioned, the binary format would be in the file with the .fil extension, and is generally used as input for post-processors. The .dat file can of course be viewed by any text editor, and it will show lots of numbers associated with the input processing, the analysis steps, and lastly, the requested outputs (*NODE PRINT and *EL PRINT). These output data can of course be used for plotting graphs or as inputs to other programming codes, depending on the user. Many users would view the results using post-processors like ABAQUS/Post, ABAQUS/Viewer, PATRAN, and so on. The choice is entirely up to the preference of the user, and of course, the availability of these post-processors. In this book, the results are viewed using PATRAN, and the results are shown below.

Figure 13.3 shows the deformation plot of the cantilever beam as obtained in PATRAN. This plot shows how the cantilever beam deforms under the applied loads. The actual



beam. db–default_viewport–default_group–Entity

MSC/PATRAN Version 8.5 07-Feb-0217:29:34
Deform: Static, Step1, Total Time = 0._2: Deformation, Displacements

default_Deformation:
Max 2.90–01 @Nd11

Figure 13.3. Deformation plot from PATRAN.

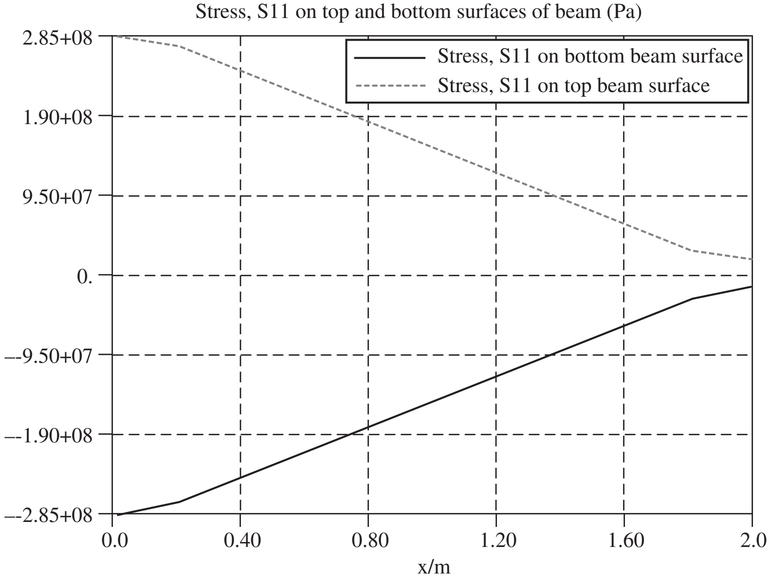


Figure 13.4. XY-plot of stress, σ_{xx} along the beam.

displacements of the nodes can also be included in the deformation plot, but if there are many nodes, it makes viewing them on-screen difficult. Figure 13.4 shows an XY-plot obtained in PATRAN of the stress, σ_{xx}, on the top and bottom surface of the beam. The plot clearly shows a tensile stress on the top and a compressive stress at the bottom. XY-plots of strain and displacements can be similarly obtained in PATRAN.

13.6 GENERAL PROCEDURES

How to write the ABAQUS input file of a simple problem of a cantilever beam has been shown in the previous section. This chapter will not be sufficient to go through the many keywords that are available in ABAQUS, and therefore the focus will not concentrate on that. Readers and users should consult the manuals for more information regarding the keywords. This section thus aims to provide a general guide not just to using ABAQUS, but generally to most finite element software.

From the previous example, it can be seen that certain information must be provided for the software to carry out the analysis. This information is required to solve for the finite element problem, and it has been highlighted throughout this book that the information is mainly used to formulate the necessary matrices. Of course, there are certain parameters that govern the algorithm and the way in which the equations are solved in the program as well. So in this sense, there should not be much difference between different software other than the format in which the information is supplied and the way in which results

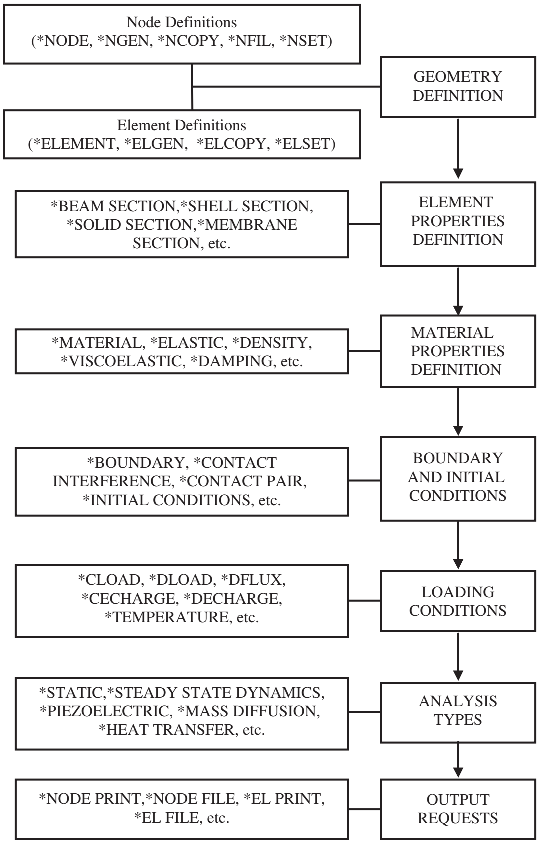

Node Definitions
(*NODE, *NGEN, *NCOPY, *NFIL, *NSET)
GEOMETRY DEFINITION
Element Definitions
(*ELEMENT, *ELGEN, *ELCOPY, *ELSET)
*BEAM SECTION,*SHELL SECTION, *SOLID SECTION,*MEMBRANE SECTION, etc.
ELEMENT PROPERTIES DEFINITION
*MATERIAL, *ELASTIC, *DENSITY, *VISCOELASTIC, *DAMPING, etc.
MATERIAL PROPERTIES DEFINITION
*BOUNDARY, *CONTACT INTERFERENCE, *CONTACT PAIR, *INITIAL CONDITIONS, etc.
BOUNDARY AND INITIAL CONDITIONS
*CLOAD, *DLOAD, *DFLUX, *CECHARGE, *DECHARGE, *TEMPERATURE, etc.
LOADING CONDITIONS
*STATIC,*STEADY STATE DYNAMICS, *PIEZOELECTRIC, *MASS DIFFUSION, *HEAT TRANSFER, etc.
ANALYSIS TYPES
*NODE PRINT,*NODE FILE, *EL PRINT, *EL FILE, etc.
OUTPUT REQUESTS

Figure 13.5. General information required by finite element software.

are presented. Figure 13.5 is a summary of the general information that finite element software requires to solve most problems. The keywords provided on the left are some of the keywords used in ABAQUS to provide that particular information. To summarize, we would first need to define the geometry by defining the nodes and the elements. Remember that in the finite element method, the whole domain is discretized to small elements. This is generally called meshing. Next, we would need to provide some properties for the elements used. For example, using 1D beam elements would require one to provide the type of cross-section and the cross-section dimension; or when using 2D plate elements it would require the thickness of the plate to be provided, and so on. After that we would need to define the properties of the material or materials being used and associated with the elements. We would then need to provide information regarding the boundary and initial conditions the model is under. This is necessary for the solver to evaluate the equations. Similarly for the loading conditions, which must also be provided unless there is no load on the model like in many analyses involving natural frequencies extraction. After all this, the model is more or less defined. The next step would be to tell the software what type of problem or analysis this is. Is the problem a static analysis, or a transient dynamic analysis, or a heat transfer analysis? The software would require this information and the user must provide it with the analysis type. Finally, the user can also tell the software what are the results he or she is seeking. For example, for most applied mechanics problems, the displacements are the true nodal variables that the solver will compute. The software, however, can also compute the stress and strain from interpolation of these nodal displacements automatically, and this can be done by specifying them in the input file.

REFERENCES

ABAQUS user's manual, volumes I, II and III, version 6.1: Hibbitt, Karlsson & Sorensen, Inc., 2000.

ABAQUS keywords manual, version 6.1: Hibbitt, Karlsson & Sorensen, Inc., 2000.

ABAQUS theory manual, version 6.1: Hibbitt, Karlsson & Sorenson, Inc., 2000.

Argyris, J. H., Fried, I. and Scharpf, D. W., The TET 20 and TEA 8 elements for the matrix displacement method. *Aero. J.* Vol. 72, 618–625, 1968.

Barsoum, R. S., On the use of isoparametric finite elements in linear fracture mechanics, *International Journal for Numerical Methods in Engineering*, Vol. 10, 25–38, 1976.

Barsoum, R. S., Triangular quarter point elements as elastic and perfectly elastic crack tip elements, *International Journal for Numerical Methods in Engineering*, Vol. 11, 85–98, 1977.

Bathe, K. J., *Finite Element Procedures*, Prentice Hall, Englewood Cliffs, 1996.

Belytschko, T., Liu, K. L. and Moran, B., *Nonlinear Finite Elements for Continua and Structures* John Wiley & Sons, Ltd, 2000.

Bettess, P., *Infinite Elements*, Penshaw Press, 1992.

Cheung, Y. K., *Finite Strip Method in Structured Analysis*, Pergamon Press, 1976.

Clough, R.W. and Penzien, J., *Dynamics of Structures*, McGraw-Hill, New York, 1975.

Cook, R. D., *Finite Element Modeling for Stress Analysis*, John Wiley & Sons, Inc., 1995.

Cook, R. D., *Concepts and Applications of Finite Element Analysis*, 2nd edition, John Wiley & Sons, 1981.

Crandall, S. H., *Engineering Analysis: A Survey of Numerical Procedures*, McGraw-Hill, New York, 1956.

Crocker, M. J., editor, *Handbook of Acoustics*, Chapter 1. John Wiley & Sons, 1998.

Daily, J. W. and Harleman, D. R. F., *Fluid Dynamics*, Addison-Wesley, Reading, Mass., 1966.

Eisenberg, M. A. and Malvern, L. E., On finite element integration in natural coordinates. *Int. J. Num. Meth. Eng*. Vol. 7, 574–575, 1973.

Finlayson, B. A., *The Method of Weighted Residuals and Variational Principles*, Academic Press, New York, 1972.

Finlayson, B. A. and Scriven, L. E., The method of weighted residuals – a review, *Applied Mechanics Review*, Vol. 19, 735–748, 1966.

Fung, Y. C., *Foundations of Solid Mechanics*, Prentice-Hall, Englewood Cliffs, 1965.

Henshell, R. D. and Shaw, K. G., Crack tip elements are unnecessary, *International Journal for Numerical Methods in Engineering*, Vol. 9, 495–509, 1975.

Hilber, H. M., Hughes, T. J. R., and Taylor, R. L., Collocation, dissipation and 'overshoot' for time integration schemes in structural dynamics, *Earthquake Engineering and Structural Dynamics*, Vol. 6, 99–117, 1978.

Hughes, J. R. T., *The Finite Element Method*. Prentice-Hall International, Inc., 1987.

Kardestuncer, H. editor-in-chief, *Finite Element Handbook*, McGraw-Hill, 1987.

Kausel, E. and Roësset, J. M., Semianalytic hyperelment for layered strata, *Journal of the Engineering Mechanics Division*, Vol. 103(4), 569, 1977.

Liu, G. R., *Mesh Free Methods: Moving Beyond the Finite Element Method*, CRC Press, Boca Raton, 2002.

Liu G. R., A combined finite element/strip element method for analyzing elastic wave scattering by cracks and inclusions in laminates. *Computational Mechanics*. Vol. 28, 76–81, 2002.

Liu, G. R. and Achenbach, J. D., A strip element method for stress analysis of anisotropic linearly elastic solids. *ASME J. Appl. Mech.*, Vol. 61, 270–277, 1994.

Liu, G. R. and Achenbach, J. D., A strip element method to analyze wave scattering by cracks in anisotropic laminated plates. *ASME J. Appl. Mech.*, Vol. 62, 607–613,1995.

Liu, G. R., Achenbach, J. D., Kim, J. O. and Li, Z. L., A combined finite element method/boundary element method for V(z) curves of anisotropic-layer/substrate configurations. *Journal of the Acoustical Society of America*, Vol. 92(5), 2734–2740, 1992.

Liu, G. R. and Quek, S. S. Jerry, A non-reflecting boundary for analyzing wave propagation using the finite element method. *Finite Elements in Analysis and Design*. (in press.)

Liu, G. R. and Quek, S. S. Jerry, A finite element study of the stress and strain fields of InAs quantum dots embedded in GaAs. *Semiconductor Science and Technology*, Vol. 17, 630–643, 2002.

Liu, G. R. and Xi, Z. C. *Elastic Waves in Anisotropic Laminates*, CRC Press, 2001.

Liu, G. R., Xu, Y. G. and Wu, Z. P., Total solution for structural mechanics problems. *Computer Methods in Applied Mechanics and Engineering*. Vol. 191, 989–1012, 2001.

Mindlin, R. D., Influence of rotary inertia and shear on flexural motion of isotropic elastic plates, *Journal of Applied Mechanics*, Vol. 18, 31–38, 1951.

MSC/Dytran user's manual, version 4: The MacNeal-Schwndler Corporation, USA, 1997.

Murnaghan, F. D., *Finite Deformation of an Elastic Solid*, John Wiley & Sons, 1951.

NAFEMS, *A Finite Element Primer*, Dept. of Trade and Industry, 1986.

Newmark, N. M., A method of computation for structural dynamics, *ASCE Journal of Engineering Mechanics Division*, Vol. 85, 67-94, 1959.

Ottosen, N. S. and Pertersson, H., *Introduction to the Finite Element Method*, Prentice Hall, New York, 1992.

Peterson, L. A. and Londry K. J., Finite-element structural analysis: a new tool for bicycle frame design. *Bike Tech*, Vol. 5(2), 1986.

Petyt, M., *Introduction to Finite Element Vibration Analysis*, Cambridge University Press, Cambridge, 1990.

Quek, S. S., NUS industrial attachment report for session 1997/98: DSO National Laboratories.

Rao, S.S., *The Finite Element in Engineering*, 3rd edition, Butterworth-Heinemann, 1999.

Reddy, J. N., *Finite Element Method*, John Wiley & Sons Inc., New York, 1993.

Reddy, J. N., *Energy and Variational Methods In Engineering*, John Wiley, New York, 1984.

Reissner, E., The effect of transverse shear deformation on the bending of elastic plates, *Journal of Applied Mechanics*, Vol. 67, A67–A77, 1945.

Segerlind, L. J., *Applied Finite Element Analysis*, 2nd edition, John Wiley & Sons, Inc., 1984.

Tassoulas, J. L. and Kausel, E., Elements for the numerical analysis of wave motion in layered strata. *Int. J. Numer. Methods Eng.,* Vol. 19, 1005–1032, 1983.

Timoshenko, S., *Theory of Plates and Shells*, McGraw, London, 1940.

Timoshenko, S. P. and Goodier, J. N., *Theory of Elasticity*, 3rd edition, McGraw-Hall, New York, 1970.

Washizu, K. *et al. Finite Elements Handbook*, Vols 1 and 2. Baitukan, Japan (in Japanese), 1981.

Zienkiewicz, O. C., *The Finite Element Method*, 4th edition, McGraw-Hill, London, 1989.

Zienkiewicz, O. C. and Taylor, R. L., *The Finite Element Method*, 5th edition, Butterworth-Heinemann, 2000.

INDEX