A FINITE ELEMENT BASED PLATE/SHELL MACRO ELEMENT

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED
SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

ÖZGÜR KURÇ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF MASTER OF SCIENCE
IN
THE DEPARTMENT OF CIVIL ENGINEERING

JULY 1999

Approval of the Graduate School of Natural and Applied Sciences.

Prof. Dr. Tayfur ÖZTÜRK
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Fuat ERBATUR
Head of Department for

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assoc.Prof. Uğur POLAT
Supervisor

Examining Committee Members

Prof. Dr. S. Tanvir WASTI

Prof. Dr. Çetin YILMAZ

Prof. Dr. Mehmet UTKU

Prof. Dr. Turgut TOKDEMİR

Assoc. Prof. Uğur POLAT

ABSTRACT

A FINITE ELEMENT BASED PLATE/SHELL MACRO ELEMENT

Kurç, Özgür

M.S., Department of Civil Engineering

Supervisor: Assoc. Prof. Dr. Uğur Polat

July 1999, 128 Pages

In this study, the derivation and implementation of a finite element based plate/shell macro (super) element with stiffeners is carried out. The element is intended for the analysis of structural systems with planar components. An object-oriented finite element program NESONEL having this macro element in its element library is developed. The macro elements can be used in modeling any kind of planar structural component with the only restriction that its boundaries must be composed of straight line segments.

Quadrilateral elements are used to model the two-dimensional bending and stretching behavior. It is possible to have stiffeners in the macro element. The element is generated from a regular finite element model of the region by first condensing the inner domain nodes to boundary nodes on the

perimeter of the region by static condensation. Then, a prescribed displacement field is imposed on each boundary via some constraint equations thus eliminating the nodes on the boundaries. This way, the macro element definition is confined to a few coupling nodes on the boundary of the element domain.

Object-oriented paradigm is preferred in the analysis, design and the implementation of the computer program. The problem statement is considered in four different parts; mechanical problem, geometrical problem, mathematical problem and database problem. Each part is analyzed separately and the design and the implementation of the objects are made. The C++ language is used in the program.

In order to check the accuracy and the efficiency of the macro element, some test structures subjected to different kinds of loading are modeled using macro elements and analyzed. The results are compared with regular finite element solutions.

Key words: finite element method, macro element, constraint equations, static condensation , object-oriented programming,

ÖZET

SONLU ELEMANLAR TABANLI PLAK/KABUK MAKRO ELEMANI

Kurç, Özgür

Yüksek Lisans, İnşaat Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr. Uğur Polat

Temmuz 1999, 128 Sayfa

Bu çalışmada yapısal sistemlerin çözümlenmesi için bir makro elemanın formülasyonu yapılmıştır. Nesne yönelimli bakış açısı kullanılarak geliştirilen sonlu elemanlar programı "NESONEL" sayesinde bu elemanın pratik kullanımının sağlanması amaçlanmıştır. Geliştirilen makro elemanın doğrusal kenarlara sahip olan her türlü geometrideki plakların modellenmesinde kullanılması mümkündür.

İki boyutlu eleman davranışının modellenmesi için dörtgen elemanlar kullanılmıştır. Ayrıca makro elemana güçlendirici kirişlerin de eklenmesine olanak sağlanmıştır. Hesaplama sırasında ilk olarak iç düğüm noktalarının etkisi statik yoğunlaştırma yöntemiyle kenar düğüm noktalarına aktarılır. Daha sonra kenar düğüm noktalarına kısıt denklemleri uygulanarak

davranışlarının aktif düğüm noktalarında ifade edilmesi sağlanır. Kısıt denklemlerinin hesabında düz bir kenar boyunca gerçekleşeceği varsayılan yerdeğiştirme değişimi kullanılır. Bu sayede makro elemanın birkaç aktif düğüm noktasında tanımlanması sağlanmıştır.

Eldeki problemin incelenmesi, anlaşılması, ve bilgisayar programının geliştirilmesi sırasında nesne yönelimli bakış açısı kullanılmıştır. Problem dört ana başlık altında incelenmiştir, mekanik problem, geometrik problem, matematiksel problem ve veri tabanı problemi. Her bir başlık üzerinde teker teker çalışılmış ve gerekli nesnelerin tasarımı ve geliştirilmesi tamamlanmıştır. Program geliştirilirken C++ programlama dili kullanılmıştır.

Macro elemanın doğruluğunu ve kullanılabilirliğini test etmek amacıyla çeşitli yüklemeler altındaki değişik tipteki yapılar çözülmüş, sonlu eleman çözüm sonuçları makro eleman çözümleriyle karşılaştırılmıştır.

Anahtar Sözcükler: sonlu elemanlar yöntemi, makro eleman, kısıt denklemeleri, statik yoğunlaştırma , nesne yönelimli programlama.

To my family

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to Assoc.Prof.Dr.Uğur Polat for his great patience, encouragement, support and friendly approach towards me during this three year period that we worked together. I learned many things from him, not only about technical subjects but also about life and personal relationships.

I would like to thank my all of my friends at the Structural Mechanics Lab., especially Ömer Erbay, Uğurhan Akyüz, Hakan Argeşo, Onur Sonuvar, Erdem Canbay and Altuğ Erberik, for their friendship and support.

Special thanks to my dearest friends, Timur Koloğlu, Aydin Dirican, İbrahim Suat Gürsoy, Nazlı Seyithanoğlu and Burcu Akata. They were always near me and their support helped me a lot during my master degree studies. Also thanks to my English teacher, Başak Atasoy, for her hard work in connection with checking my thesis.

Finally thanks to my family, especially to my mother. They made the hardest thing very successfully, living with me during these last two years. Thank you for your patience and understanding.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURES

LIST OF SYMBOLS

ε	Strain
β	Rotations of the normal to the undeformed middle point of an edge
γ	Shear strain
κ	Curvature
μ	Poisson's ratio
θ	Nodal rotations
σ	Normal Stress
τ	Shear Stress
ξ	s/L
[B]	Strain-displacement relationship
[C]	Matrix of constraint constants
$[C_c]$	Corresponding coefficient matrix of D_c
$[C_r]$	Corresponding coefficient matrix of D_r
{D}	Displacement vector
$\{D_c\}$	Constrained displacement vector
$\{D_r\}$	Constrained displacement vector
$[D_k]$	Stress-strain relationship for plane stress case
E	Modulus of elasticity
h	Height of a beam
G	Shear modulus
[K]	Stiffness matrix
$[K_{con}]$	Condensed stiffness matrix
L	Length of an edge
N	Interpolation functions
NS	Serendipity shape functions
dN	Derivatives of interpolation functions
$\{R_{con}\}$	Condensed load vector
$\{R^p_r\}$	Vector for the transfer of constrained forces to the retained dof
s	Distance from the coupling node of an edge

t	Thickness of a plate
$\{u\}$	Nodal displacements
U	Total strain energy
U_b^e	Element strain energy

CHAPTER 1

INTRODUCTION

1.1 Statement of the Problem

Most of the civil engineering structures involve two-dimensional elements such as load bearing walls or floor slabs. The finite element method is the most accurate and widely used analysis tool for these elements. However, in order to get satisfactory results, the density of the finite element mesh must be high enough and the size of the elements must be small enough. Although computer technology develops incredibly fast, the analysis of large structures still requires very high capacities and takes a lot of computational time. Moreover, the preparation of the input data is a very tiring, time consuming and rather boring task for an engineer. This is also an error prone step of the analysis due to its nature.

The current approach to the analysis of such structures is to use simple finite elements to model the two-dimensional structural components. The floor systems are considered to be infinitely rigid for in-plane actions in an attempt to reduce the size of the model and thus the computational effort. Therefore, they are modeled as rigid diaphragms. The loads acting on them are transferred to the neighbouring beams or other edge supports by the yield line plastic analysis approach. The errors introduced to structural calculations by this approximate method of load transfer to edge supports

increase significantly as the floor geometry becomes more and more complex and irregular. In modeling the structural walls, one-dimensional simple bending elements are widely used. Unfortunately, this approach can not take into account the Poisson effect and shear deformations. Moreover, for arbitrarily shaped or pierced walls, it is not possible to pick-up the real behaviour by such a simplistic model. This approach also creates difficulties in modeling the connection of some other simple structural components to these planar elements.

The idea of developing a finite element based plate/shell macro element for the analysis of two-dimensional structural subsystems sounds very attractive. This approach is expected to decrease not only the required time for solution, but also the required computer capacity for a large structural systems. It is anticipated that this, in turn, helps the engineer work on more sophisticated and realistic mathematical models of structures.

Just a few decades ago, computer programs in the field of civil engineering were composed of not more than a few thousand statements on well-known problems. However, as computer technology develops, today not only do the user demands from a program increase, but also the solution of more complicated problems becomes possible. Hence, it is not unusual to find source codes consisting of hundreds of thousands or millions of statements. Therefore, the methods which facilitate the understanding of complex problems and are suitable for teamwork must be used in the software development.

Until very recently, the main problems of software development were the memory management and the speed of execution. Today, software has to satisfy some additional requirements besides. According to Meyer[9], the important software quality parameters can be summarized as follows:

- correctness,
- robustness,
- extensibility,
- reusability,
- compatibility,
- efficiency,
- functionality,
- ease of use, and
- ease of maintenance.

The object-oriented paradigm is highly attractive for this purpose since it can improve significantly the above four factors: Correctness, robustness, extensibility and reusability. It also has important contributions to compatibility, ease of use, maintenance and functionality. The similarities between the object-oriented analysis of a complex problem and the way of thinking for human beings is another advantage. However, in order to benefit from the facilities the object-oriented technology provides a detailed analysis of the complex problem and a good design of classes must be performed.

1.2 Literature Survey:

The theory and applications of the finite element method are covered in various textbooks. Those which are popular and also used frequently in this study are the ones by Cook[1], Bathe[5], Reddy[2], Zienkiewicz[3,4] and Owen[6]. The theory of plate and shell structures is reviewed, among others, in Boresi[7] and Timoshenko[8].

The derivation of a finite element based plate/shell macro element was previously done by H.Argeso[12] with the utilization of triangular elements. The accuracy and efficiency of these elements were tested on different problems by H.B.Bahat[13]. The same approach is also used and the work is extended in this study by the use of quadrilateral elements.

The imposition of constraint equations is very well explained by Cook[1]. Three different methods for the implementation of constraint equations are described: transformation method, Lagrange multipliers method and the penalty method. The application of transformation method to the development of a macro element was done by H.Argeso[12]. He applied the constraint equations node by node. By this way he avoided multiplication and addition of large matrices. In this study the constraint equations are imposed node by node again but this time simple row and column operations were used as described in Reddy[2]. While developing the static condensation algorithm the frontal solution algorithm first suggested by Bruce M.Irons[14] is modified. The computer program given by Owen[6] is very useful in understanding Iron's solution approach. Also, examining the works of Beer and Haas[15] and Thompson and Shimazaki[16] is very beneficial.

The finite element method developed very rapidly after the late 1950s. The first ten to fifteen years activity mainly focused on the element development. Also, there are many different methods used in the calculation of the element stiffnesses in finite element analysis. A brief classification of these methods is presented in Table 1.1

One of the main tasks of a finite element practitioner is to select the correct element. It is possible to find hundreds of different kinds of elements in the literature. The elements used for the analysis of two-dimensional elements for solid mechanics can be divided into three main groups; Plate bending, membrane and shell elements.

Table 1.1 Classification of finite element methods (Taken from Ref[17])

Finite Element Method	Variational Principle	Assumed functions inside the element	Along Interelement boundaries	Unknowns in Final Equations
Displacement	Minimum Potential Energy	Continuous Displacements	Displacement Compatibility	Nodal Displacements
Equilibrium	Minimum Complementary Energy	Continuous and Equilibrating Stresses	Equilibrium Boundary Tractions	a) Generalized Displacements b) Stress Parameters
Hybrid Stress Method	Modified Complementary Energy	Continuous and Equilibrating Stresses	Assumed Compatible Displacements	Nodal Displacements
Hybrid Displacement Method(1)	Modified Potential Energy	Continuous Displacements	Assumed Compatible Displacements	Nodal Displacements
Hybrid Displacement Method(2)	Modified Potential Energy	Continuous Displacements	Assumed Equilibrating Boundary Tractions	Nodal Displacements and Boundary Forces
Reissner's Principle	Reissner Method as modified by Hermann	Continuous Stress and Displacement Functions	Combinations of Boundary Tractions and Displacements	Combinations Displacements and Tractions
Generalized Displacement Method	Modified Potential Energy	Continuous Displacements	Lagrangian Multipliers	Nodal Displacements and Lagrangian multipliers
Generalized Equilibrium Method	Modified Complementary Energy	Continuous and Equilibrating Stresses	Lagrangian Multipliers	Nodal Displacements and Lagrangian Multipliers

In this study, twelve-degrees-of-freedom displacement based quadrilateral plate bending elements are preferred because of their better performance. Hence the review of the plate bending elements is limited to rectangular and quadrilateral elements having three degrees of freedom at each node. The Melosh[18] element, which was one of the first elements, uses the analogy between a plate and a system of crossing beams. He used complete cubic interpolation and two quadratic terms. Also compatible rectangular elements were presented by Deak and Pian[19], involving particular polynomial expressions on four subregions. Dhatt[20] used the Kirchhoff assumptions in his rectangular element. The discrete Kirchhoff assumptions were also studied by Fried[21]. The discrete Kirchhoff quadrilateral element used in this study was actually a generalisation of the Dhatt element.

One approach in defining a quadrilateral element is through the assemblage of several triangular elements. Some elements use more than two subregions and utilize static condensation. The Q15 element by Bathe[22] and Q19 element by Clough[23] are of this type. The isoparametric formulation is used in the QUAD4 element by McNeal[24]. He included the effect of transverse shear deformations. Its formulation does not seem to be simple and the results depend on an adjustable parameter. Later the elements proposed by Cook[25], Hughes[26], McNeal[24] and Robinson[27] also depend on thick plate theory including the effect of transverse shears. Batoz[28] used Kirchhoff formulation in the derivation triangular element together with Bathe and Ho. They developed a very efficient triangular element. Afterwards, Batoz[30] used the same principles while deriving discrete Kirchhoff Quadrilateral element. The A.Ibrahimbegovic and E.L.Wilson[31] thick plate element uses the Mindlin-Reissner theory. This element is built on a special hierarchical displacement field. In this study, this element is used as a thick plate element.

There were various attempts to develop membrane elements with drilling degrees of freedom. McNeal and Harder[33] presented unsatisfactory early efforts on this subject. After that Irons and Ahmad[34] believed that such attempts to develop membrane elements with drilling degrees of freedom was a waste of time. However, two independent works done by Allmann et al.[35] and Bergan and Felippa[37] achieved a level of success that was not attained previously. They created fresh interest to the problem and numerous works done by different researchers. These solutions based on free formulations depending on interpolation fields. They actually need a theoretical base for such elements.

Hughes and Brezzi[38] presented an approach based on variational formulation employing an independent rotation field. They extended the Reissner's approach. A.Ibrahimbegovic et al.[39] have also extended the applications of Hughes and Brezzi by combining Allmann type interpolation for the displacement field with an independently interpolated rotation field. This element was used as a membrane element.

The elements presented up to this point are full or selective integration elements. They are widely used because they are always stable. However, constructing element stiffness is computationally expensive. The elements using reduced integration, i.e. one quadrature point, are computationally very efficient. Unfortunately they permit spurious zero-energy modes that disturb the solution. This kind of element was first developed by Belytschko[41]. Engelmann and Whirley[42] introduced physical hourglass control. However, this element did not pass the Kirchhoff Patch Test. Then, Belytschko and Leviathan[43] developed a quadrilateral shell element with physical stabilization. In this element, they improved the transverse shear behaviour that helped the element to pass the Kirchhoff Test. However, these elements are all five degrees of freedom formulations. The one-point quadrilateral shell element with six degrees of freedom per node was

developed by Zhu and Zacharia[44]. This work is very helpful in understanding the concept significantly.

The software technology started after the innovation of structured programming in 1960s. 1970s were the times of structured and modular programming. As the software complexities had increased, new methodologies started to be searched. The data flow diagrams, structure charts and entity relationship diagrams were the most popular analysis and design methodologies during 1980s. 1990s are the days for the object-oriented paradigm that has many advantages compared with the previous procedural paradigms.

The implementation of object-oriented technology into finite element programming was previously studied by different researchers, Baugh and Rehak[45], Forde[46], Zimmermann[47,48]. Baugh and Rehak[45] define an object-oriented framework for finite element analysis. They based it on a geometric model having two classes, vertex and edge. Their system consists of three finite element classes, element, node and material. Forde[46] presented an object-oriented finite element program for linear elastic analysis with plane, isoparametric elements. He used five finite element classes. In addition to element, node and material classes, he used dispBC and forceBC classes for handling the boundary conditions. The C++ implementation of such kind of a program was done by Scholz[52]. Zimmermann and Dubois-Pélerin[47,48] developed a prototype program in order to examine if object-oriented programming was applicable to finite elements. They created two different kinds of classes. The Domain class deals with the management and the solution of the system. They used four finite element classes: element, node, material and load. Later, they presented C++ implementation of the program containing static and dynamic analysis features as well as non-linear material models. Mackie[49] tested the possibility of changing from procedural programming to object-

oriented programming. At the end of his study, he accomplished an enhancement of the program structure as well as reusability.

1.3 Object and Scope:

The main object of this study is to formulate and implement a finite element based plate/shell macro element which will possibly facilitate the analysis of structures having panel walls and floor systems. Since the macro elements are defined by a few global coupling nodes just like an ordinary finite element, the size of the element and hence the structural stiffness matrices will be rather small compared to the regular finite element model of the same structure. As a result, the required computer capacity and the solution time are expected to be of little concern even for large structural systems.

Three different groups of nodes are identified for a macro element: **inner domain** nodes, **boundary** nodes and the **coupling** nodes. First the effect of inner domain nodes is transferred to the boundary nodes by static condensation. In order to speed up this process, a modified frontal algorithm will be used. Then, the boundary node stiffnesses are transferred to the coupling nodes by using the assumed displacement fields along the edges. Constraint equations are used for this purpose.

The transformation method is used for imposing the constraint equations. Each boundary node is eliminated one by one to speed up the solution. At the end of this process, the macro element property matrices are obtained in terms of the coupling nodes. These matrices are used in the analysis of the whole structure. After the solution is obtained at the coupling nodes, the boundary node displacements are calculated by transferring the coupling node displacements using the constraint equations. Then, the inner domain nodal displacements are found by using the condensed stiffness matrix.

Four kinds of finite elements are used in this work. All elements are quadrilateral elements since it is known that they have more efficient and accurate behavior than the triangular elements. Triangular elements are very useful when the geometry is very complex but in civil engineering structures it is very rare to have such complex shapes.

As a thin plate element, J.L.Batoz[30] quadrilateral element that is based on Kirchhoff's thin plate theory is used. For thick plate element, the element of A.Ibrahimbegovic[31] is preferred. This element uses very similar interpolation functions as the ones in the membrane element. A.Ibrahimbegovic[39]'s membrane element is used for the in-plane actions since it has drilling degrees of freedom and the formulation is based on a mathematical theory developed by Hughes and Brezzi[38]. The last element is for the stiffeners. The joist beam element is derived from a beam element. First the neutral axis of the beam is transferred to the plate's neutral axis. Then the degrees of freedom of the beam element is written in terms of the plate degrees of freedom. In all of these calculations the constraint equations are used.

A computer program is developed for this purpose. The program is able to make finite element analysis and use macro elements. It has a complete object-oriented structure and its design is made to run with an event-oriented program. The C++ language is chosen for the core of the program since it provides object-oriented programming. However, for the mathematical operations like linear solution algorithms and matrix operations FORTRAN language is preferred due to its power in number crunching.

The theory of object-oriented paradigm is given in detail by Booch[9], Miller[10] and Martin[11]. The Booch notation was used throughout the thesis to describe class relationships.

The accuracy and the efficiency of the macro elements are tested on several test models. For this purpose, the finite element program SAP90[54] is preferred as a test program. The elements used in both programs are the same. Hence the correctness of our program is first checked by comparing the finite element results of both programs. Then the macro element solution results are compared with the results of the finite element solutions. During this process, the structures are first modeled by one macro element having different number of global nodes. Afterwards the same structure is modeled by different number of macro nodes. The displacement and nodal forces responses are compared.

CHAPTER 2

PLANAR SHELL MACRO ELEMENT

2.1 Introduction

There are some serious difficulties encountered by the engineers in the finite element analysis of structural systems containing some planar shell substructures such as panel walls with or without openings and floor panels. The speed of the solution and the capacity of the computer become major concerns for such structures even with todays sophisticated computers. Just because of these difficulties the engineers usually opt for some unduly simplified mathematical models.

The problems faced by the engineers carrying out the finite element analysis of a relatively complex civil engineering structure can be classified mainly in two groups. First of all, there is a memory and speed problem even with the powerful computers of today. This is caused by the excessive number of active degrees of freedom in the finite element model. The severity of the problem increases very rapidly with the increasing size and complexity of the structural system. This problem emerges immediately when the finite element model of the structural system contains plate/shell elements besides the simple 1-D bar elements. Secondly, there are serious difficulties in the finite element model-building and post-processing phases of the analysis which also increase with the increasing size and complexity of the structure.

Basic difficulties encountered in various stages of the finite element modeling of a given structure can be summarized as follows.

- Discretization and calculation of the nodal point coordinates to be used in defining the finite element mesh of the structure.
- Meshing the structure based on the nodal points assigned and defining the element connectivity in terms of these nodal points. This process becomes more difficult when the geometry is complicated and the regions are multiply connected such as walls or floors with openings. Another problem in this stage arises from the fact that the subregions of the structure meshed separately are required to have the same nodal points along their interfaces.
- Normally, the external excitations acting on the structure are defined with reference to the nodal points. This requires some preliminary calculations by the engineer before the analysis is carried out.

This is a rather boring and time consuming process demanding a lot of patience and energy on the part of the engineer. The process also creates a very conducive environment for making mistakes.

All of these problems decrease the efficiency and the speed of the structural analysis. The work in this study is focused on formulating and implementing a plate/shell macro element to be used in the analysis of structures with load-bearing walls and floor diaphragms. Such an element is believed to relieve substantially the difficulties stated above. Imagine the simple illustrative structure shown in Figure 2.1. The usual approach to analysis of such a structure is to first create a finite element mesh for the floor diaphragm and then to divide the edge beams properly to obtain displacement conformity, to some extend, along the plate/beam interface boundaries.

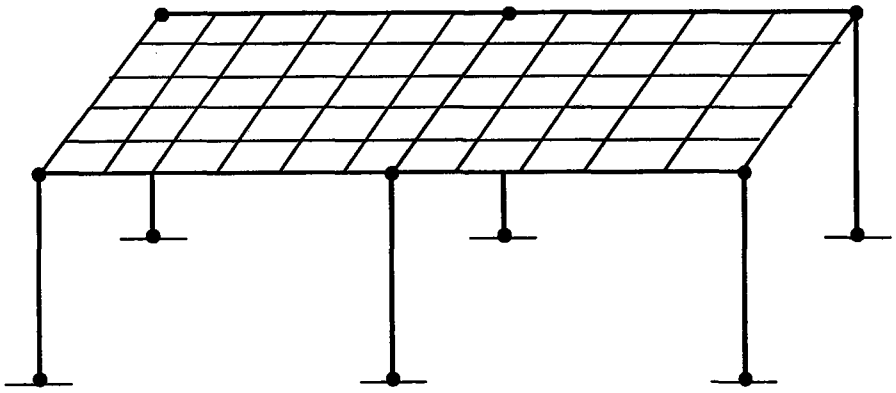


Figure 2.1 A typical discretization of a structure

The finite element modeling and the analysis of this simple structure would be a lot easier if it were possible to represent deformation behavior of the floor diaphragm in terms of the boundary nodes indicated in Figure 2.1. The macro element idea emerges just from this desire in which the behaviour of the element is defined in terms of some coupling nodes along the element boundary. A typical macro element is shown in Figure 2.2.

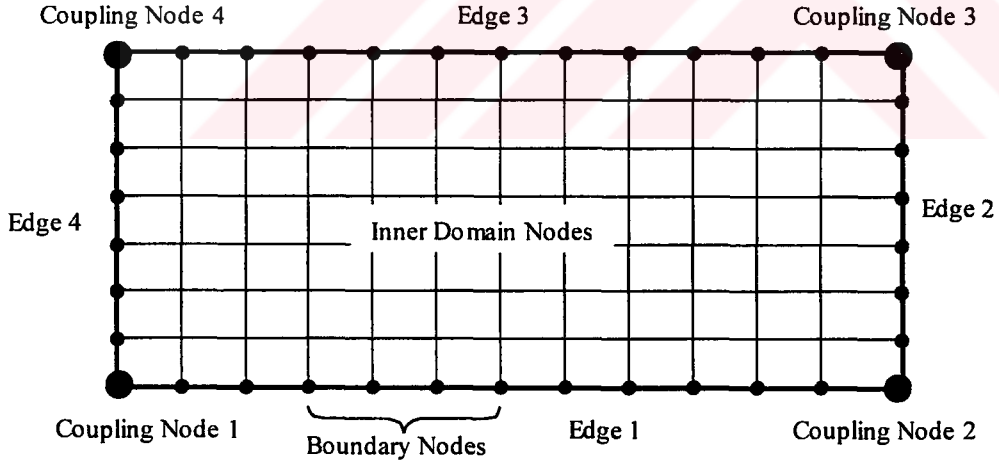


Figure 2.2 The Macro Element and its components

In the generation of property matrices for such an element the element region is discretized and meshed and a regular finite element model is built. Next, the inner domain nodes of the two-dimensional macro element are condensed out to the boundary nodes along the element edges, and then the effect of the boundary nodes is transferred to the coupling nodes by the help of some constraint equations. The constraint equations impose the prescribed displacement fields along the straight line segments between adjacent coupling nodes on the perimeter of the macro element region. These displacement fields are described in the following sections.

2.2 Assumed In-Plane Displacements along Boundaries

It is possible to describe the displacements and rotations of a boundary node along any edge of a macro element in terms of the displacements of the end nodes of the corresponding edge which are also the coupling nodes of the macro element. Imagine a macro element in global Cartesian plane XY and its edge IJ and let us define a typical local Cartesian coordinate system 123 for the boundary. Axis1 is defined as a vector running from Node I to Node J and the axis 3 is taken to be parallel to the global Z-axis. The in-plane displacement components at the coupling nodes I and J and at an arbitrary edge node K (U_{1I}, U_{2I}, θ_{ZI}, U_{1J}, U_{2J}, θ_{ZJ}, U_{1K}, U_{2K}, θ_{ZK}) are shown in figure 2.3 below.

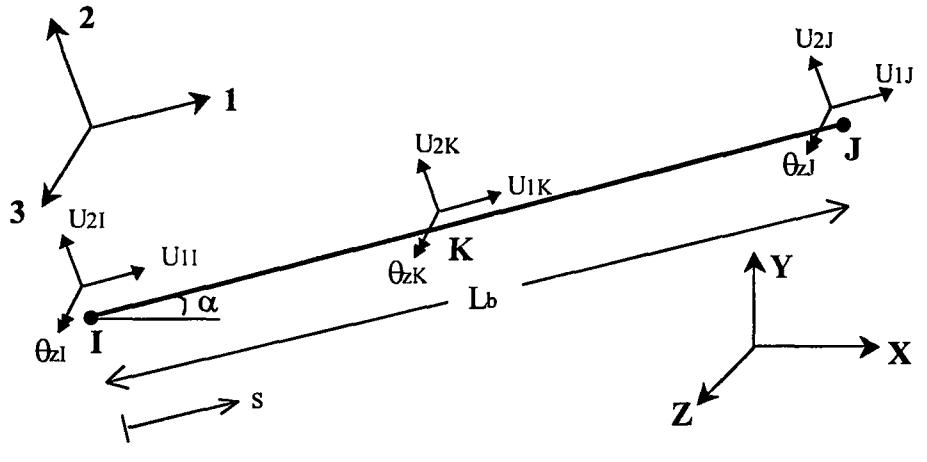


Figure 2.3 In-plane degrees of freedom along a typical edge

For the tangential displacement $U_1(\xi)$ along the local axis 1, a linear variation is assumed:

$$U_1(\xi) = N_1.U_{1I} + N_2.U_{1J} \tag{2.1}$$

$$N_1 = 1 - \xi \tag{2.2}$$

$$N_2 = \xi \tag{2.3}$$

where ξ = s/L in which s defines the distance along the local axis 1 measured from Node I.

For the normal displacement $U_2(\xi)$ along the local axis 1, a cubic variation is assumed. Using first degree Hermitte polynomials the normal displacement field is given by:

$$U_2(\xi) = N_3.U_{2I} + N_4.w_{ZI} + N_5.U_{2J} + N_6.w_{ZJ} \tag{2.4}$$

$$N_3(\xi) = 2\xi^3 - 3\xi^2 + 1 \tag{2.5}$$

$$N_4(\xi) = L_b.(\xi^3 - 2\xi^2 + \xi) \tag{2.6}$$

$$N_5(\xi) = -2\xi^3 + 3\xi^2 \tag{2.7}$$

$$N_6(\xi) = L_b.(\xi^3 - \xi^2) \tag{2.8}$$

where L_b is the length of the boundary. In case of small deformations the relationship between the rotation in Z-direction and normal displacement can be expressed as $\theta_Z(s) = U_{2,s}$. After the application of the chain rule, the expression becomes $\theta_Z(\xi) = U_{2,\xi} / L_b$.

For the rotation about Z-axis, a quadratic variation becomes:

$$\theta_Z(\xi) = dN_3.U_{2I} + dN_4.\theta_{ZI} + dN_5.U_{2J} + dN6.\theta_{ZJ} \quad (2.9)$$

$$dN_3(\xi) = (6\xi^2 - 6\xi) / L_b \quad (2.10)$$

$$dN_4(\xi) = 3\xi^2 - 4\xi + 1 \quad (2.11)$$

$$dN_5(\xi) = (-6\xi^2 + 6\xi) / L_b \quad (2.12)$$

$$dN_6(\xi) = 3\xi^2 - 2\xi \quad (2.13)$$

2.3 Assumed Out-Of-Plane Displacements along Boundaries

The out-of-plane displacement components at the coupling nodes I and J and at an arbitrary edge node K (U_{ZI}, θ_{1I}, θ_{2I}, U_{ZJ}, θ_{1J}, θ_{2J}, U_{ZK}, θ_{1K}, θ_{2K}) are shown in Figure 2.4.

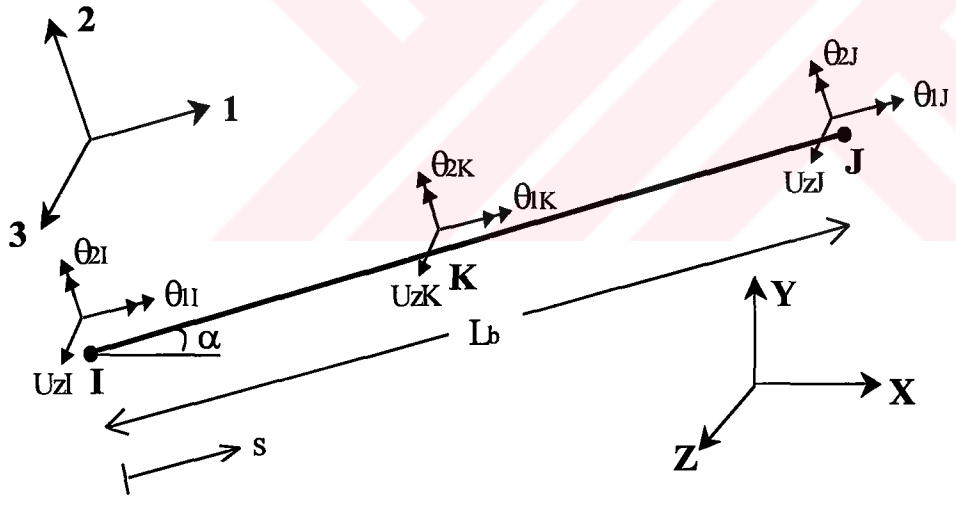


Figure 2.4 Out-of plane degrees of freedom along a typical edge

For the out-of plane displacement U_z, a cubic variation is assumed:

$$U_z(\xi) = N_3.U_{zI} - N_4.\ \theta_{2I} + N_5.U_{zJ} - N_6.\ \theta_{2J} \qquad (2.14)$$

The interpolation functions N_3, N_4, N_5 and N_6 are defined by Equations 2.5-2.8.

For the rotation around local axis 1, a linear variation is assumed. Hence the interpolation function is:

$$\theta_1(\xi) = N_1.\theta_{1I} + N_2.\theta_{1J} \qquad (2.15)$$

Similarly, the interpolation functions N_1 and N_2 are defined in Equations 2.2 and 2.3. It is possible to write similar expressions as in Equation 2.9 for the rotations around axis 2 for the case of small deformations. So, for a cubic interpolation, the rotation variation becomes:

$$\theta_2(\xi) = -dN_3.U_{zI} + dN_4.\theta_{2I} - dN_5.U_{zJ} + dN_6.\theta_{2J} \qquad (2.16)$$

The derivatives of the Hermitian Cubic interpolation functions are given in Equations 2.10-2.13.

2.4 Methods for the Imposition of Constraint Equations

In a general structural system, there may be a relationship among different degrees of freedom or the value of a degree of freedom may be prescribed. Constraint equations are the tools that impose such relationships or single-node value on the system.

There are mainly three different methods utilized for imposing constraint equations; transformation, Lagrange multipliers and penalty functions. In this study, transformation method is preferred since it does not disturb the symmetry of the stiffness matrix. Also, it is possible to apply it at element level.

Lagrange multipliers will be very effective if a few constraints are applied to a large number of degrees of freedom. However, it can be imposed only at structural level. The size of the equations increase and the stiffness matrix becomes non-positive definite. Hence, special solution algorithms are required instead of regular equation solvers.

In the penalty method, the penalty numbers are added to the stiffness matrix and force vector to impose zero or non-zero values of the degrees of freedom. The main problem in this method is the determination of the penalty number. It must be large enough to be effective but not so large to prevent numerical difficulties.

2.5 The Transformation Method

It is possible to write the constraint equations that relate the degrees of freedom in the form

$$[C].\{D\} = \{Q\} \tag{2.17}$$

where [C] is the matrix of constraint constants, {D} is the displacement matrix and {Q} contains constants. For a common case, {Q} is taken as equal to zero. The Equation 2.17 can be partitioned in the following form

$$\begin{bmatrix} \underline{C}_r & \underline{C}_c \end{bmatrix} \cdot \begin{bmatrix} \underline{D}_r \\ \underline{D}_c \end{bmatrix} = \underline{0} \tag{2.18}$$

in which the subscript r represents the degrees of freedoms to be retained and c refers to those which are to be constrained. The Equation 2.18 can be expanded as

$$\underline{C}_r.\underline{D}_r + \underline{C}_c.\underline{D}_c = 0 \tag{2.19}$$

Solving Equation 2.19 for $\underline{D}_c$ gives:

$$\underline{D}_c = - [\ \underline{C}_c\ ]^{-1}.\underline{C}_r.\underline{D}_r \tag{2.20}$$

Now, let

$$\underline{C}_{cr} = - [\ \underline{C}_c\ ]^{-1}.\underline{C}_r \tag{2.21}$$

Inserting Equation 2.21 into Equation 2.20 gives

$$\underline{D}_c = \underline{C}_{cr}.\underline{D}_r \tag{2.22}$$

The stiffness matrix has to be partitioned into two parts; retained degrees of freedom and the constrained degrees of freedom, for the application of constraint equations. The partitioned stiffness matrix will have the form:

$$\begin{bmatrix} \underline{K}_{rr} & \underline{K}_{rc} \\ \underline{K}_{cr} & \underline{K}_{cc} \end{bmatrix} \cdot \begin{bmatrix} \underline{D}_r \\ \underline{D}_c \end{bmatrix} = \begin{bmatrix} \underline{R}_r \\ \underline{R}_c \end{bmatrix} \tag{2.23}$$

The matrix in Equation 2.23 can be expanded into separate equations:

$$\underline{K}_{rr} \cdot \underline{D}_r + \underline{K}_{rc} \cdot \underline{D}_c = \underline{R}_r \tag{2.24}$$

$$\underline{K}_{cr} \cdot \underline{D}_r + \underline{K}_{cc} \cdot \underline{D}_c = \underline{R}_c \tag{2.25}$$

Now, substituting the Equation 2.22 into the Equation 2.24 and 2.25

$$(\underline{K}_{rr} + \underline{K}_{rc} \cdot \underline{C}_{cr}) \cdot \underline{D}_r = \underline{R}_r \tag{2.26}$$

$$(\underline{K}_{cr} + \underline{K}_{cc} \cdot \underline{C}_{cr}) \cdot \underline{D}_c = \underline{R}_c \tag{2.27}$$

are obtained. The solution of Equation 2.26 will give retained displacement vector $\underline{D}_r$. Then, by the help of Equation 2.27, the constrained displacements can be obtained. However, the left-hand side of the Equation 2.26 produces an unsymmetrical stiffness matrix that increases the solution time significantly. In order to avoid this deficiency, the following modifications can be made.

Defining a vector $\underline{R}^p_r$ to transfer the constrained forces to the retained degrees of freedom will produce total effective loads at the retained degrees of freedom. The vector $\underline{R}^p_r$ will have the form:

$$\underline{R}^{p}_{r} = \underline{C}_{cr}^{T}.\underline{R}_{c} \tag{2.28}$$

then the effective load vector will be:

$$\underline{R}^{t}_{r} = \underline{R}_{r} + \underline{R}^{p}_{r} \tag{2.29}$$

Now, arrange the equations (2.26) and (2.27) by using the equations (2.29), the condensed load vector and the stiffness matrix are obtained as

$$\underline{R}_{con} = \underline{R}^{t}_{r} \tag{2.30}$$

$$\underline{K}_{con} = (\underline{K}_{rr} + \underline{K}_{rc}.\underline{C}_{cr} + \underline{C}_{cr}^{T}.\underline{K}_{cr} + \underline{C}_{cr}^{T}.\underline{K}_{cc}.\underline{C}_{cr}).\underline{D}_{r} \tag{2.31}$$

2.6 Constraint Equations for a Single Node

Derivation of constraint matrices for a single node by using the same interpolation functions was previously done by H.Argeso[12]. The same constraint matrices are used in this work. For the completeness of the subject, the basic steps of the derivation is repeated here.

For a typical straight boundary, the active degrees of freedom are shown in Figure 2.5. There is only a single node on the edge. The displacement interpolations for this node are as follows:

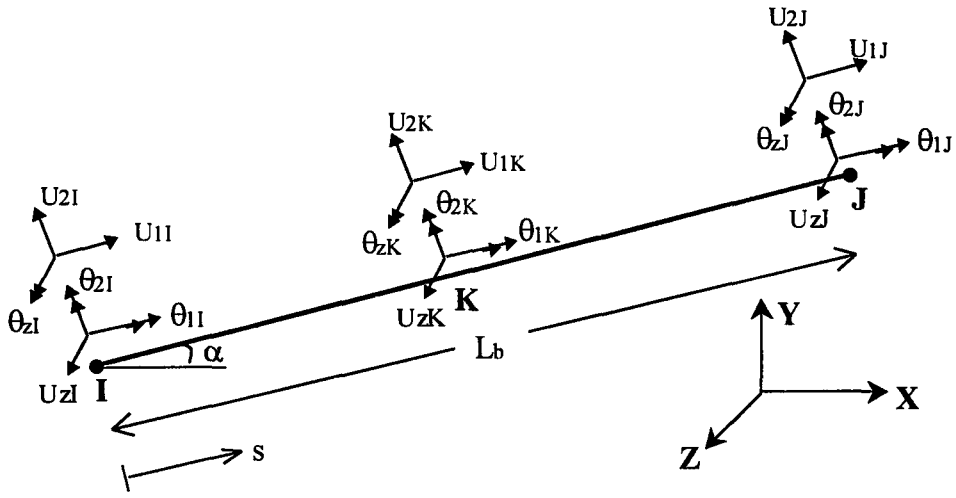


Figure 2.5 Active degrees of freedom along a boundary

$$U_1(\xi) = [N_1 \quad N_2] \cdot \begin{bmatrix} U_{1I} \\ U_{1J} \end{bmatrix} \tag{2.32}$$

$$U_2(\xi) = [N_3 \quad N_4 \quad N_5 \quad N_6] \cdot \begin{bmatrix} U_{2I} \\ \theta_{zI} \\ U_{2J} \\ \theta_{zJ} \end{bmatrix} \tag{2.33}$$

$$\theta z(\xi) = [dN_3 \quad dN_4 \quad dN_5 \quad dN_6] \cdot \begin{bmatrix} U_{2I} \\ \theta_{zI} \\ U_{2J} \\ \theta_{zJ} \end{bmatrix} \tag{2.34}$$

$$Uz(\xi) = [N_3 \quad -N_4 \quad N_5 \quad -N_6] \cdot \begin{bmatrix} U_{zI} \\ \theta_{2I} \\ U_{zJ} \\ \theta_{2J} \end{bmatrix} \tag{2.35}$$

$$\theta_1(\xi) = [N_1 \quad N_2] \cdot \begin{bmatrix} \theta_{1I} \\ \theta_{1J} \end{bmatrix} \tag{2.36}$$

$$\theta_2(\xi) = \begin{bmatrix} -dN_3 & dN_4 & -dN_5 & dN_6 \end{bmatrix} \cdot \begin{bmatrix} U_{zI} \\ \theta_{2I} \\ U_{zJ} \\ \theta_{2J} \end{bmatrix} \tag{2.37}$$

The interpolation functions and their derivatives are the functions of ξ. For the degrees of freedom at the boundary node K, ξ will be equal to ξ_K. With the above interpolation functions, the $\underline{C}_r$ matrix becomes:

$$C_r = \begin{bmatrix} \underline{A} & 0 & \underline{C} & 0 \\ 0 & \underline{B} & 0 & \underline{D} \end{bmatrix} \tag{2.38}$$

and the matrix $\underline{C}_c$ is:

$$C_c = \begin{bmatrix} -\underline{I} & 0 \\ 0 & -\underline{I} \end{bmatrix} \tag{2.39}$$

where A,B,C,D are 3x3 matrices and $\underline{I}$ is the identity matrix. The explicit forms of these terms are as follows:

$$A = \begin{bmatrix} N_1(\xi_k) & 0 & 0 \\ 0 & N_3(\xi_k) & N_4(\xi_k) \\ 0 & dN_3(\xi_k) & dN_4(\xi_k) \end{bmatrix} \tag{2.40}$$

$$B = \begin{bmatrix} N_3(\xi_k) & 0 & -N_4(\xi_k) \\ 0 & N_1(\xi_k) & 0 \\ -dN_3(\xi_k) & 0 & dN_4(\xi_k) \end{bmatrix} \tag{2.41}$$

$$C=\begin{bmatrix} N_2(\xi_k) & 0 & 0 \\ 0 & N_5(\xi_k) & N_6(\xi_k) \\ 0 & dN_5(\xi_k) & dN_6(\xi_k) \end{bmatrix} \tag{2.42}$$

$$D=\begin{bmatrix} N_5(\xi_k) & 0 & -N_6(\xi_k) \\ 0 & N_2(\xi_k) & 0 \\ -dN_5(\xi_k) & 0 & dN_6(\xi_k) \end{bmatrix} \tag{2.43}$$

Similarly the vector $\underline{D}_r$ is partitioned as:

$$D_r=\begin{bmatrix} U_I \\ U_J \end{bmatrix} \tag{2.44}$$

and $\underline{D}_c$ represents the degrees of freedom at the boundary node K.

Using the Equation 2.18 one can relate the degrees of freedom at boundary node K with degrees of freedom at the coupling nodes I and J. However, the derivation of this relationship was done in the local coordinates of the edge. It should be transformed to the global coordinates. Two transformation matrices, one for the in-plane deformations and the other for the out-of plane deformations are used for this purpose. Note that, the local axis 3 and the global axis Z are taken to be parallel to each other. Therefore the transformation is done in two dimensions.

The transformation matrix for the in-plane deformations,

$$a_1 = \begin{bmatrix} \cos\alpha & \sin\alpha & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{2.45}$$

and the transformation matrix for the out-of plane deformations:

$$a_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & \sin\alpha \\ 0 & -\sin\alpha & \cos\alpha \end{bmatrix} \tag{2.46}$$

where α is the angle between the local axis 1 and global axis X measured in the counterclockwise direction from +X.

By using the above transformation matrices, the relationship between the local coordinate system and the global coordinate system can be written as:

$$\underline{U}_{glbInPlane} = \underline{a}_1^{\,T} \cdot \underline{U}_{lclInPlane} \tag{2.47}$$

$$\underline{U}_{glbOutPlane} = \underline{a}_2^{\,T} \cdot \underline{U}_{lclOutPlane} \tag{2.48}$$

It is required to calculate the $\underline{C}_{cr}$ matrix for the transformation method. By first applying the coordinate transformation and then using the Equation 2.21, the $\underline{C}_{cr}$ matrix is obtained as:

$$C_{cr} = \begin{bmatrix} \underline{ZI}_1 & 0 & \underline{ZJ}_1 & 0 \\ 0 & \underline{ZI}_2 & 0 & \underline{ZJ}_2 \end{bmatrix} \tag{2.49}$$

where

$$ZI_1 = \begin{bmatrix} c^2 N_1(\xi_k) + s^2 N_3(\xi_k) & cs.N_1(\xi_k) - cs.N_3(\xi_k) & -s.N_4(\xi_k) \\ cs.N_1(\xi_k) - cs.N_3(\xi_k) & s^2 N_1(\xi_k) + c^2 N_3(\xi_k) & c.N_4(\xi_k) \\ -s.dN_3(\xi_k) & c.dN_3(\xi_k) & dN_4(\xi_k) \end{bmatrix} \quad (2.50)$$

$$ZI_2 = \begin{bmatrix} N_3(\xi_k) & s.N_4(\xi_k) & -c.N_4(\xi_k) \\ s.dN_3(\xi_k) & c^2 N_1(\xi_k) + s^2 dN_4(\xi_k) & cs.N_1(\xi_k) - cs.dN_4(\xi_k) \\ -c.dN_3(\xi_k) & cs.N_1(\xi_k) - cs.dN_4(\xi_k) & s^2 N_1(\xi_k) + c^2 dN_4(\xi_k) \end{bmatrix} \quad (2.51)$$

$$ZJ_1 = \begin{bmatrix} c^2 N_2(\xi_k) + s^2 N_5(\xi_k) & cs.N_2(\xi_k) - cs.N_5(\xi_k) & -s.N_6(\xi_k) \\ s.dN_5(\xi_k) & c^2 N_2(\xi_k) + s^2 dN_6(\xi_k) & c.N_6(\xi_k) \\ -s.dN_5(\xi_k) & c.dN_5(\xi_k) & dN_6(\xi_k) \end{bmatrix} \quad (2.52)$$

$$ZJ_2 = \begin{bmatrix} N_5(\xi_k) & s.N_6(\xi_k) & -c.N_6(\xi_k) \\ s.dN_5(\xi_k) & c^2 N_2(\xi_k) + s^2 dN_6(\xi_k) & cs.N_2(\xi_k) - cs.dN_6(\xi_k) \\ -c.dN_5(\xi_k) & cs.N_2(\xi_k) - cs.dN_6(\xi_k) & s^2 N_2(\xi_k) + c^2 dN_6(\xi_k) \end{bmatrix} \quad (2.53)$$

In the above equations s, c, cs represent cosα, sinα and sinα.cosα, respectively. Note also that the matrices $\underline{ZI}_1$ and $\underline{ZJ}_1$ are the equations for the in-plane actions and similarly $\underline{ZI}_2$ and $\underline{ZJ}_2$ are for the out-of plane actions.

CHAPTER 3

OBJECT-ORIENTED ANALYSIS AND DESIGN OF A FINITE ELEMENT PROGRAM WITH MACRO ELEMENTS

3.1 Introduction

The program developed for this thesis has a complete object-oriented structure. It consists of two different parts; the main program and the mathematical library. The main program is a simple control program that uses the mathematical library during solution for some matrix operations. The mathematical library is created as a dynamic-link library (DLL). Dynamic link libraries are very advantageous especially when the size of the program becomes larger. They also provide the possibility of using different programming languages in the same program. The mathematical library was developed using FORTRAN 90 language since it is very strong in number crunching. C++ is used for the main program because it provides object-oriented programming.

The design of the program is made to run with an event-oriented and object based graphical interface. However, the current version of the program works under DOS operating system and needs an input file. In its current state the computer program "NESONEL" has seventy different classes. These classes are divided into five distinct parts: Mathematical classes, geometric classes, mechanical classes, database classes and system classes.

The structure of these classes and the relationship among them are explained later in this chapter.

3.2 Object-Oriented Analysis and Design of the Program

The most important part of the software development is the definition of the problem and the determination of the requirements. A good analysis of the problem and design of the program structure will facilitate the future modifications, change requirements and extensions. Simplification of any kind of a complex problem is another important point. Decision of the right abstractions, in other words, dividing the problem into simpler parts will speed up the development time, decrease the number of programmer mistakes and provide a strong structure. The next step in the design is the determination of the relationships between the parts of the problem.

The project is considered to be "the development of a computer code for the stress analysis of a solid structure by using the finite element method". The program is desired to meet the following requirements:

- Easy to link to a graphical interface or a CAD program.
- Accept any kind of geometric information to define the finite element model of a general structure.
- Allow addition of different kinds of finite elements without requiring excessive modification in the program structure.
- Easy extension to handle the non-linear problems by the addition of non-linear solution algorithms.
- Contain a wide range of post-processing functions.
- The classes must be general to be used with different programs.

In the light of the above requirements, the project is divided into four main parts.

- the geometric problem,
- the mechanical problem,
- the mathematical problem, and
- the database problem

Each problem is examined separately and for each group specific classes are developed. In the object-oriented design, two different models are utilized, static and dynamic models. The static model describes the pathways that the messages between the objects will use. It also specifies the classes within the application and the relations that exist among them. On the other hand, dynamic model describes the sequence of messages that flow between collaborating objects. It specifies the order in which the messages are sent, the types of messages and the data that messages carry.

Only the static models for each group of classes is presented in this chapter. The Booch notation is used for the description of the class relationships. The detailed explanation and its C++ implementation is described in Appendix B

3.3 Geometric Problem

3.3.1 The Definition of the Problem

Three types of geometric entities are considered to be sufficient to define the problem domain; points, lines and polygons. The following are the major requirements to be satisfied in the process of class creation and determination of the relations among them.

- Definition of the problem domain and the interpretation of the results.

    - Some of the entities must have a parametric structure. In other words, the parametric entities must have pointers to their neighbouring elements. For example, a nodal point must know its neighbouring polygons and vice versa. By this way, the condensation of the inner domain nodes in a macro element, finding the polygon over which a point load reides becomes much easier and faster. Moreover, the nodal points along an edge must be known by the edge in order to facilitate the application of the edge boundary conditions and the constraint equations on the boundary nodes in a macro element.
    - An object of a given class can represent different physical entities. A polygon is sometimes a main polygon and sometimes a macro polygon. Similarly, a line object may sometimes represent a line load and sometimes a polygon boundary. Especially, a node object can have different properties. It can be used as a coupling node, an inner domain node or a boundary node.
    - A geometric object must be able to hold different kinds of data. For instance, the point object can store the magnitude of a point load or the node object can store the restraint information of a coupling node.

- Mesh Generation

    A polygon can be divided into smaller polygon for the purpose of finite element analysis. The mesh generator must be able to construct a parametric relationship between the entities. It must also be able to determine the element neighbours, edge nodes and assign the edge boundary conditions.

Another important problem for the mesh generator is its ability to handle the common edges of two neighbouring polygons. The nodes must be created only once along such common edges.

3.3.2 Design

First, an abstract class, GeometricEntity, is created from which all geometric classes are inherited. It has the AttributeList property that is used to store all kinds of data for an object. AttributeList can contain an infinite number of AttributeData. AttributeData has two properties, AttributeCode and Data. AttributeCode is used to learn the type of stored data. In the program, ObjectCode, ElementCode and ElementType AttributeCodes are used to identify the geometric entities. Also, the GeometricEntity has another list object to hold its corresponding Mechanical Entities.

The parametric classes, Node3d, LinkedLine and Polygon are developed to provide the parametric structure of the domain. All three classes have a geometric entity list to hold the pointers of their neighbouring elements.

Three classes are created for the mesh generation; Quadrilateral Mesh Generator, which is an inherited class from Surface Generator, an abstract class, Line Divider and Point Assembler. Generate() method is added to the Geometric Entity class. During the interpretation of the geometric data, generate() method is called for each entity. If a generator object is found in the object list of the entity, the generation will be carried out for that object. When creating a mesh for a polygon, the edge nodes are added to the geometric object list of the edge. Hence, when another mesh is created for the neighbouring polygon, the previously created edge node pointer is obtained from the edge object and used in element creation.

Static model of geometric classes are shown in Figure 3.1, below.

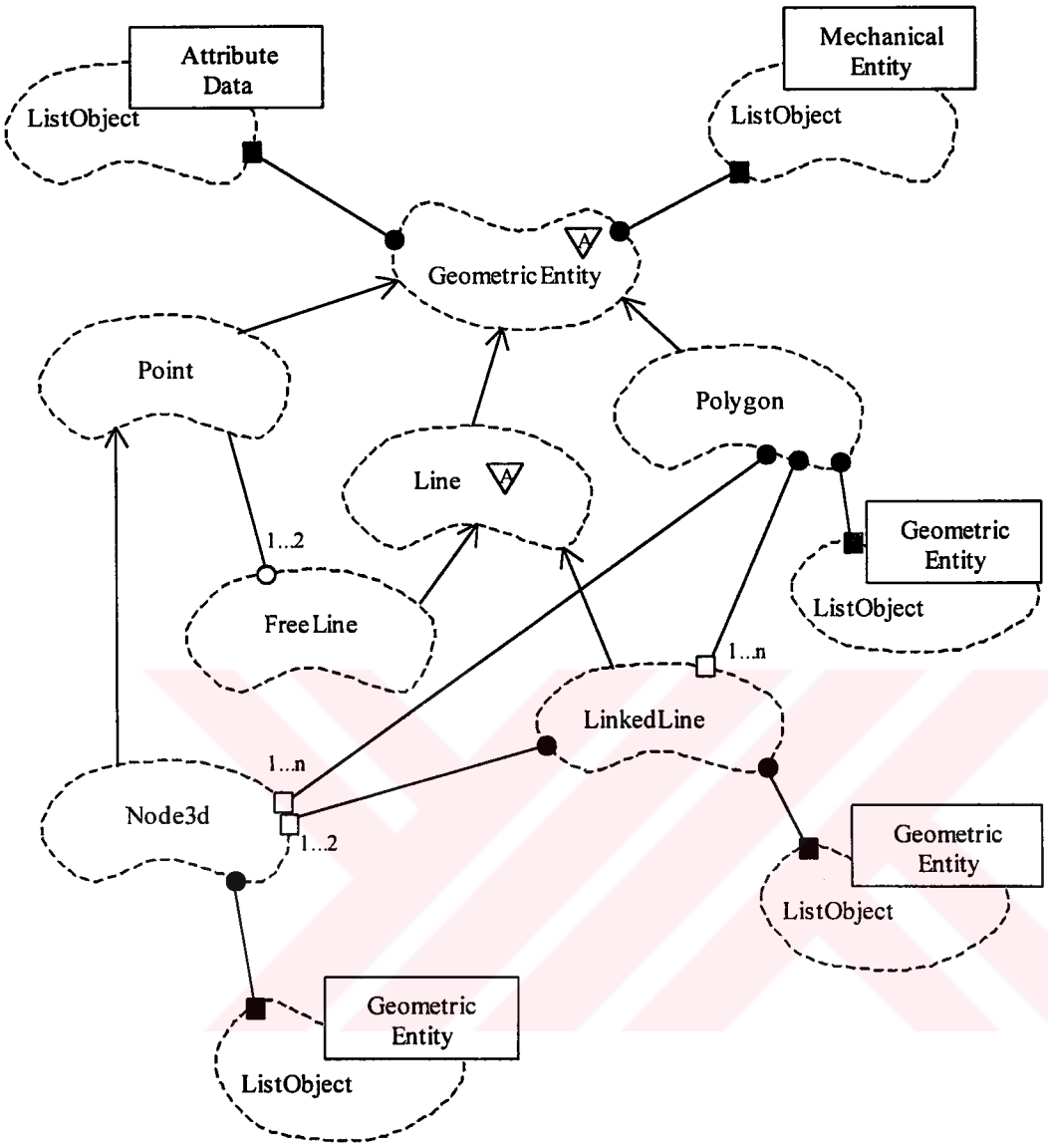


Figure 3.1 The static model of geometric classes

3.4 Mechanical Problem

3.4.1 Problem Definition

The mechanical problem involves the formation of the structural stiffness matrix and the load vector. It also requires the calculation of the element stresses and the nodal forces after the solution. The mechanical classes must possess the following qualities

- Flexibility: Formation of the structural stiffness matrix and the load vector must be independent of the elements used. Using membrane elements or plate bending elements should not change the working of the solution algorithm.
- Extendibility: New elements must easily be adopted to the program without any change requirements in other classes. The newly added element must be able to assign its equation numbers, assemble itself, calculate its stresses and the nodal responses.

3.4.2 Design

First, an abstract Element class is defined. It has virtual assignDof() and assemble() methods. Also a Node class is defined for holding the equation numbers and nodal displacements. In the solution phase, the solver object calls the assignDof() method of each element. The elements know which degrees of freedom they are using and assigns them accordingly. Then, the solver object calls the assemble() method of each element for the formation of the structural stiffness matrix.

The abstract Element class has five main virtual functions that the new elements have to override. These functions can be classified in two groups. The ones which are used during the solution; assignDof() and assemble(), and the ones for post-processing; printS(), printD() and printNF().

Static model of mechanical classes are shown in Figure 3.2, below.

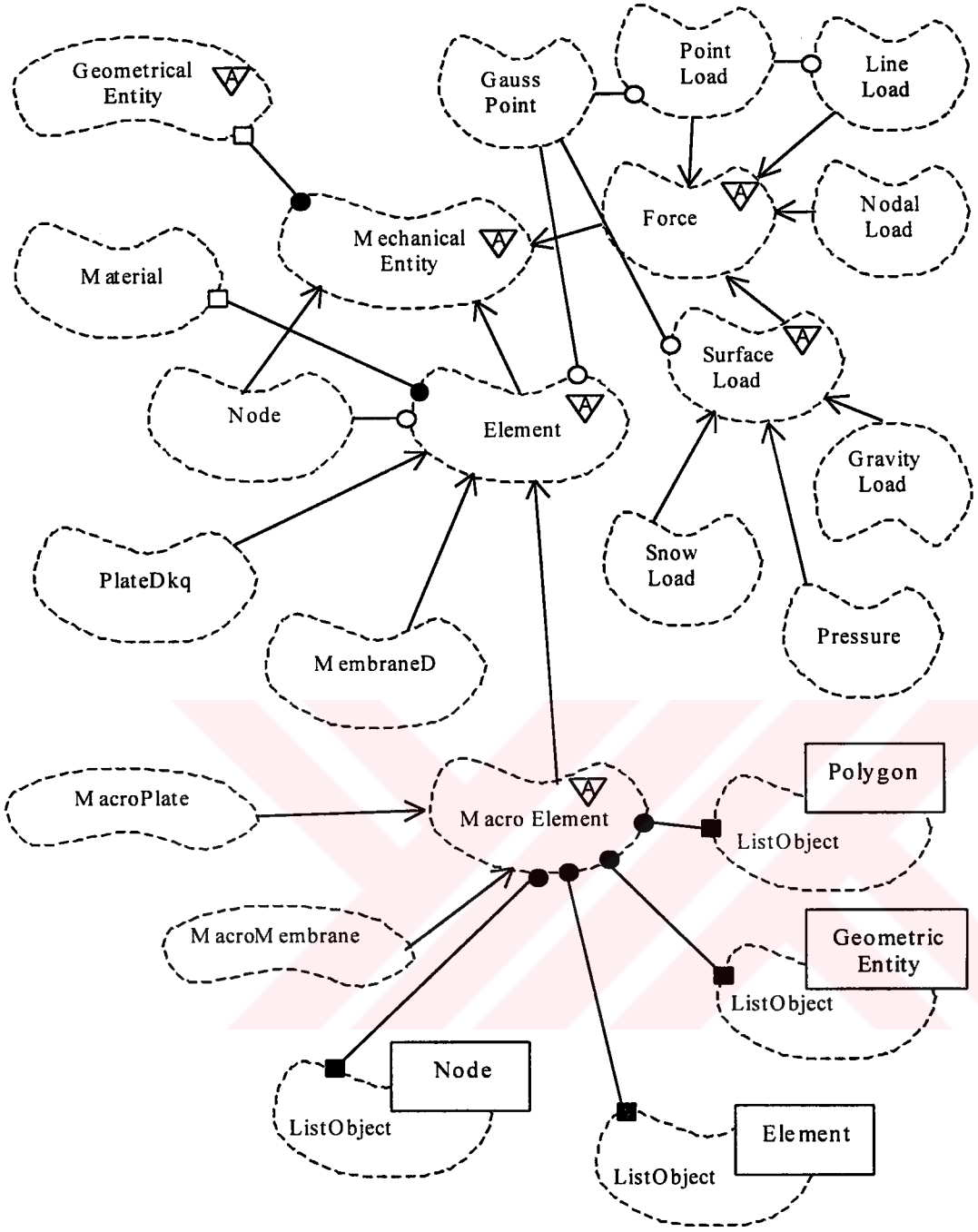


Figure 3.2. The static model of mechanical classes

3.5 Mathematical Problem

3.5.1 Matrix Objects

Finite element representation of a structure requires a good number of vectors and matrices and its calculations involve extensive matrix operations. The usage of the correct matrix types increases the efficiency and speed significantly. In the program NESONEL two main groups of matrix classes are developed; static and dynamic matrix classes. Sparse matrices are developed as dynamic matrices whose column sizes depend on the data added to them. The data is stored as a linked list. If a matrix object is of static matrix class, the storage space is allocated at the time of object creation and it is not possible to change the dimensions of the matrix afterwards.

Both of these matrix classes are inherited from an abstract class AMatrix which defines the basic operations common to all matrices. Two virtual functions, get() and set(), used for learning and changing the data respectively, are overridden by the other matrix classes according to their properties.

Static model of matrix classes is shown in Figure 3.3, below.

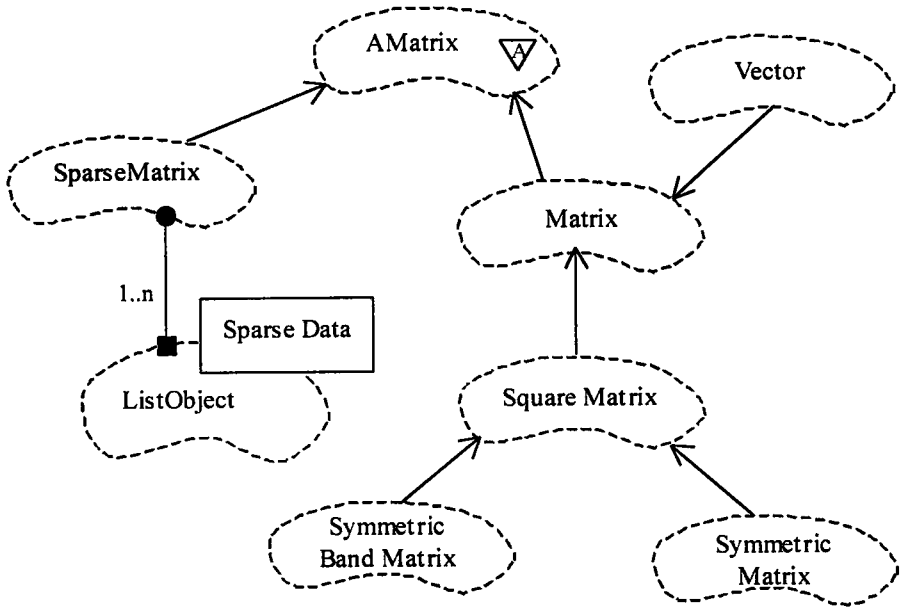


Figure 3.3. The static model of matrix classes.

3.5.3 Solver Objects

Currently, there are two linear equation solution algorithms developed for the program NESONEL. One of these solution algorithms is for the symmetric band matrix classes. After the assignment of equation numbers, the solver loops over each element and calculates the maximum half bandwidth. For a fast equation solution and low memory usage, a bandwidth reduction operation is required. The other solution algorithm is for sparse matrices. The sparse matrices do not need optimization. Their most important disadvantage is that they need to spend some time in searching for a specific data. However, the solution algorithm works very efficiently since it does not deal with the missing terms (zero coefficients) in the equations.

The static model of solver classes is shown in Figure 3.4, below.

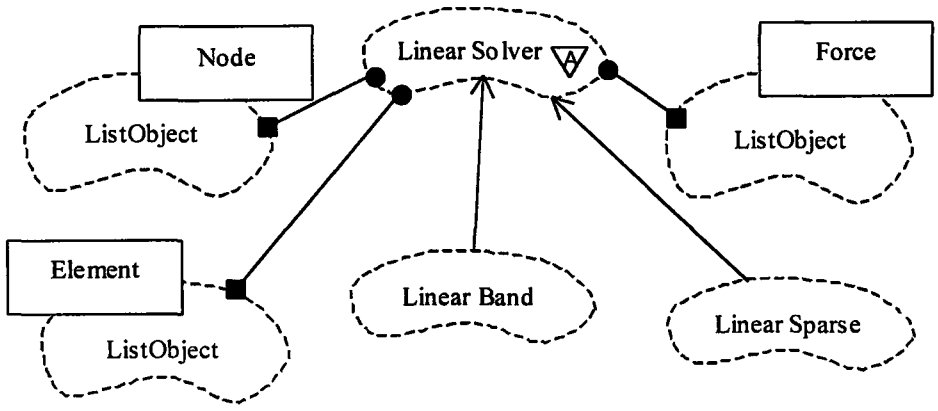


Figure 3.4 The static model of solver classes.

3.6 Database Problem

Finite element programs have to deal with many different kinds of elements. An efficient and flexible database structure is required for organizing and storing relevant data for such programs. The List class is developed to store any kind of objects. It creates a dynamic pointer object which points to the location of the previous and the next object in the memory space of the computer. In order to be able to add and store an arbitrary kind of object in the database of class List, all objects must be inherited from an abstract class. The Object abstract class is created and used for this purpose.

The static model of the database classes is shown in Figure 3.5, below.

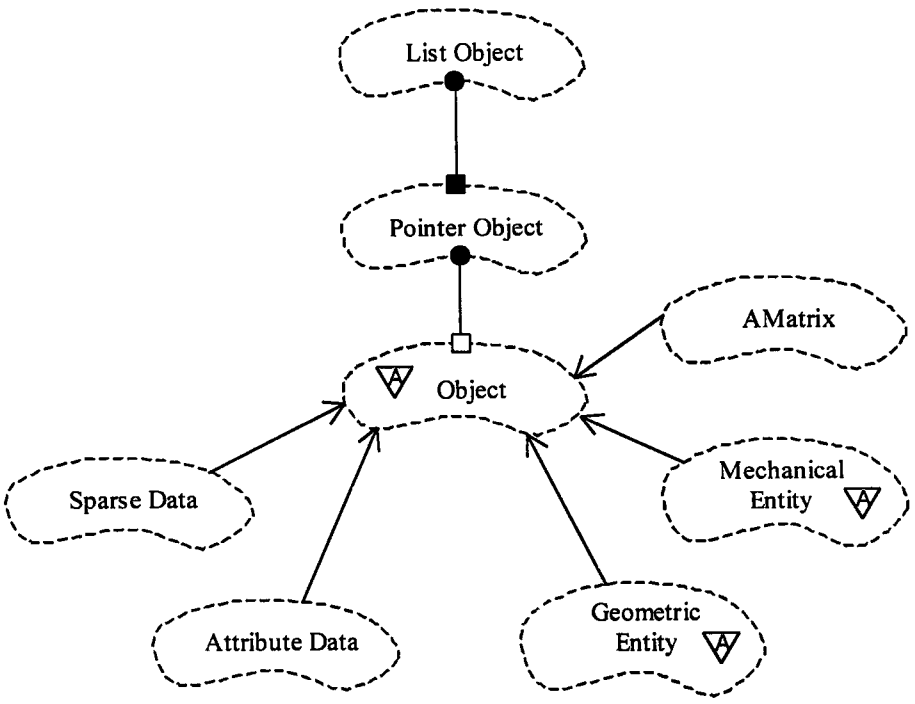


Figure 3.5. The static model of database classes.

3.7 The Macro Element Class

The macro elements are super elements which are composed of a varying number of simple finite elements. They are defined by the coupling nodes on their outer boundaries and they interface with the rest of the structure through these coupling nodes. The active degrees of freedom at the inner domain nodes are first eliminated by static condensation and then those at the boundary nodes are carried to the coupling nodes by the imposition of some constraint equations. The macro element algorithm works as a kind of a routine finite element algorithm consisting of the pre-processing, the solution and the post-processing (recovery) algorithms.

The macro element class is adopted to NESONEL by overriding the five virtual methods required for the addition of a new finite element. The algorithms for the macro element are shown in Figures 3.6 and 3.7, below.

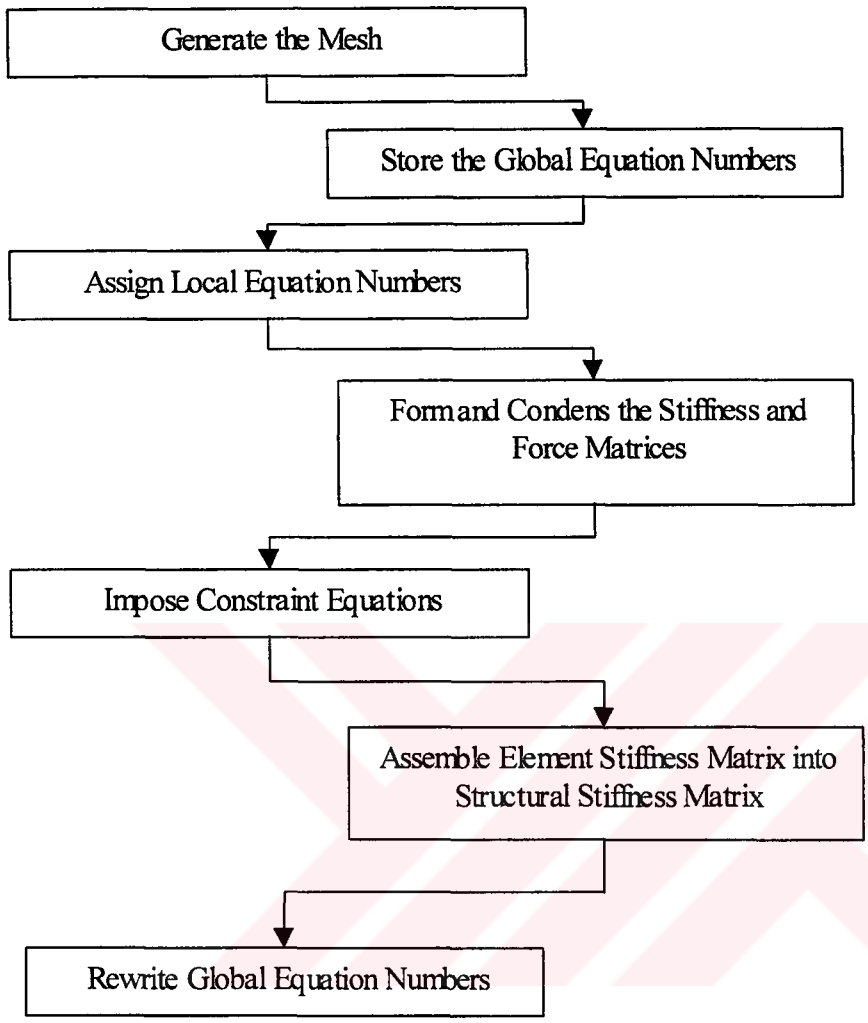


Figure 3.6 Flow chart of the pre-processing and solution algorithms.

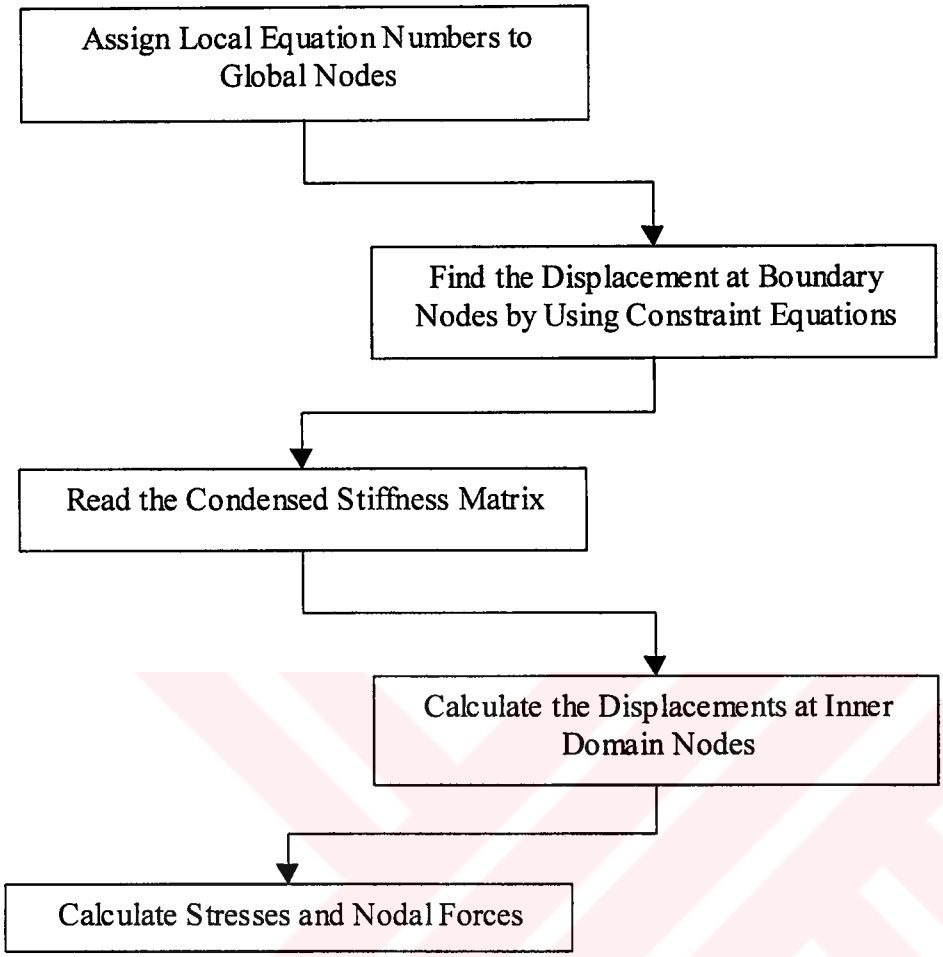


Figure 3.7 Flow chart of post-processing (recovery) algorithm

3.7.1 Local Equation Numbering

During the equation number assignment, first the coupling nodes and then the boundary nodes are numbered. Finally, the inner domain nodes are numbered. This makes it possible to detect the type of node (coupling, boundary or inner domain) from equation numbers. This numbering system facilitates the static condensation.

3.7.2 Static Condensation

The assemblage of element stiffness matrices and the condensation are carried out simultaneously. The dynamic sparse matrices are used to store the contributions of inner domain nodes. The parametric structure facilitates this algorithm significantly. Each node knows the polygons that it belongs to. For example, in Figure3.8, the inner domain node A stores the pointers of the polygons 1,2,3 and 4.

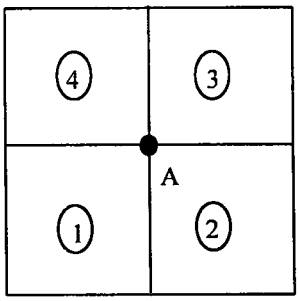


Figure 3.8 A typical inner domain node and its neighbouring elements

Each polygon stores the pointers to the mechanical elements that it represents as shown in Figure 3.9. For a stiffened plate, for example, the polygon will have pointers to two different elements; a pointer to a shell element and a pointer to a joist beam element.

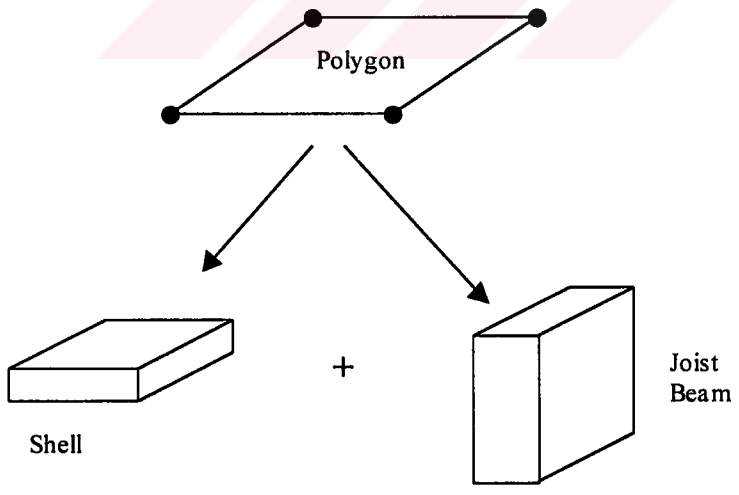


Figure 3.9 Geometric and mechanical entity relation

During the assembly phase, the mechanical elements owned by the neighbouring polygons of the inner domain node is assembled first. After having assembled all neighbouring elements, the inner node is eliminated since it does not get any other contributions from other elements.

The stiffness terms corresponding to boundary nodes are stored in a static square matrix. The contributions of inner domain nodes to boundary nodes and the inner domain node stiffness terms are stored in a dynamic sparse matrix. At the end of the elimination of each node, its stiffness terms are deleted from the sparse matrix to minimize the memory usage.

3.7.3 Imposition of Constraint Equations

In transformation method, the condensed stiffness matrix and the condensed load vector are given as

$$\underline{R}_{con} = \underline{R}_r + \underline{C}_{cr}^{\ T} \cdot \underline{R}_c \tag{3.1}$$

$$\underline{K}_{con} = \underline{K}_{rr} + \underline{K}_{rc}\underline{C}_{cr} + \underline{C}_{cr}^{\ T}\underline{K}_{cr} + \underline{C}_{cr}^{\ T}\underline{K}_{cc}\underline{C}_{cr} \tag{3.2}$$

The formulae above implies matrix multiplication and addition which requires considerable amount of time and memory. This demand increases rapidly as the size of the matrices increase. In the transformation method, it is possible to apply constraint equations node by node. Also, if the above equations are examined carefully, it will be seen that, they imply simple row and column operations.

As described by Reddy[2], the transformation can be done in the following way. Suppose that we have a system having six degrees of freedom. In this system, the fifth degree of freedom is related to fourth degree of freedom as

$$u_6 = u_5 \cdot c \tag{3.3}$$

The transformation equation for this case has the following form

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & c \end{bmatrix} \cdot \begin{Bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{Bmatrix} \tag{3.4}$$

In order to obtain the transformed stiffness matrix and the load vector, the following operations are to be carried out

$$\begin{aligned} &\underline{K}'_{ij} = \underline{K}_{ij} && \text{i,j = 1,2,3,4} \\ &\underline{K}'_{5i} = \underline{K}_{5i} + \underline{K}_{6i} \cdot c \\ &\underline{K}'_{i5} = \underline{K}_{i5} + \underline{K}_{i6} \cdot c && \text{i = 1,2..5} \\ &\underline{F}'_{i} = \underline{F}_{i} && \text{i,j = 1,2,3,4} \\ &\underline{F}'_{5} = \underline{F}_{5} + \underline{F}_{6} \cdot c \end{aligned} \tag{3.5}$$

If the constraint equations for a macro element are examined, it is seen that each coupling node degree of freedom is related to three boundary node degrees of freedom. Therefore, the above equations are modified as follows:

For i = 1 to nRow , j = 1 to nDof

For any coupling node I of the macro element

$$\underline{KK}_{i,nGlbI(j)} = \underline{KK}_{i,nGlbI(j)} + \sum_{k=1}^{nDof} \underline{KK}_{i,nCon(k)} \cdot \underline{CcrI}(j,k) \quad (3.6)$$

$$\underline{KK}_{nGlbI(j),i} = \underline{KK}_{nGlbI(j),i} + \sum_{k=1}^{nDof} \underline{KK}_{nCon(k),i} \cdot \underline{CcrI}(j,k) \quad (3.7)$$

For any coupling node J of the macro element

$$\underline{KK}_{i,nGlbI(j)} = \underline{KK}_{i,nGlbI(j)} + \sum_{k=1}^{nDof} \underline{KK}_{i,nCon(k)} \cdot \underline{CcrI}(j,k) \quad (3.8)$$

$$\underline{KK}_{nGlbJ(j),i} = \underline{KK}_{nGlbJ(j),i} + \sum_{k=1}^{nDof} \underline{KK}_{nCon(k),i} \cdot \underline{CcrJ}(j,k) \quad (3.9)$$

where

nDof: number of degrees of freedom per node

nCon: array of dof to be constrained (have a size of nDofx1)

nGlbI: coupling node dof for node I.

nGlbJ: coupling node dof for node J.

CcrI: constraint coefficients for node I.

CcrJ: constraint coefficients for node J.

CHAPTER 4

CONSTITUENT FINITE ELEMENTS OF THE PLATE/SHELL MACRO ELEMENT WITH STIFFENERS

4.1 Introduction

The plate/shell macro element with stiffeners is comprised of a number of plane stress membrane elements, thick and thin plate bending elements and beam bending elements. The membrane element is the one suggested by A.Ibrahimbegovic[39] and has drilling degrees of freedom. The thin plate element is the DKQ element of J.L.Batoz[30] and the thick plate element is the one also suggested by A.Ibrahimbegovic[31]. A Hermitian beam element is used for the stiffeners. Basic features of these finite elements are briefly summarized in the following sections.

4.2 Plate Bending Theory

A plate structure is defined as a structural element whose middle surface lies in a flat plate. It is characterised by its relatively small thickness compared with the in-plane dimensions. Hence, the behavior of the plate structure strongly depends on its thickness. Plates can be classified according to significance of their transverse shear deformations; thin plates and thick plates. Actually, there is no sharp line between the two classes of plates. In the field of civil engineering, both types of plate structures exist.

A typical plate element and the notation used for describing the stresses that act on its cross-sections are shown in Figure 4.1

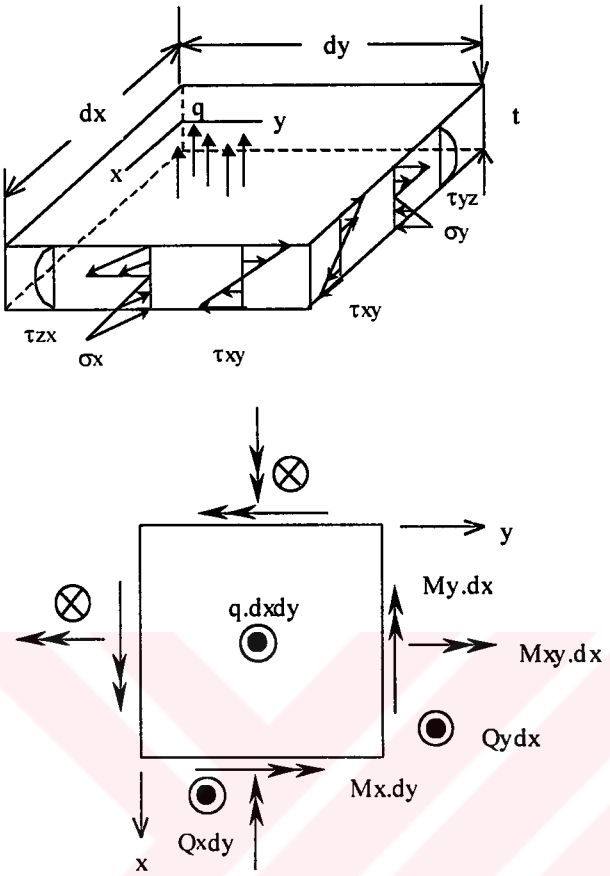


Figure 4.1 A plate element and the stresses acting on it.

The assumed stress distributions for a plate element are as follows:

- Normal stresses σ_x and σ_y vary linearly with z, and they are associated with the bending moments M_{xx} and M_{yy}.
- Shear stress τ_{xy} varies linearly with z, and it is associated with the twisting moment M_{xy}.
- Normal stress σ_z is negligible when compared with σ_x, σ_y, τ_{xy}.
- Transverse shear stresses τ_{yz}, τ_{zx}, vary quadratically with z.

- $\sigma_y = \sigma_x = \tau_{xy}$ = on the mid-surface z = 0.

The stress resultants, the moments and the shear forces, produced by the stresses can be calculated by integrating the corresponding stresses over the plate thickness.

$$Mx = \int_{-t/2}^{t/2} \sigma x \cdot z dz \qquad My = \int_{-t/2}^{t/2} \sigma_y \cdot z dz \qquad Mxy = \int_{-t/2}^{t/2} \tau xy \cdot z dz \qquad (4.1)$$

$$Qx = \int_{-t/2}^{t/2} \tau zx \cdot dz \qquad Qy = \int_{-t/2}^{t/2} \tau yz \cdot dz \qquad (4.2)$$

4.3 Kirchhoff Plate Bending Theory

Kirchhoff theory is applicable to thin plates in which transverse shear deformations are negligible. The main assumptions of the Kirchhoff plate bending theory are as follows:

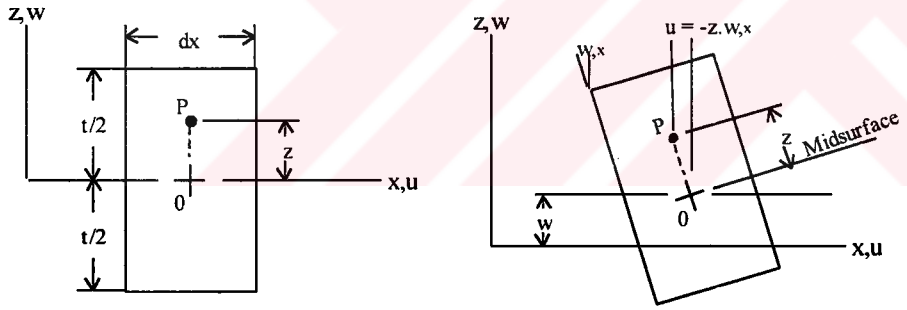


Figure 4.2 Deformations associated with Kirchhoff plate bending theory.

- Points on the mid-surface only move in z direction as the plate deforms in bending.

- A line that is straight and normal to the mid-surface before loading is assumed to remain straight and normal to the mid-surface after loading.
- Shear deformations are assumed to be zero.

Hence the strain-displacement relationship for a thin plate is as follows:

$$u = -z.w_{,x} \tag{4.3}$$

$$v = -z.\ w_{,y} \tag{4.4}$$

and,

$$\varepsilon_x = u_{,x} \quad = -z.\ w_{,xx} \tag{4.5}$$

$$\varepsilon_y = v_{,y} \quad = -z.\ w_{,yy} \tag{4.6}$$

$$\gamma_{xy} = u_{,y} + v_{,x} \quad = -2z.\ w_{,xy} \tag{4.7}$$

The stress-strain relationship for an isotropic thin plate is similar to the state of plane stress in which the stress components $\sigma_z, \tau_{yz}, \tau_{zx}$ are not involved for a plate in XY plane.

$$[D_k] = \frac{Et^3}{12(1-\mu^2)} \cdot \begin{bmatrix} 1 & \mu & 0 \\ \mu & 1 & 0 \\ 0 & 0 & \frac{1-\mu}{2} \end{bmatrix} \tag{4.8}$$

The moment curvature relation is obtained by inserting the Equations 4.5 to 4.8 into Equation 4.1 which results in the following:

$$\begin{Bmatrix} M_x \\ M_y \\ M_{xy} \end{Bmatrix} = [D_k] \cdot \begin{Bmatrix} w,_{xx} \\ w,_{yy} \\ 2w,_{xy} \end{Bmatrix} \tag{4.9}$$

4.4 Mindlin Plate Bending Theory

Mindlin plate elements account for the transverse shear deformations as well as the bending deformations. Therefore, they are appropriate for the analysis of thick plate structures The main assumptions of Mindlin plate bending theory are as follows:

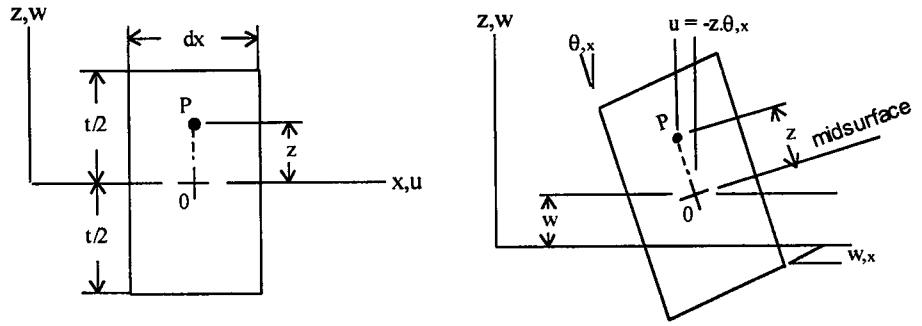


Figure 4.3 Deformations associated with Mindlin plate theory

- A line that is straight and normal to the mid-surface before loading is assumed to remain straight but not necessarily normal to the mid-surface after loading.
- Transverse shear deformation is allowed.
- The motion of a point not on the mid-surface depends on rotations θx and θy of lines that were normal to the mid-surface of the undeformed plate.

The strain-displacement relationship for a thick plate can be expressed as follows:

$$u = -z.\theta_x \tag{4.10}$$

$$v = -z.\theta_y \tag{4.11}$$

and,

$$\varepsilon_x = -z.\ \theta_{x,x} \tag{4.12}$$

$$\varepsilon_y = -z.\ \theta_{y,y} \tag{4.13}$$

$$\gamma_{xy} = -z.(\ \theta_{x,y} + \theta_{y,x}\ ) \tag{4.14}$$

$$\gamma_{yz} = w_{,y} - \theta_y \tag{4.15}$$

$$\gamma_{zx} = w_{,x} - \theta_x \tag{4.16}$$

The moment curvature relation is obtained using the same procedure as in Kirchhoff theory, but this time shear terms in Equation 4.2 must also be included. The resulting relation is as given below:

$$\begin{Bmatrix} M_x \\ M_y \\ M_{xy} \\ Q_x \\ Q_y \end{Bmatrix} = \begin{bmatrix} & & & 0 & 0 \\ & [D_k] & & 0 & 0 \\ & & & 0 & 0 \\ 0 & 0 & 0 & G_{yz}t & 0 \\ 0 & 0 & 0 & 0 & G_{zx}t \end{bmatrix} \cdot \begin{Bmatrix} \theta_{x,x} \\ \theta_{y,y} \\ \theta_{x,y} + \theta_{y,x} \\ \theta_y - w_{,y} \\ \theta_x - w_{,x} \end{Bmatrix} \tag{4.17}$$

4.5 Discrete Kirchhoff Quadrilateral Element

The formulation of DKQ element is based on the enforcement of zero transverse shear strain energy at some discrete points along the element edges. The total strain energy, U, and the element strain energy U_b^e of a Kirchhoff plate bending element are

$$U = \sum_e U_b^e \qquad U_b^e = \frac{1}{2}\iint_A [B]\{D_k\}[B]dxdy \tag{4.18}$$

where B is the strain displacement relation and D_k is the stress-strain relation. The B matrix is defined as:

$$B = \begin{bmatrix} \partial\beta x/\partial x \\ \partial\beta y/\partial y \\ \partial\beta x/\partial y + \partial\beta y/\partial x \end{bmatrix} \tag{4.19}$$

in which β_x and β_y are the rotations of the normal to the undeformed middle surface in X-Z and Y-Z planes, respectively.

For a typical quadrilateral plate element in Figure 4.4, the formulation is based on the following considerations:

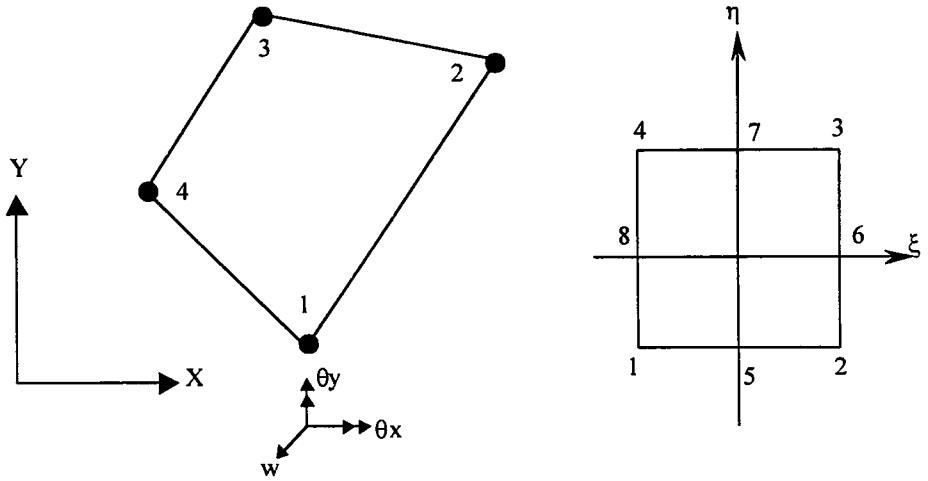


Figure 4.4 A typical quadrilateral element and reference coordinate system.

- β_x and β_y are defined by incomplete cubic polynomial expressions:

$$\beta x = \sum_{i=1}^{8} N_i \beta_{xi} \qquad \beta y = \sum_{i=1}^{8} N_i \beta_{yi} \tag{4.20}$$

- The Kirchhoff assumptions are introduced:

  a) at the corner nodes:

$$\begin{Bmatrix} \beta_{xi} + w,_{xi} \\ \beta_{yi} + w,_{yi} \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} \qquad \text{i} = 1, 2, 3, 4 \tag{4.21}$$

  b) at the mid-side nodes:

$$\beta_{sk} + w,_{sk} = 0 \qquad \text{k} = 1, 2, 3, 4 \tag{4.22}$$

- $w_{,sk}$ is the derivative of the transverse displacement w with respect to s at the mid-side node k, where w is defined by a cubic expression along each element side:

$$w,_{sk} = \frac{-3}{2l_{ij}}(w_i - w_j) - \frac{1}{4}(w,_{si} + w,_{sj}) \tag{4.23}$$

and k = 5, 6, 7, 8 for ij = 12, 23, 34, 41, respectively.

- β_n varies linearly along the sides

$$\beta_{nk} = \frac{1}{2}(\beta_{ni} + \beta_{nj}) = -\frac{1}{2}(w,_{ni} + w,_{nj}) \tag{4.24}$$

By using the above equations and assumptions, the explicit expressions of rotations β_x and β_y of a general quadrilateral element in terms of the nodal variables are obtained as:

$$\beta_x = \langle H^x(\xi,\eta)\rangle\{U_n\} \tag{4.25}$$

$$\beta_y = \langle H^y(\xi,\eta)\rangle\{U_n\} \tag{4.26}$$

where

$$H_1^x = \frac{3}{2}(a_5N_5 - a_8N_8) \tag{4.27}$$

$$H_2^x = (b_5N_5 + b_8N_8) \tag{4.28}$$

$$H_3^x = (N_1 - c_5N_5 + c_8N_8) \tag{4.29}$$

$$H_1^y = \frac{3}{2}(d_5N_5 - d_8N_8) \tag{4.30}$$

$$H_2^y = (-N_1 + e_5N_5 + e_8N_8) \tag{4.31}$$

$$H_3^y = -(b_5N_5 + b_8N_8) \tag{4.32}$$

Note that the functions H_4, H_5,..., H_{12} are obtained by replacing N_1 by N_2 and the indices 8 by 5 and 5 by 6 for the next three equations for H. Similarly, the next three terms are obtained by replacing N_1 by N_3, 8 by 6 and 5 by 7. Finally, for the last three equations, N_1 must be replaced by N_4, 8 by 7 and 5 by 8. The terms a_k, b_k, c_k, d_k, e_k are given below:

$$a_k = -\frac{X_{ij}}{l_{ij}^2} \quad b_k = \frac{3}{4}\left(\frac{X_{ij}Y_{ij}}{l_{ij}^2}\right) \quad c_k = \left(\frac{1}{4}X_{ij}^{\,2} - \frac{1}{2}Y_{ij}^{\,2}\right)\Big/ l_{ij}^2 \tag{4.33}$$

$$d_k = -\frac{Y_{ij}}{l_{ij}^2} \qquad e_k = \left(-\frac{1}{2}X_{ij}^{\,2} + \frac{1}{4}Y_{ij}^{\,2}\right)\Big/ l_{ij}^2 \tag{4.34}$$

Hence, the B matrix is

$$B = \begin{bmatrix} \langle H^x{}_{,x} \rangle \\ \langle H^y{}_{,y} \rangle \\ \langle H^x{}_{,y} + H^y{}_{,x} \rangle \end{bmatrix} \tag{4.35}$$

4.6 Thick Plate Element

First, a two-node Timoshenko beam in which the shear deformations are included is considered. Such an element is presented in Figure 4.5.

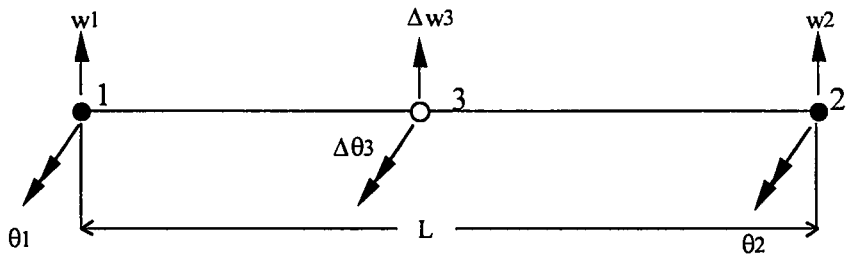


Figure 4.5 Displacements in a beam element

A linear variation is assumed along the beam for the geometry and the rotation field.

$$x = N_1(r)x_1 + N_2(r)x_2 \tag{4.36}$$

$$\theta = N_1(r)\theta_1 + N_2(r)\theta_2 \tag{4.37}$$

where N_1 and N_2 are linear interpolation functions. In order to construct the displacement and rotation interpolation free of shear locking, the displacement interpolation should be a polynomial of order one degree higher than the polynomial that interpolates the rotations. Hence, the displacement field must be quadratic.

$$w = N_1(r)w_1 + N_2(r)w_2 + N_3(r)\Delta w_3 \tag{4.38}$$

By using the above approximations, the discrete approximation for the curvature becomes

$$\kappa = \frac{d\theta}{dx} = \frac{1}{l}(\theta_2 - \theta_1) \tag{4.39}$$

and the shear strain is

$$\gamma = \frac{dw}{dx} - \theta = \frac{1}{l}(w_2 - w_1) - \frac{1}{2}(\theta_1 + \theta_2) + r\left[\frac{1}{2}(\theta_1 - \theta_2) - \frac{4}{l}\Delta w_3\right] \tag{4.40}$$

Note that if the relative nodal displacement Δw_3 is set to zero, then all fields are approximated by linear interpolation. In this case, the linearly varying

term in shear is directly related to the change of curvature. So, it is not possible to have a constant shear strain in bending behaviour. This inconsistency causes shear locking. This phenomena is avoided by constraining the terms in brackets in Equation 4.40 to zero. Hence, the displacement interpolation becomes:

$$w = N_1(r)w_1 + N_2(r)w_2 + N_3(r)\frac{1}{8}(\theta_1 - \theta_2) \tag{4.41}$$

The way the Timoshenko beam considerations are expanded to four-node quadrilateral element is as follows.

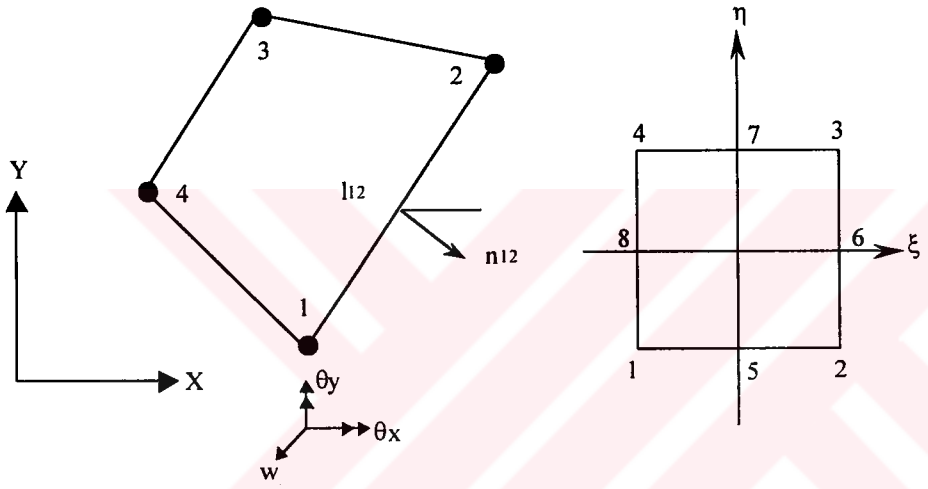


Figure 4.6 Typical quadrilateral element and reference coordinate system

The reference coordinate and the in-plane rotations are defined by linear interpolation functions.

$$x = \sum_{i=1}^{4} N_i^e(\xi,\eta)x_i \tag{4.42}$$

$$\begin{pmatrix}\theta_1\\ \theta_2\end{pmatrix}=\sum_e\sum_{i=1}^{4}N_i^e(\xi,\eta)\theta_i \tag{4.43}$$

Then, the transverse displacement interpolation is obtained by generalising the beam interpolation as

$$w^h=\sum_e\sum_{i=1}^{4}N_i^e(\xi,\eta)w_i-\sum_e\sum_{i=5}^{8}NS_i^e(\xi,\eta)\frac{l_{jk}}{8}n_{jk}(\theta_k-\theta_j) \tag{4.44}$$

where

$$NS_i(\xi,\eta)=\frac{1}{2}(1-\xi^2)(1+\eta_i\eta) \qquad \text{i= 5, 7} \tag{4.45}$$

$$NS_i(\xi,\eta)=\frac{1}{2}(1+\xi_i\xi)(1-\eta) \qquad \text{i= 6, 8} \tag{4.46}$$

By using the strain-displacement relationship defined in Equation 4.17 for Mindlin plate theory, the $\underline{B}$ matrix becomes

$$B_i^e=\begin{bmatrix}0 & N_{i,x_1}^e & 0\\ 0 & 0 & N_{i,x_2}^e\\ 0 & N_{i,x_2}^e & N_{i,x_1}^e\end{bmatrix} \tag{4.47}$$

and the shear strain interpolation is

$$G_i^e=\begin{bmatrix}N_{i,x_1}^e & -\frac{1}{8}(l_{ij}\cos\alpha_{ij}NS_{i,x_1}^e & N_i^e-\frac{1}{8}(l_{ij}\sin\alpha_{ij}NS_{l,x_1}^e\\ & -l_{ik}\cos\alpha_{ik}NS_{m,x_1}) & -l_{ik}\sin\alpha_{ik}NS_{m,x_1})\\ N_{i,x_2}^e & -N_i^e-\frac{1}{8}(l_{ij}\cos\alpha_{ij}NS_{l,x_2}^e & -\frac{1}{8}(l_{ij}\sin\alpha_{ij}NS_{l,x_2}^e\\ & -l_{ik}\cos\alpha_{ik}NS_{m,x_2}) & -l_{ik}\sin\alpha_{ik}NS_{m,x_2})\end{bmatrix} \tag{4.48}$$

where i = 1,2,3,4; m = i + 4; l = m – 4aint(1/i); k = mod(m,4) + 1; and j = 1 – 4. Finally, the stiffness matrix of the thick plate element can be calculated by

$$K^e = \int_\Omega B^{e^T} D_k B^e d\Omega + \int_\Omega G^{e^T} D_S G^e d\Omega \tag{4.49}$$

4.7 Membrane Element with Drilling Degrees of Freedom

The drilling degrees of freedom are desired in the finite element modeling of many engineering structures especially at the joints and intersections of different structural components. They become essential in the case of plate/shell macro elements for displacement interpolation along the boundaries.

In the derivation of this element, the drilling degrees of freedom are taken as independent entities by separating them from the displacement field over the element domain. An additional rotational field has to be defined over the element for the drilling degrees of freedom. An approach to this problem has been presented by Hughes and Brezzi[38] by separating the kinematic variables of displacement and rotation by introducing a symmetric (for displacements) and a skew-symmetric (for drilling rotation) fields in the following form

$$\Pi_\gamma(v,\omega) = \frac{1}{2}\int_\Omega symm\ (\nabla v).D_k.symm\ (\nabla v)d\Omega + \frac{1}{2}\int_\Omega \left|skew\ \nabla v - \omega\right|^2 d\Omega - \int_\Omega v.fd\Omega \tag{4.50}$$

Hence, the variational equation is

$$D\prod_y(u,\theta).(v,\omega) = \int_\Omega (symm\nabla v).D_k.(symm\nabla u)d\Omega - \int_\Omega v.fd\Omega + \int_\Omega (skew\nabla v - \omega)^T.(skew\nabla u - \theta)d\Omega \quad (4.51)$$

The above equation is taken as the basis for constructing the displacement type discrete formulation.

4.7.1 Finite Element Interpolation

The variation of the independent rotation field is assumed to be linear over each element. Therefore, the rotation field over an element can be expressed as

$$\theta^h = \sum_e \sum_{i=1}^{4} N_i^e(\xi,\eta)\theta_i \quad (4.52)$$

where Ni are the linear interpolation functions. Allman-type interpolation is assumed when describing the in-plane displacements. Allman derives the interpolation functions for an edge such that they relate the mid-point displacements along the edge with the nodal rotations at the two ends of the edge. For a typical edge shown in Figure 4.7,

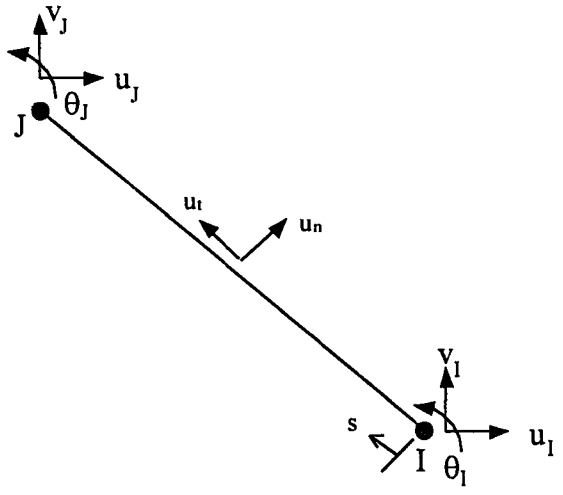


Figure 4.7 Assumed displacements for a typical edge

quadratic variation for the normal displacement and a linear variation for the tangential displacement is assumed. Let

$$u_n = a_1 + a_2 s + a_3 s^2 \tag{4.53}$$

$$u_t = a_4 + a_5 s \tag{4.54}$$

Note that there are four boundary conditions. At s = 0, $u_n = u_{nI}$ and $u_t = u_{tI}$ and at $s = l_{IJ}$, $u_n = u_{nJ}$ and $u_t = u_{tJ}$. However, in Equations 4.53 and 4.54 there are five unknowns and an additional boundary condition is needed.. The fifth boundary condition for any edge is taken as the difference between the derivatives of the transverse edge displacement at end points.

$$\left.\frac{\partial u_n}{\partial s}\right|_{s=l} - \left.\frac{\partial u_n}{\partial s}\right|_{s=0} = -\theta_J + \theta_I \tag{4.55}$$

After doing necessary calculations, the unknowns are found to be

$$a_1 = u_{nI} \qquad a_2 = \frac{1}{l_{IJ}}(u_{nJ} - u_{nI}) + \frac{1}{2}(\theta_J - \theta_I)$$

$$a_3 = \frac{1}{2l_{IJ}}(\theta_J - \theta_I) \qquad a_4 = u_{tI} \qquad a_5 = \frac{1}{l_{IJ}}(u_{tJ} - u_{tI}) \tag{4.56}$$

Hence, the boundary displacements along edge any IJ will become

$$u_n = (1 - \frac{s}{l_{IJ}})u_{nI} + (\frac{s}{l_{IJ}})u_{nJ} + \frac{1}{2}s(1 - \frac{s}{l_{IJ}})(\theta_J - \theta_I) \tag{4.57}$$

$$u_t = (1 - \frac{s}{l_{IJ}})u_{tI} + (\frac{s}{l_{IJ}})u_{tJ} \tag{4.58}$$

The interpolation can be simplified by constraining the tangential displacement to zero. By this way the mid-side displacement will be as follows:

$$u_{nIJ} = \frac{1}{2}u_{nI} + \frac{1}{2}u_{nJ} + \frac{l_{IJ}}{8}(\theta_J - \theta_I) \tag{4.59}$$

For the relative displacement, $u_{nI} = u_{nJ} = 0$. Then, the final mid-side transverse displacement becomes

$$u_{nIJ} = \frac{l_{IJ}}{8}(\theta_J - \theta_I) \tag{4.60}$$

The assumed in-plane displacement approximation for a quadrilateral element is

$$\begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \sum_e \sum_{i=1}^{4} N_i^e(\xi,\eta)u_i + \sum_e \sum_{i=5}^{8} NS_i^e(\xi,\eta)\frac{l_{jk}}{8}(\theta_k - \theta_j)n_{jk} \tag{4.61}$$

The skew-symmetric stress field is chosen constant over the element, i.e

$$skew\,\tau^h = \sum_e \tau_o^e \tag{4.62}$$

The strain-displacement relation for the symmetric part of the displacement field are defined as, in matrix notation

$$symm\nabla u^e = B_i^e u_i + G_i^e \theta_i \tag{4.63}$$

where

$$B_i^e = \begin{bmatrix} N_{i,x_1}^e & 0 \\ 0 & N_{i,x_2}^e \\ N_{i,x_2}^e & N_{i,x_1}^e \end{bmatrix} \qquad \text{i} = 1,2,3,4 \tag{4.64}$$

and

$$G_i^e = \frac{1}{8}\begin{bmatrix} l_{ij}\cos\alpha_{ij} NS_{l,x_1}^e - l_{ik}\cos\alpha_{ik} NS_{m,x_1}^e \\ l_{ij}\sin\alpha_{ij} NS_{l,x_2}^e - l_{ik}\sin\alpha_{ik} NS_{m,x_2}^e \\ (l_{ij}\cos\alpha_{ij} NS_{l,x_2}^e - l_{ik}\cos\alpha_{ik} NS_{m,x_2}^e) + (l_{ij}\sin\alpha_{ij} NS_{l,x_1}^e - l_{ik}\sin\alpha_{ik} NS_{m,x_1}^e) \end{bmatrix} \tag{4.65}$$

The first term in the discrete formulation can be calculated as

$$K^e = \int_{\Omega^e} \begin{bmatrix} B^e & G^e \end{bmatrix}^T \{D_k\} \begin{bmatrix} B^e & G^e \end{bmatrix} d\Omega \tag{4.66}$$

Furthermore, the skew-symmetric part is denoted as

$$skew\nabla u^e - \theta^e = b_i^e u_i + g_i^e \theta_i \tag{4.67}$$

where

$$b_i^e = \left\langle -\frac{1}{2} N_{i,x_2}^e \quad \frac{1}{2} N_{i,x_1}^e \right\rangle \tag{4.68}$$

$$g_i^e = \begin{bmatrix} -\frac{1}{16}(l_{ij}\cos\alpha_{ij} NS^e_{l,x_2} - l_{ik}\cos\alpha_{ik} NS^e_{m,x_2}) \\ +\frac{1}{16}(l_{ij}\sin\alpha_{ij} NS^e_{l,x_1} - l_{ik}\sin\alpha_{ik} NS^e_{m,x_1}) - N_i^e \end{bmatrix} \tag{4.69}$$

where i = 1,2,3,4; m = i + 4; l = m – 4aint(1/i); k = mod(m,4) + 1; and j = l – 4.

The second term is obtained by

$$P^e = \gamma \int_{\Omega^e} \begin{Bmatrix} b^e \\ g^e \end{Bmatrix} \begin{bmatrix} b^e & g^e \end{bmatrix} d\Omega \tag{4.70}$$

The parameter γ depends on the problem. For isotropic elasticity it could be taken as the shear modulus value. The complete stiffness for an element will then be equal to

$$\left[K^e + P^e\right] \cdot \underline{u} = \underline{f} \tag{4.71}$$

4.8 Joist Beam Element

The joist beam element is basically the classical Hermitian beam element. However, it is defined with reference to its neutral axis shifted parallel to itself. This reference axis is the mid-plane of the shell element to which the joist is attached. Since the joist is attached to a shell element, its degrees of freedom are related to the shell element degrees of freedom.

4.8.1 Neutral Axis Transformation

For a typical stiffened element as shown in Figure 4.8

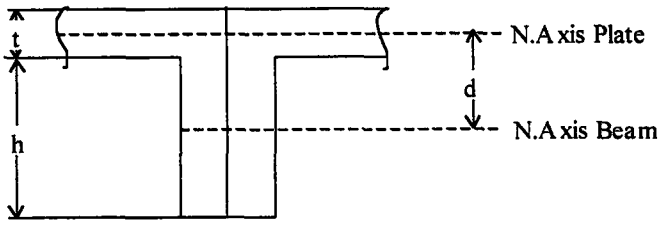


Figure 4.8 Neutral axis transformation for a joist beam element

the neutral axis of the beam must be shifted by an amount d where

$$d = (h + t)/2 \tag{4.72}$$

h is the height of the beam and t is the thickness of the plate. The transformation is done by the application of some constraint equations. For this purpose, the relations between the beam degrees of freedom and the transformed degrees of freedom are needed. The beam element degrees of freedom are defined in Figure 4.9.

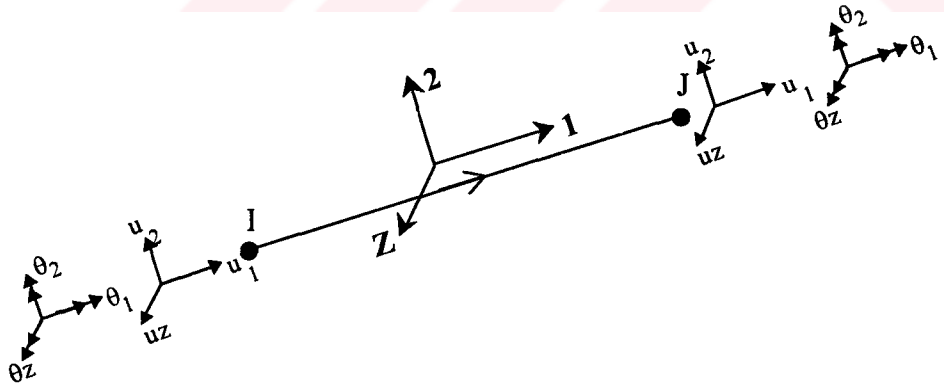


Figure 4.9 Nodal displacements for a typical beam element

The transformed displacements will be as follows:

$$u_1' = u_1 + d.\theta_2 \qquad \theta_1' = \theta_1$$
$$u_2' = u_2 - d.\theta_1 \qquad \theta_2' = \theta_2 \qquad (4.73)$$
$$u_3' = u_3 \qquad \theta_3' = \theta_3$$

By using the above relationships and using the Equation 2.21, the matrix $\underline{C}_{cr}$ for neutral axis transformation becomes

$$C_{cr} = \begin{bmatrix} \underline{A} & 0 \\ 0 & \underline{A} \end{bmatrix} \qquad (4.74)$$

where $\underline{A}$ is as given below.

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & -d & 0 \\ & 1 & 0 & d & 0 & 0 \\ & & 1 & 0 & 0 & 0 \\ & & & 1 & 0 & 0 \\ & \underline{0} & & & 1 & 0 \\ & & & & & 1 \end{bmatrix} \qquad (4.75)$$

4.8.2 Nodal Displacement Transformation

The beam degrees of freedom are transferred to the nodes of the shell element by using the same principles as in the macro element. For a stiffened element shown in Figure 4.10,

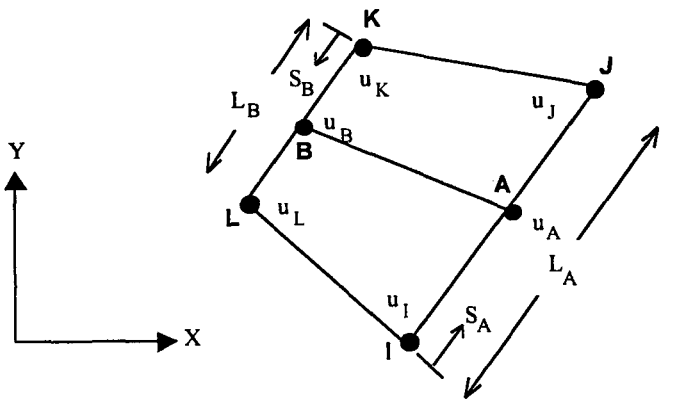


Figure 4.10 Joist beam displacements in terms of shell element displacements

the degrees of freedom at point A can be expressed in terms of the degrees of freedom at the element nodes I and J. Similarly, the degrees of freedom at point B can be expressed in terms of the degrees of freedom at the element nodes K and L. Hence, the matrices $\underline{D}_r$ and $\underline{D}_c$ takes on the following form.

$$D_r = \begin{bmatrix} u_I \\ u_J \\ u_K \\ u_L \end{bmatrix} \qquad D_c = \begin{Bmatrix} u_A \\ u_B \end{Bmatrix} \tag{4.76}$$

u represents the nodal degrees of freedom. Note that the matrix $\underline{D}_r$ changes as the position of the beam changes. Applying the same procedure as in the macro element, the matrix $\underline{C}_{cr}$ is obtained as

$$C_{cr}{}^2 = \begin{bmatrix} \underline{ZI}_1(\xi_A) & 0 & \underline{ZJ}_1(\xi_A) & 0 & 0 & 0 & 0 & 0 \\ 0 & \underline{ZI}_2(\xi_A) & 0 & \underline{ZJ}_2(\xi_A) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \underline{ZI}_1(\xi_B) & 0 & \underline{ZJ}_1(\xi_B) & 0 \\ 0 & 0 & 0 & 0 & 0 & \underline{ZI}_2(\xi_B) & 0 & \underline{ZJ}_2(\xi_B) \end{bmatrix} \tag{4.77}$$

where $\xi_A = S_A / L_A$ and $\xi_B = S_B / L_B$

The matrices ZI and ZJ are the same matrices defined previously in Equations 2.50 to 2.53.

4.8.3 Constraint Matrices for a Joist Beam Element

Now, recall the Equation 2.31, the constrained stiffness matrix has the form

$$\underline{K}_{con} = (\underline{K}_{rr} + \underline{K}_{rc}.\underline{C}_{cr} + \underline{C}_{cr}^{T}.\underline{K}_{cr} + \underline{C}_{cr}^{T}.\underline{K}_{cc}.\underline{C}_{cr}).\underline{D}_{r} \qquad (2.31)$$

By using the above equation, first, the neutral axis transformation is performed. The result is

$$\underline{K}_{con} = (\underline{K}_{rr} + \underline{K}_{rc}.\underline{C}_{cr}^{1} + \underline{C}_{cr}^{1T}.\underline{K}_{cr} + \underline{C}_{cr}^{1T}.\underline{K}_{cc}.\underline{C}_{cr}^{1}).\underline{D}_{r} \qquad (4.78)$$

Then, the nodal displacement transformation is imposed. The constrained stiffness matrix becomes

$$\underline{K}_{con} = (\underline{K}_{rr} + \underline{K}_{rc}.\underline{C}_{cr}^{1}\,\underline{C}_{cr}^{2} + \underline{C}_{cr}^{2T}\underline{C}_{cr}^{1T}.\underline{K}_{cr} + \underline{C}_{cr}^{2T}\underline{C}_{cr}^{1T}.\underline{K}_{cc}.\underline{C}_{cr}^{1}\,\underline{C}_{cr}^{2}).\underline{D}_{r} \qquad (4.79)$$

Defining the matrix C_{cr}^{*} as

$$\underline{C}_{cr}^{*} = \underline{C}_{cr}^{1} \cdot \underline{C}_{cr}^{2} \qquad (4.80)$$

simplifies the Equation 4.79 as

$$\underline{K}_{con} = (\underline{K}_{rr} + \underline{K}_{rc}.\underline{C}_{cr}^{*} + \underline{C}_{cr}^{*T}.\underline{K}_{cr} + \underline{C}_{cr}^{*T}.\underline{K}_{cc}.\underline{C}_{cr}^{*}).\underline{D}_{r} \qquad (4.81)$$

The matrix $\underline{C}_{cr}^{*}$ has the following form:

$$C_{cr}^{*} = \begin{bmatrix} \underline{ZI}_1^*(\xi_A) & \underline{ZI}_3^*(\xi_A) & \underline{ZJ}_1^*(\xi_A) & \underline{ZJ}_3^*(\xi_A) & 0 & 0 & 0 & 0 \\ 0 & \underline{ZI}_2^*(\xi_A) & 0 & \underline{ZJ}_2^*(\xi_A) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \underline{ZI}_1^*(\xi_B) & \underline{ZI}_3^*(\xi_B) & \underline{ZJ}_1^*(\xi_B) & \underline{ZJ}_3^*(\xi_B) \\ 0 & 0 & 0 & 0 & 0 & \underline{ZI}_2^*(\xi_B) & 0 & \underline{ZJ}_2^*(\xi_B) \end{bmatrix}$$

(4.82)

where $\underline{ZI}_1^*$, $\underline{ZI}_2^*$, $\underline{ZJ}_1^*$, $\underline{ZJ}_2^*$ are defined in Equations 2.50 to 2.53, respectively. $\underline{ZI}_3^*$ and $\underline{ZJ}_3^*$ matrices are given below.

$$ZI_3^* = \begin{bmatrix} -d.s.dN_3(\xi) & -d.(c^2 N_1(\xi) + s^2 dN_4(\xi)) & -d(cs.N_1(\xi) - cs.dN_4(\xi)) \\ d.N_3(\xi) & d.s.N_4(\xi) & -d.c.N_4(\xi) \\ 0 & 0 & 0 \end{bmatrix}$$

(4.83)

$$ZJ_3^* = \begin{bmatrix} -d.s.dN_5(\xi) & -d.(c^2 N_2(\xi) + s^2 dN_6(\xi)) & -d.(cs.N_2(\xi) - cs.dN_6(\xi)) \\ d.N_5(\xi) & d.s.N_6(\xi) & -d.c.N_6(\xi) \\ 0 & 0 & 0 \end{bmatrix}$$

(4.84)

CHAPTER 5

CASE STUDIES

5.1 Introduction

The accuracy and efficiency of the macro element are tested on several structures. The kinematic and the static responses of the macro elements are compared with those obtained from their full finite element analysis. Finite element program SAP90 [54] is used for this purpose.

As a first case study, a square plate which is simply supported at the corner points is selected. The deflected shape of this structure has a double curvature. First, the effect of mesh density, the effect of the number of coupling nodes and the effect of macro element density are tested by comparing the response predictions with finite element results. Afterwards, the nodal force response inside the plate was controlled. The second test structure is a rectangular plate which is clamped along three sides and hinge supported along the other. The statical response is checked both along the boundaries and inside the plate to see whether the stress resultants are suitable to be used in design. The next study is carried out on a cantilever plate. The effect of the number of coupling nodes used in the definition of the macro element and the macro element density in the finite element mesh of a given structure on the accuracy of the predictive capability of the element are tested. Finally, a cantilever plate structure having a square hole in the middle is studied using macro elements. The above properties again checked for this structure.

5.2 Square Plate Simply-Supported at the Corners

A square plate having an edge length of 6m. is analyzed under a uniform distributed load of $1t/m^2$. The geometric and material properties of the plate are given in Figure 5.1.

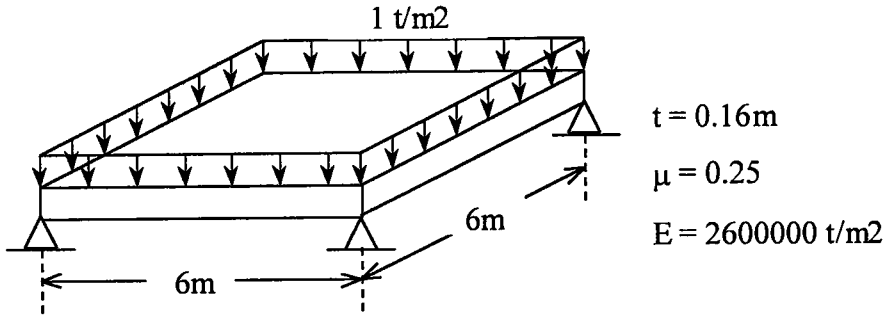


Figure 5.1 Simply supported square plate

First, the effect of the finite element mesh density used in the macro element definition on the macro element behavior is tested. For this purpose the results of a 4-node macro element having 4x4, 8x8, 16x16, 32x32 mesh densities, as shown in Figure 5.2, are compared. The displacement response (U_z, θ_x, θ_y) along Y = 1.5m and at Y = 3m are used. The results are given in Figures 5.3 -5.4.

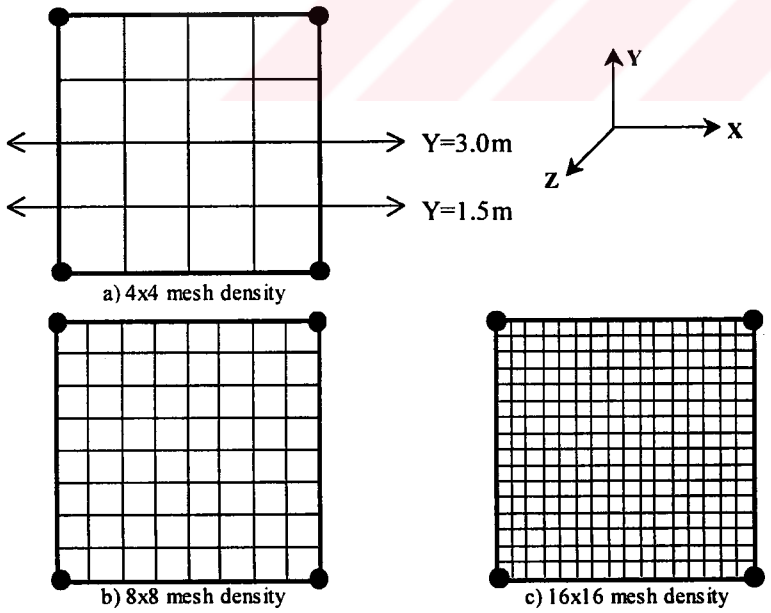


Figure 5.2 Macro elements having different mesh densities

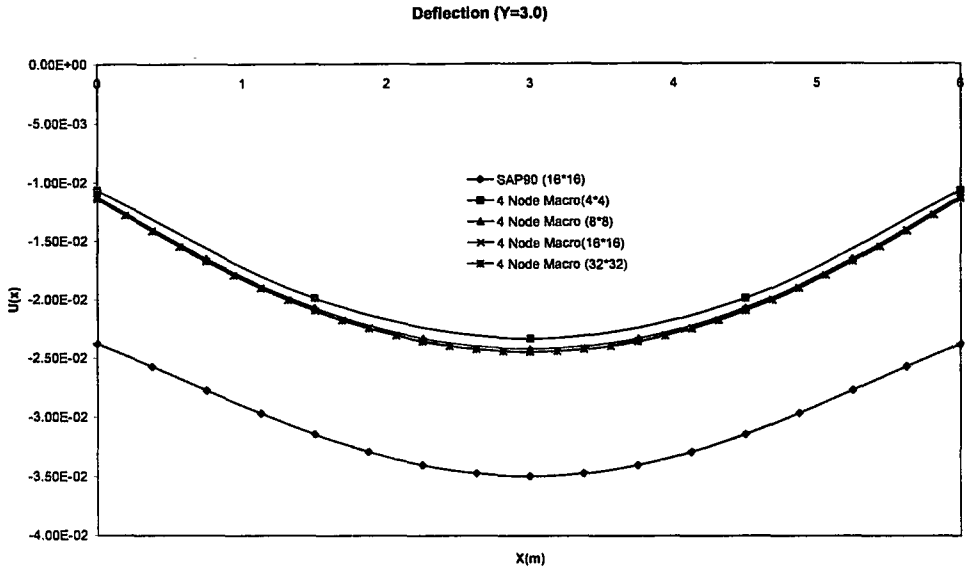


Figure 5.3 Deflection along Y=3m

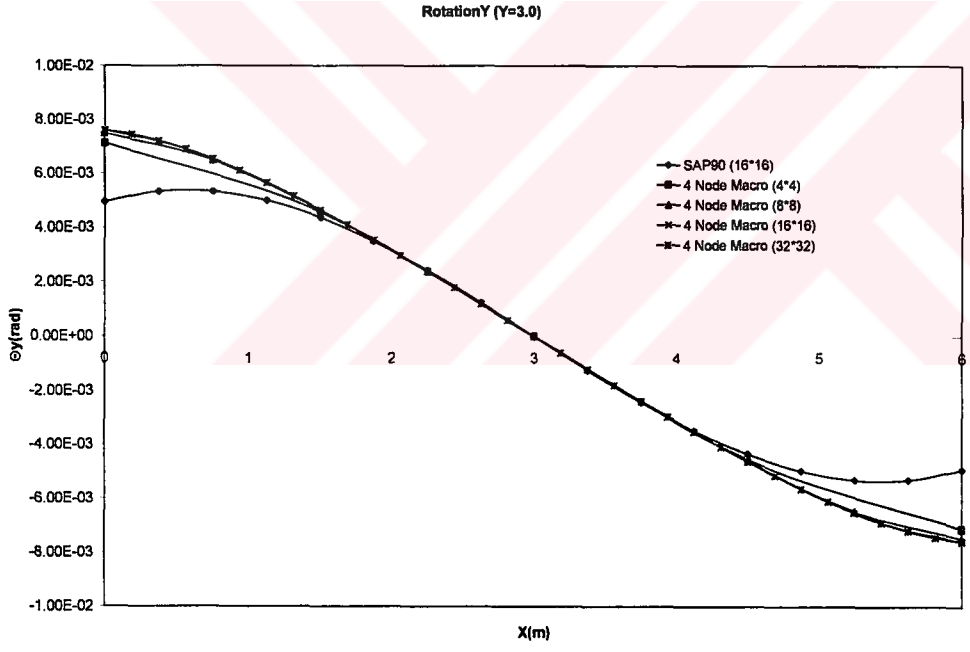


Figure 5.4 Rotation about Y-axis along Y=3.0m

Another test is performed to observe the influence of finite element mesh density on the behavior of the macro element. This time 8-noded macro elements are used with 4x4, 8x8, and 16x16 mesh densities. The results are

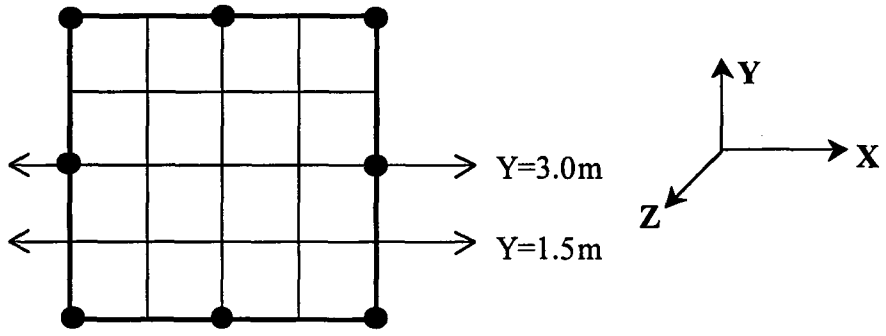


Figure 5.5 8-Node macro element

compared with the finite element analysis results using a 16x16 mesh. The results are given in Figures 5.6-5.8, below.

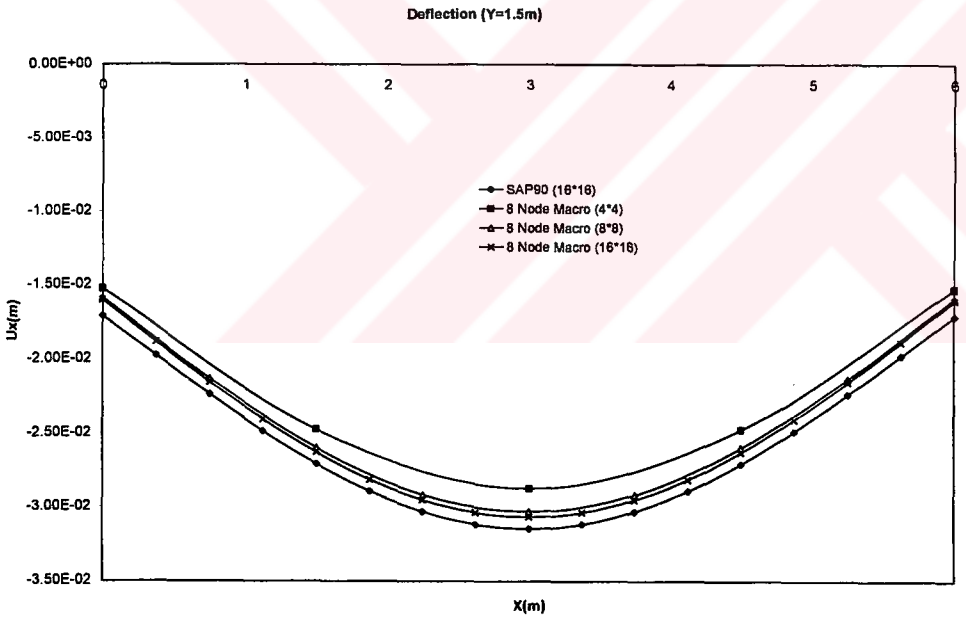


Figure 5.6 Deflection along Y=1.5m

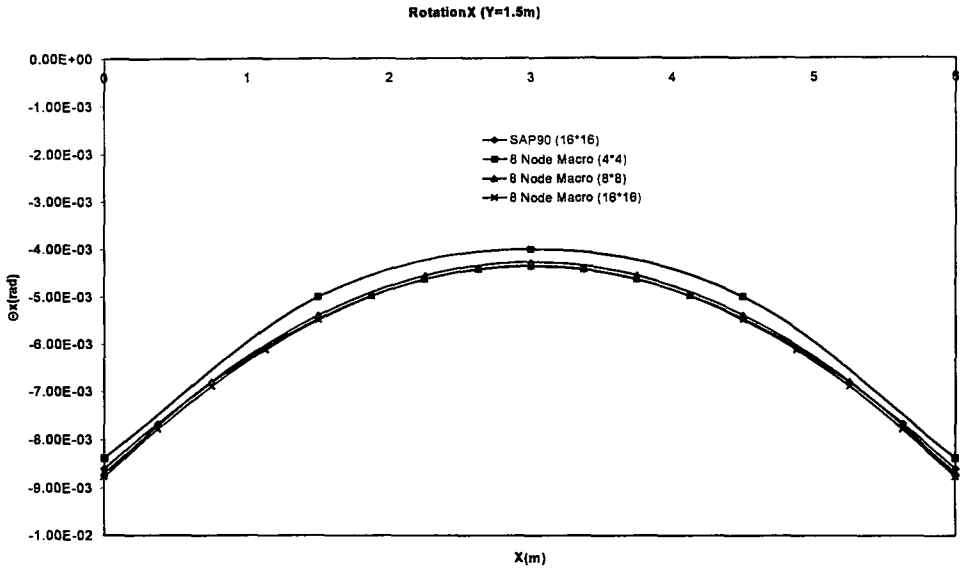


Figure 5.7 Rotation about X-axis along Y=1.5m

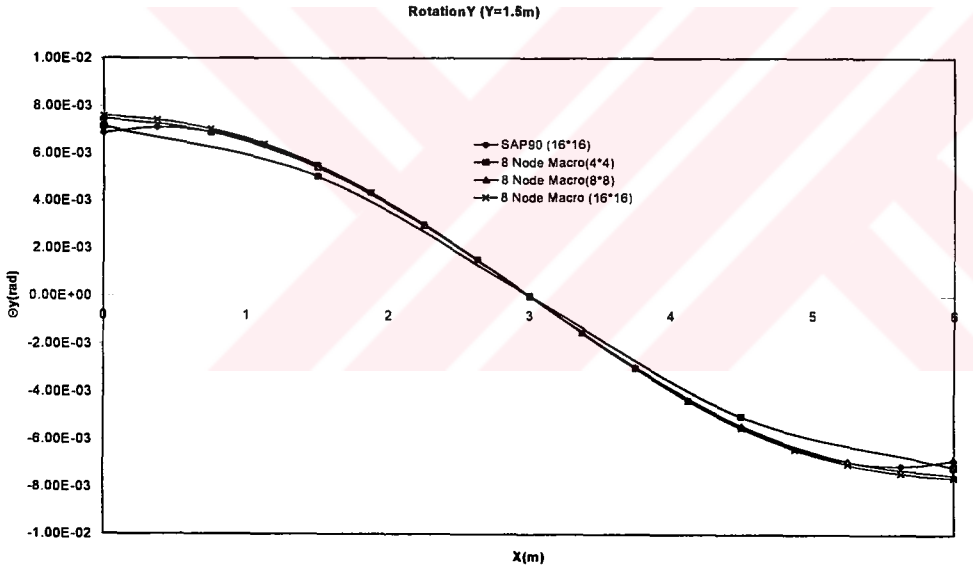


Figure 5.8 Rotation about Y-axis along Y=1.5m

In order to see the effect of the number of coupling nodes, the problem is solved by using 4-node macro element, 8-node macro element, and 12-node macro element, as shown in Figure 5.9. For the 4-node macro element and 8-node macro element the mesh density is 8x8, for the 12-node macro element it is 12x12. The displacement response is compared with that of a 16x16 mesh regular finite element solution. The results are presented in Figures 5.10-5.14

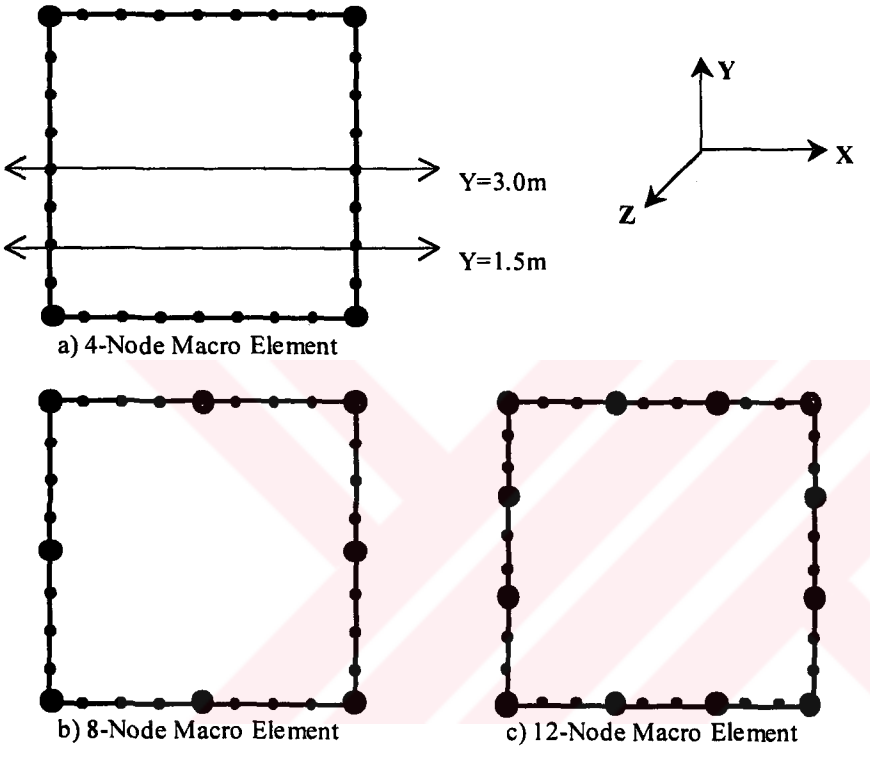


Figure 5.9 Macro elements with different number of coupling nodes

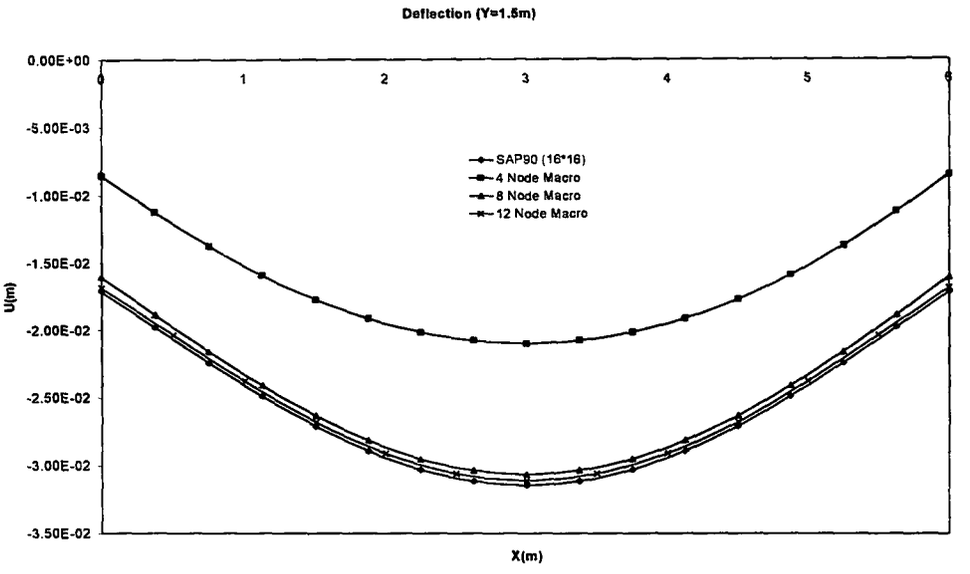


Figure 5.10 Deflection along Y=1.5m

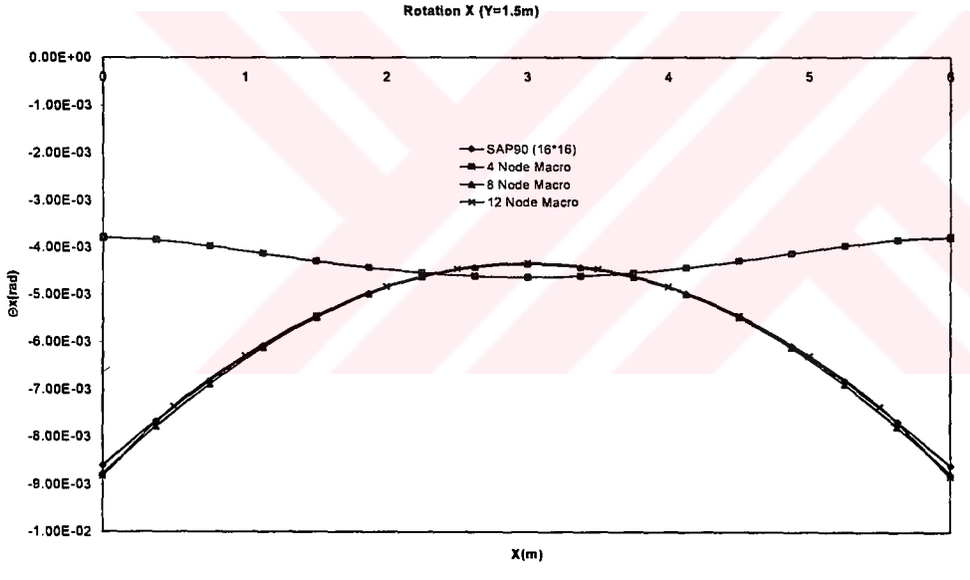


Figure 5.11 Rotation about X-axis along Y=1.5m

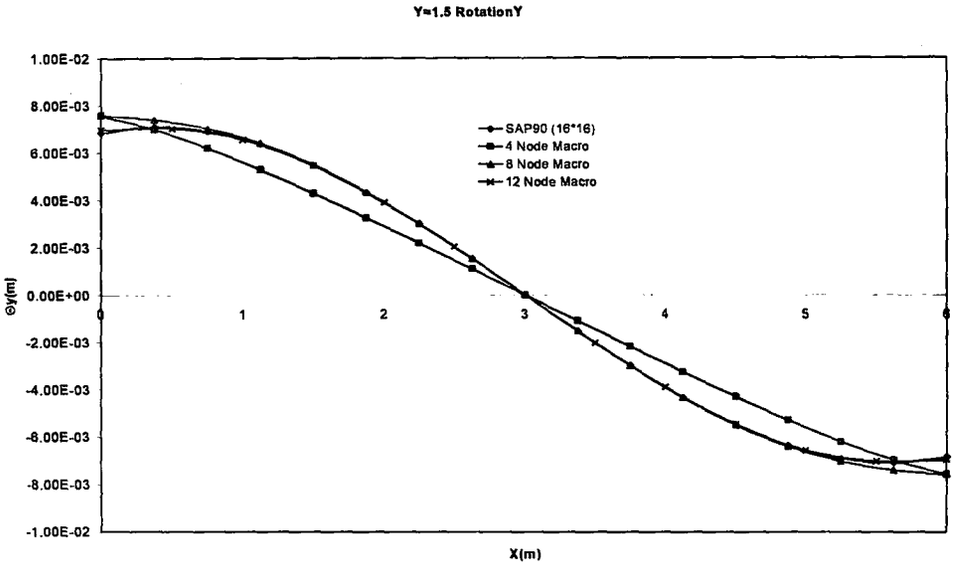


Figure 5.12 Rotation about Y-axis along Y=1.5m

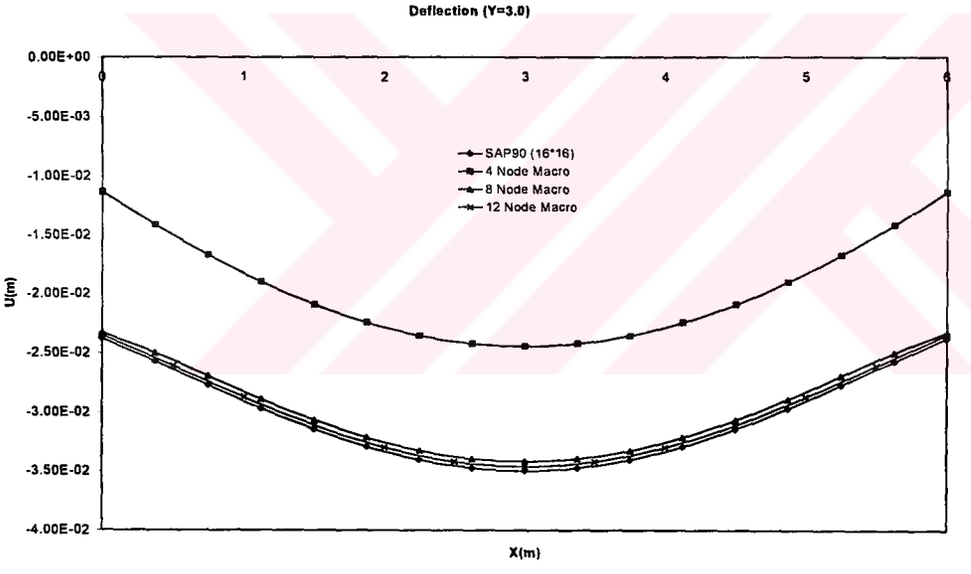


Figure 5.13 Deflection along Y=3.0m

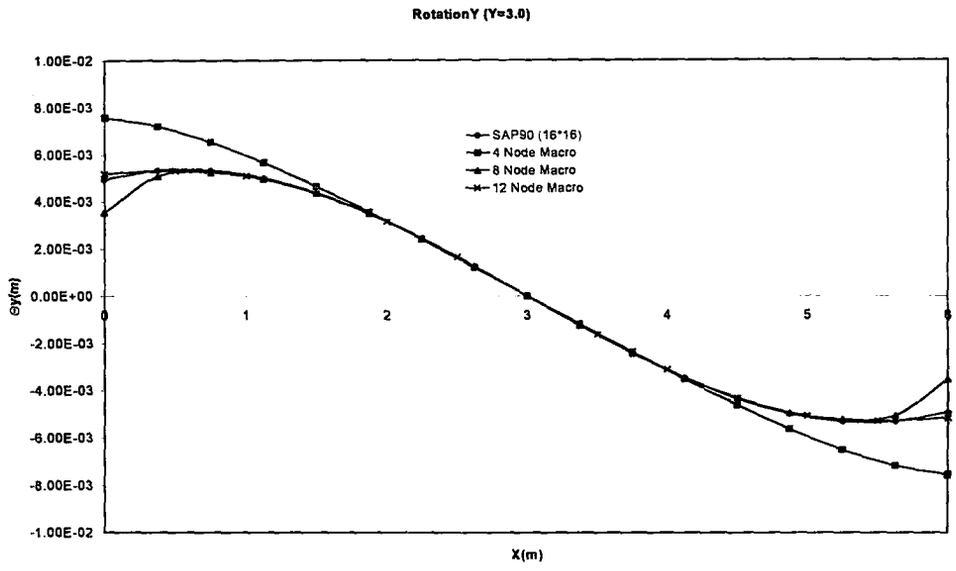
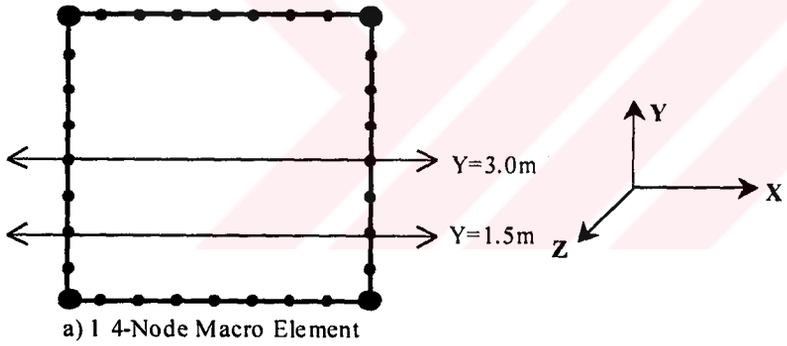
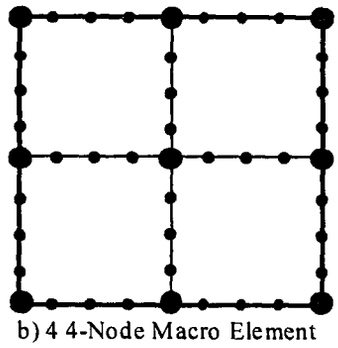


Figure 5.14 Rotation about Y-axis along Y=3.0m

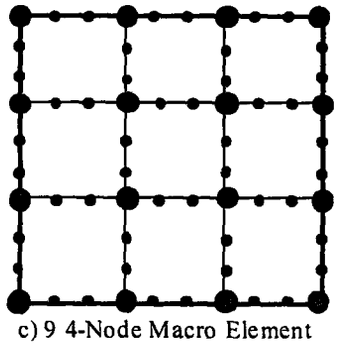
Next, the effect of macro element density is tested. This time, the same problem is solved by using a single macro element, 4 macro elements and 9 macro elements, as shown in Figure 5.15. Again, the displacement response is compared with that of a 16x16 mesh finite element solution. The results are shown in Figures 5.16-5.22



a) 1 4-Node Macro Element



b) 4 4-Node Macro Element



c) 9 4-Node Macro Element

Figure 5.15 Different number of macro elements

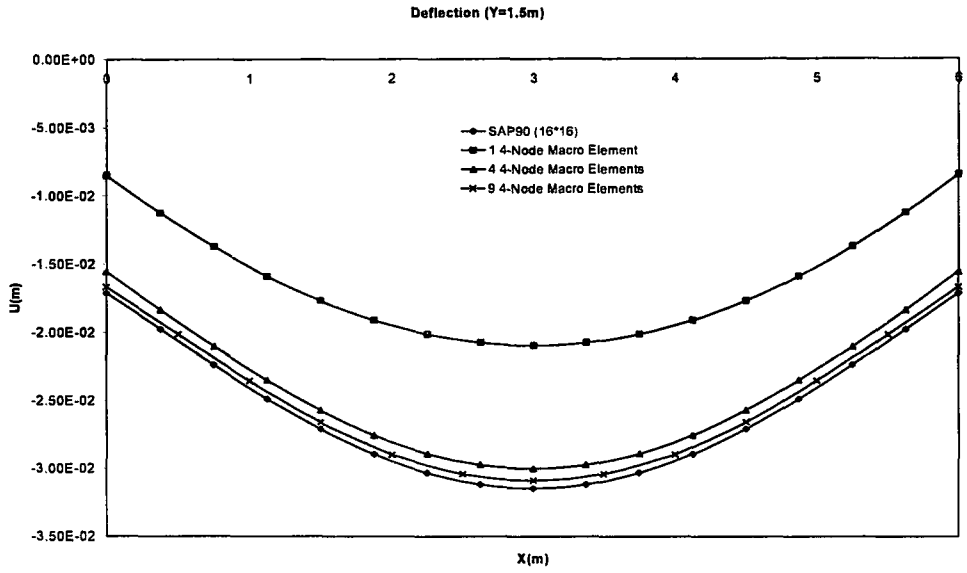


Figure 5.16 Deflection along Y=1.5m

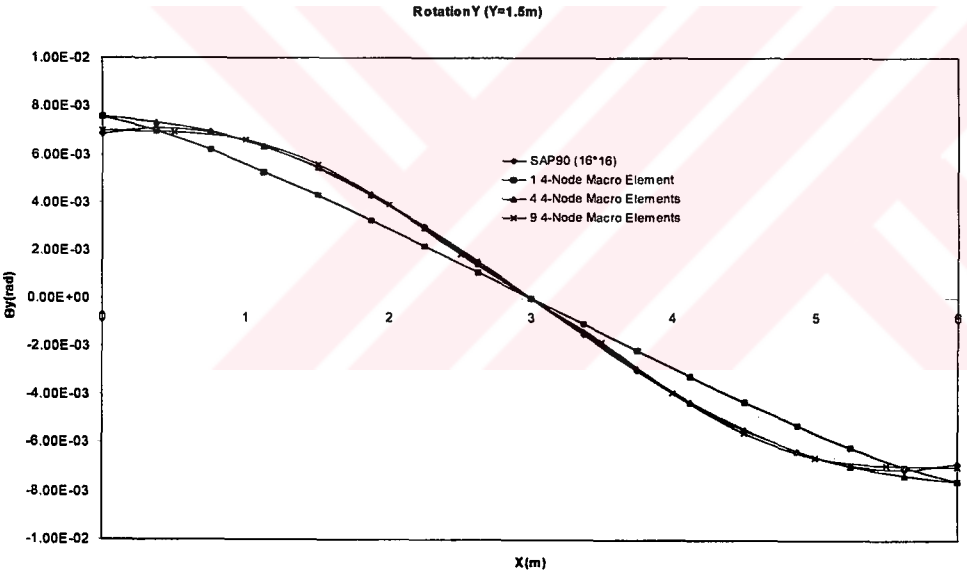


Figure 5.17 Rotation about Y-axis along Y=1.5m

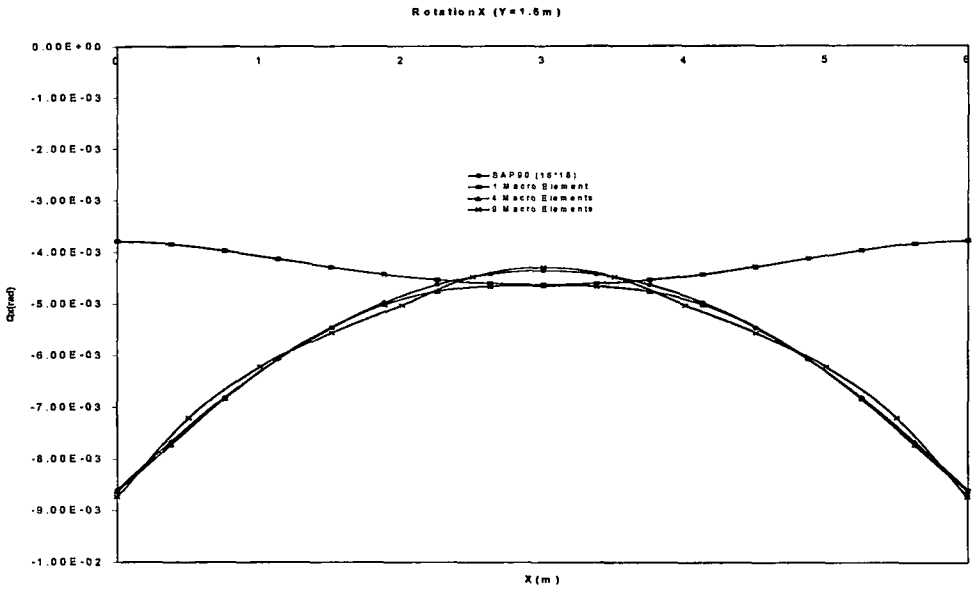


Figure 5.18 Rotation about X-axis along Y=1.5m

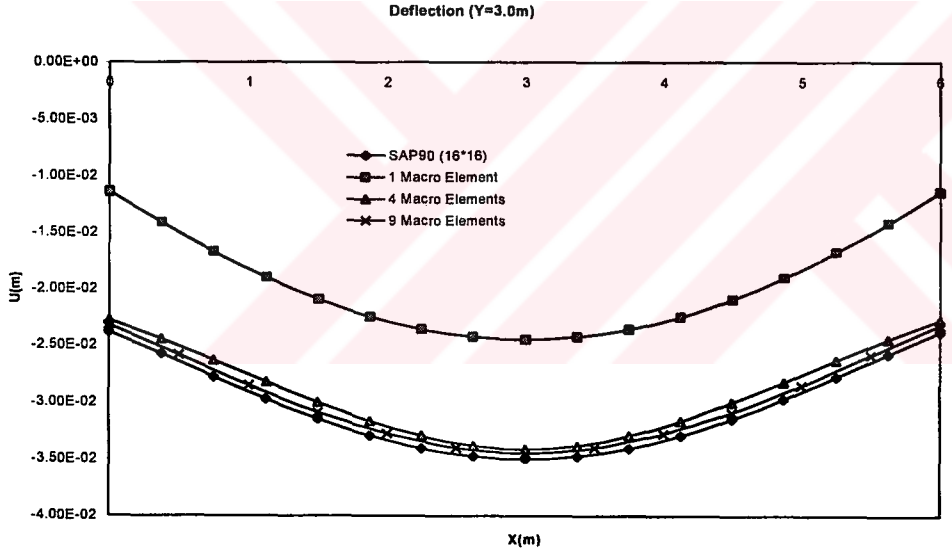


Figure 5.19 Deflection along Y=3.0m

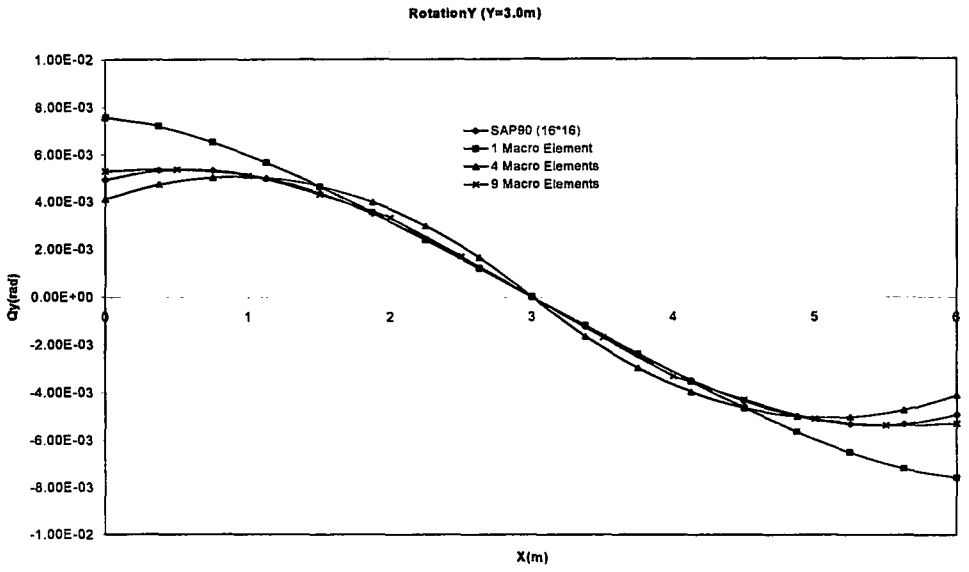


Figure 5.20 Rotation about Y-axis along Y=3.0m

Finally, the results of a regular finite element solution with a mesh density of 16x16 elements, a single 8-node macro element solution and the solution using 9 macro elements are compared. The displacements at Y = 1.5m and the nodal force response at the boundary X = 0.0 are presented below.

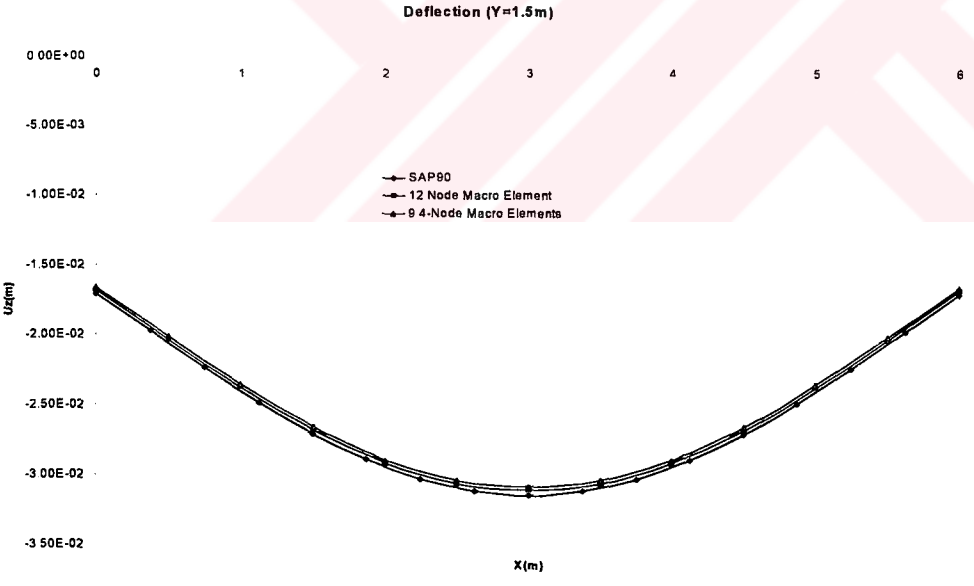


Figure 5.21 Deflection along Y=1.5m

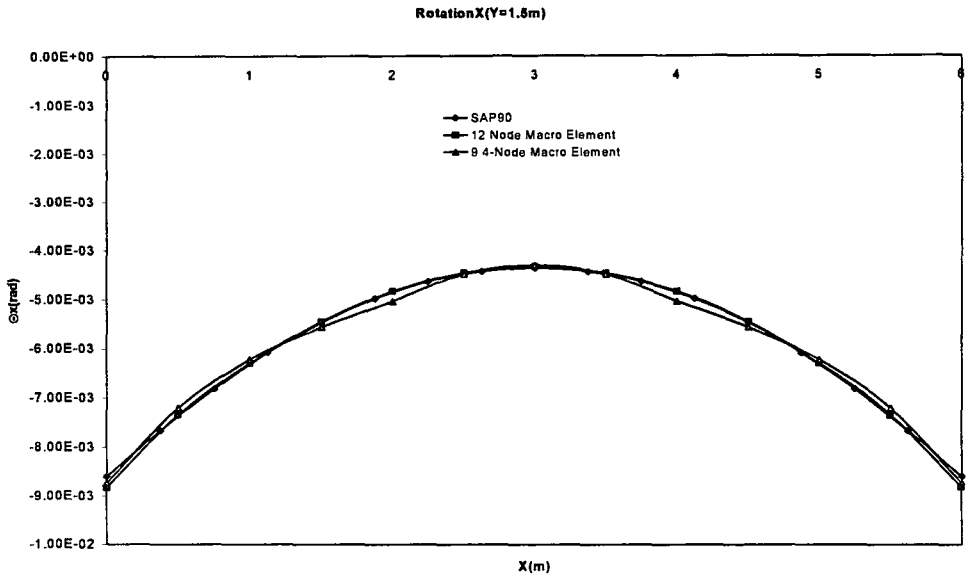


Figure 5.22 Rotation about X-axis along Y=1.5m

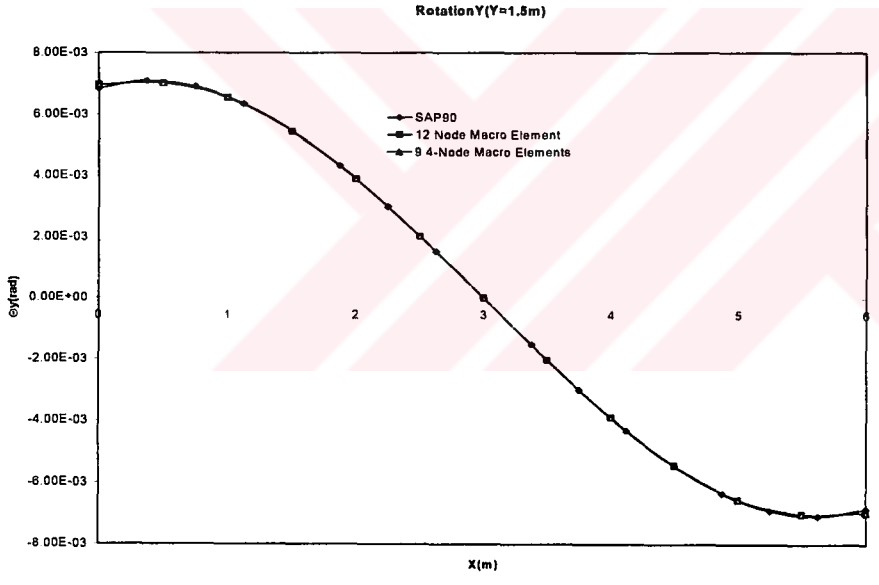


Figure 5.23 Rotation about Y-axis along Y=1.5m

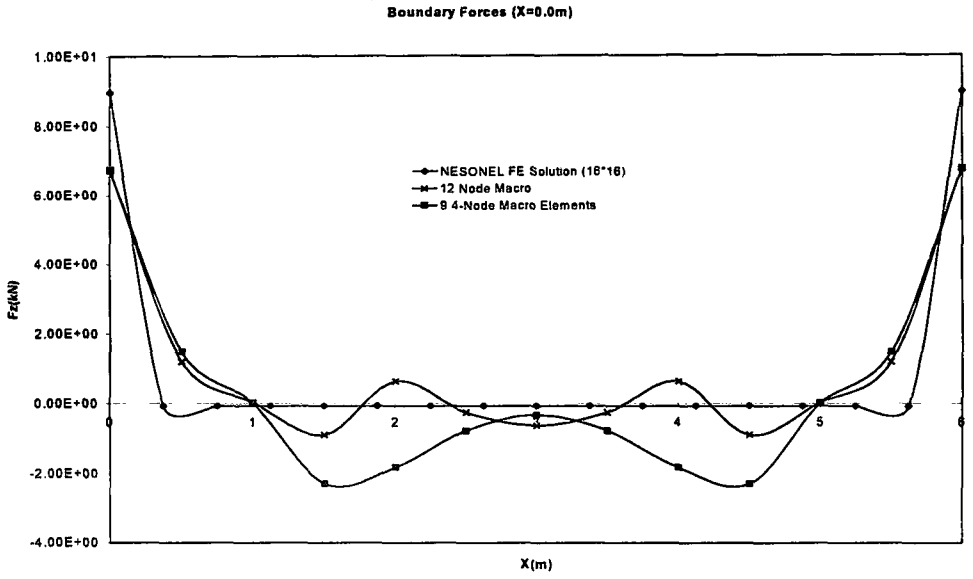


Figure 5.24 Force in Z-direction along X=0.0m

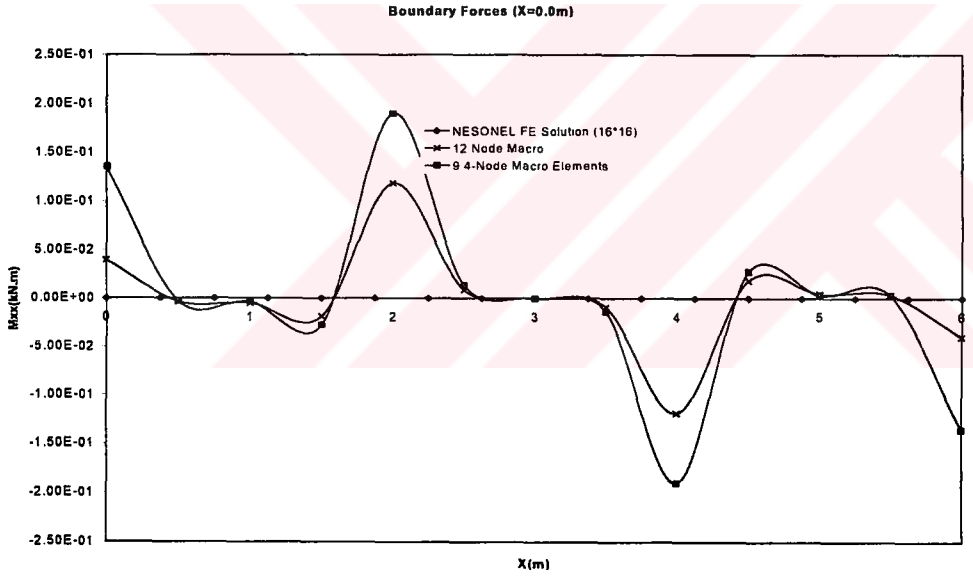


Figure 5.25 Moment about X-axis along X=0.0m

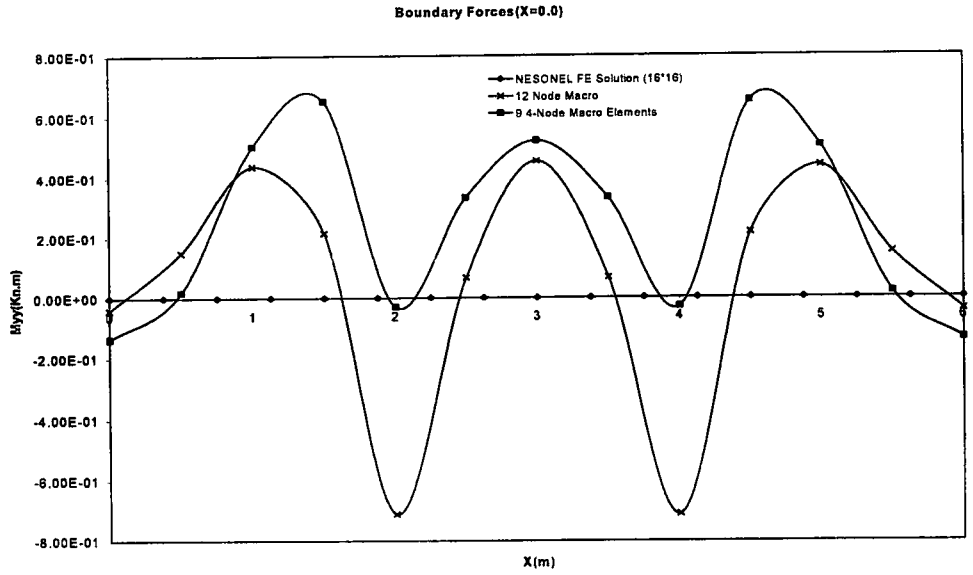


Figure 5.26 Moment about Y-axis along X=0.0m

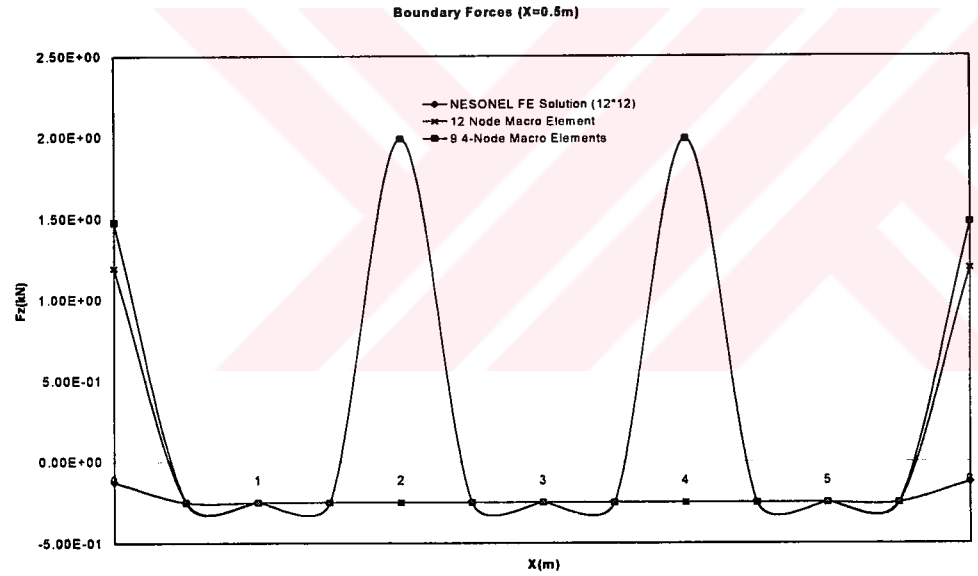


Figure 5.27 Force in Z-direction along X=0.5m

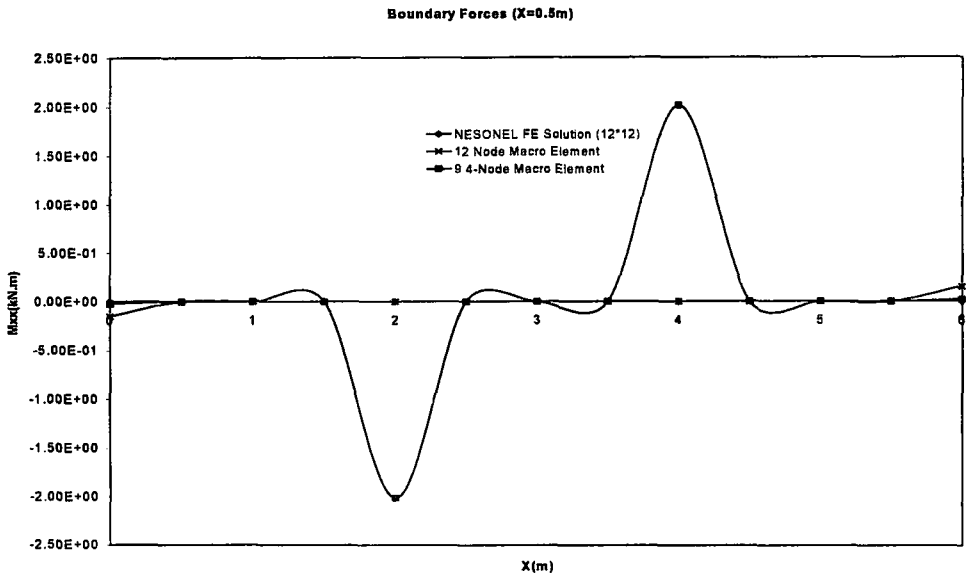


Figure 5.28 Moment about X-axis along X=0.5m

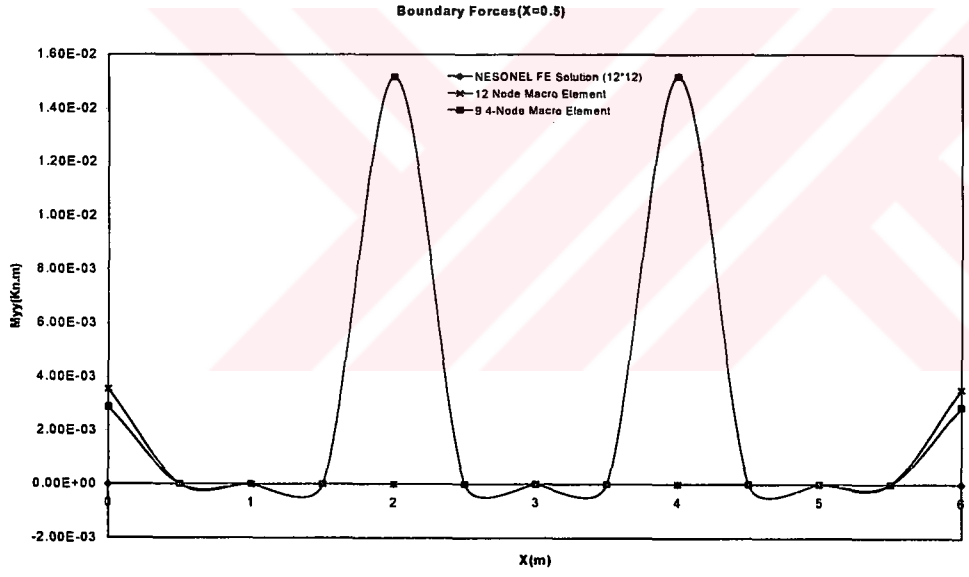


Figure 5.29 Moment about Y-axis along X=0.5m

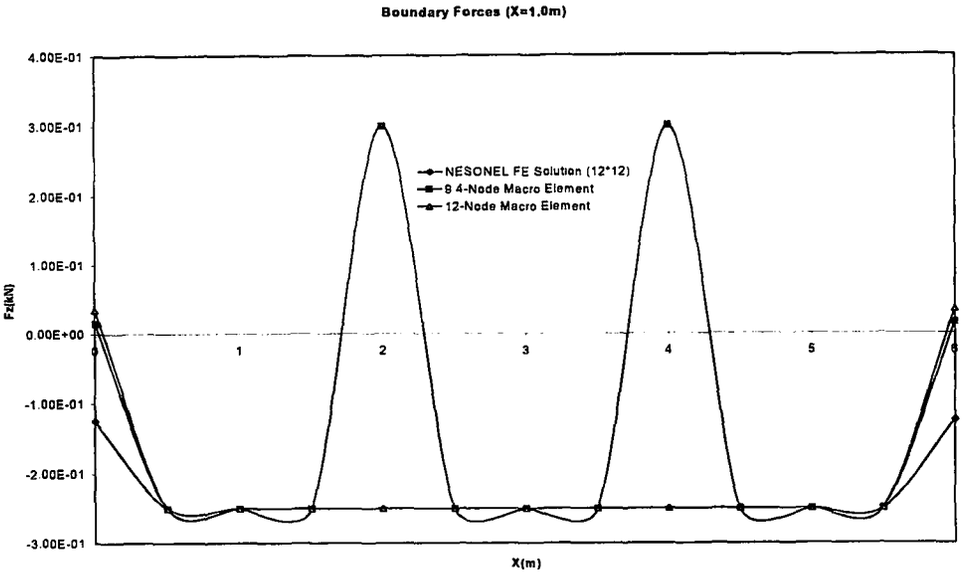


Figure 5.30 Force in Z-direction along X=1.0m

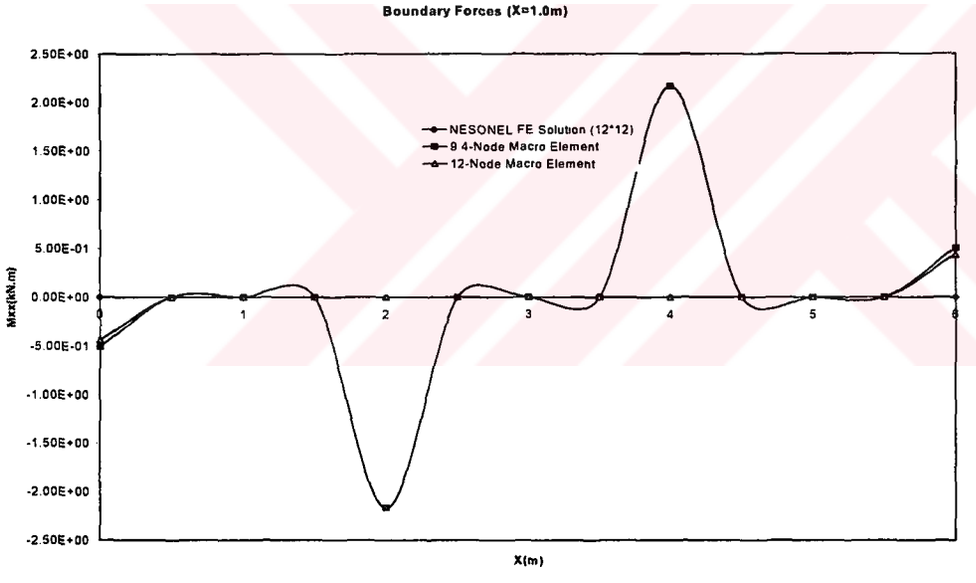


Figure 5.31 Moment about X-axis along X=1.0m

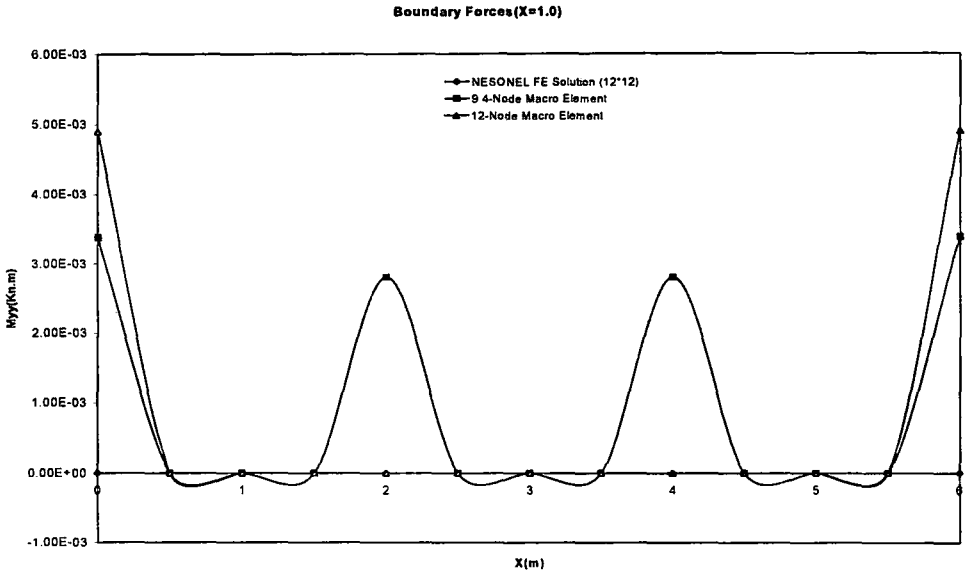


Figure 5.32 Moment about Y-axis along X=1.0m

5.3 Square Plate with Edge Boundary Conditions

The above square plate is again analyzed by changing the boundary conditions. The three edges are assumed to be fixed supported and the other edge is simply supported. The material and the geometric properties of the plate structure are given in Figure 5.33

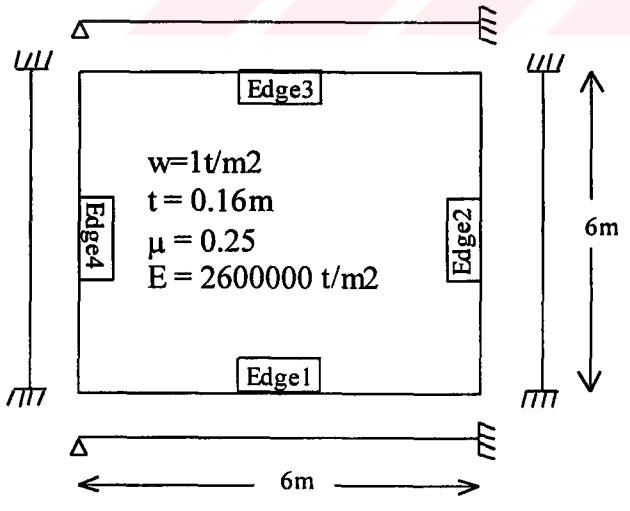


Figure 5.33 Square Plate with edge boundary conditions

The nodal force response along each boundary for a model with a single 12-node macro element and a model with 9 4-node macro elements are compared with the regular finite element solution. The results are shown below.

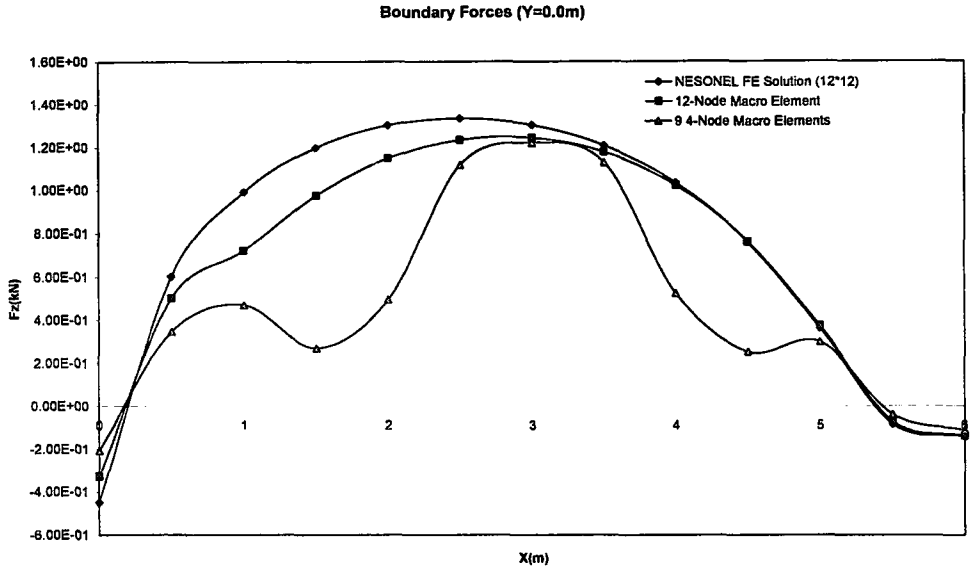


Figure 5.34 Force in Z-direction along Edge1

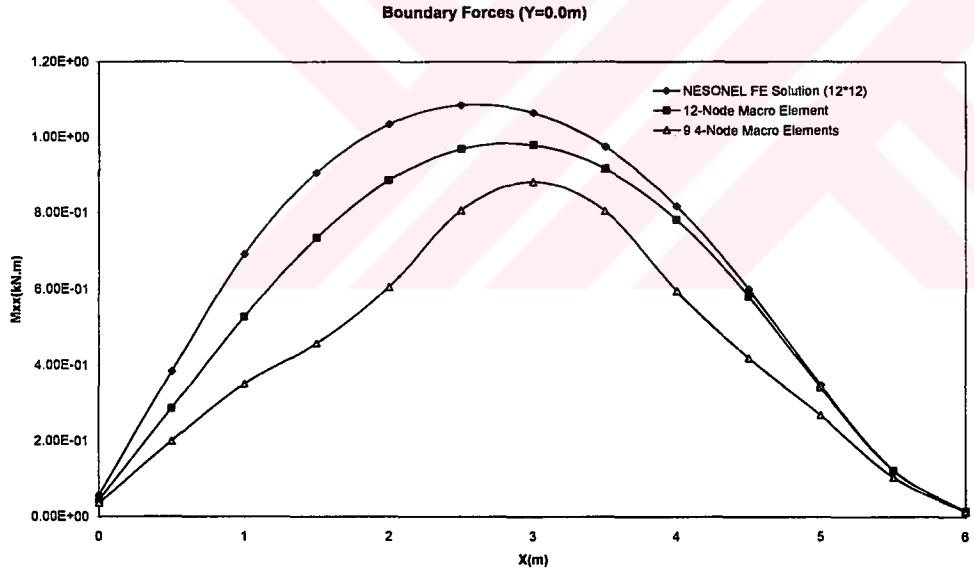


Figure 5.35 Moment about X-axis along Edge1

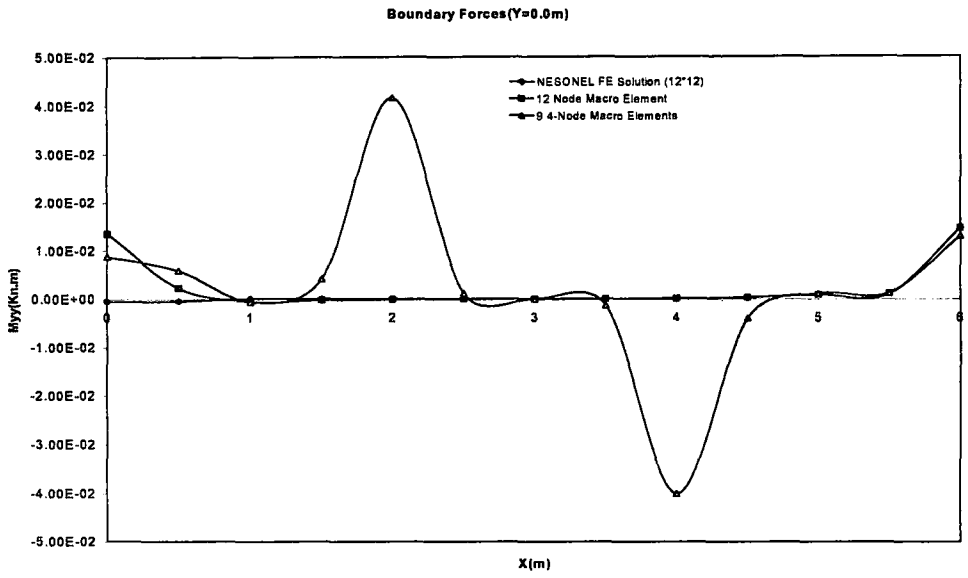


Figure 5.36 Moment about Y-axis along Edge1

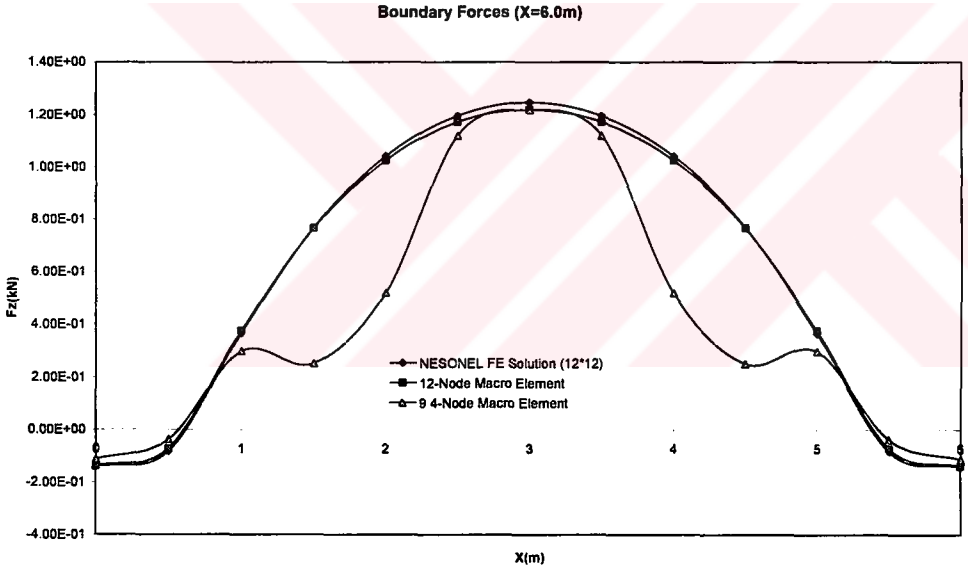


Figure 5.37 Force in Z-direction along Edge2

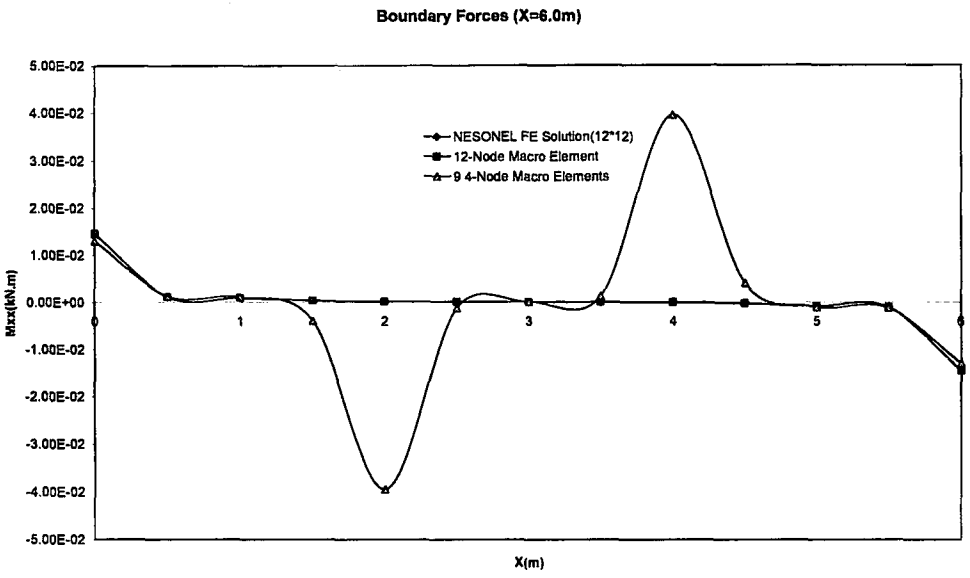


Figure 5.38 Moment about X-axis along Edge2

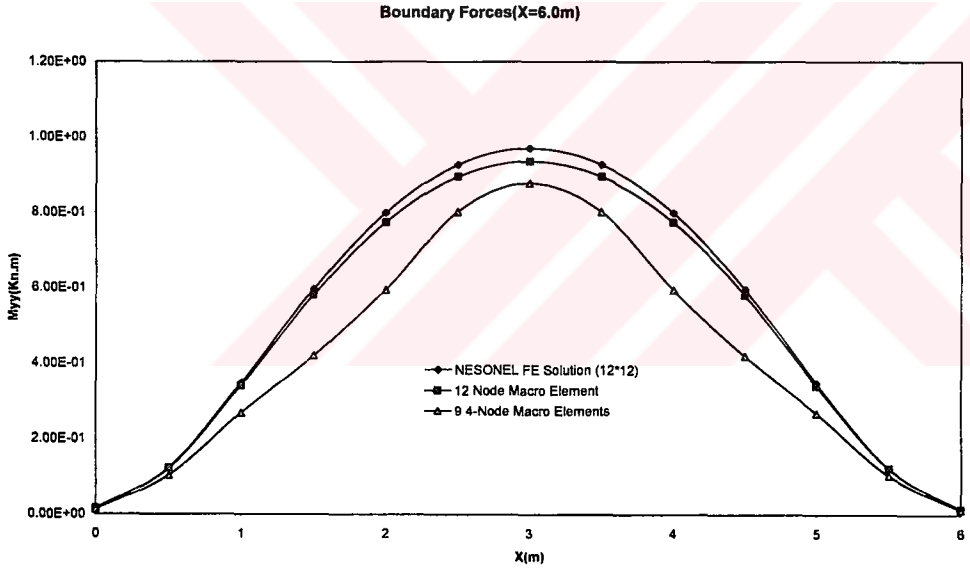


Figure 5.39 Moment about Y-axis along Edge2

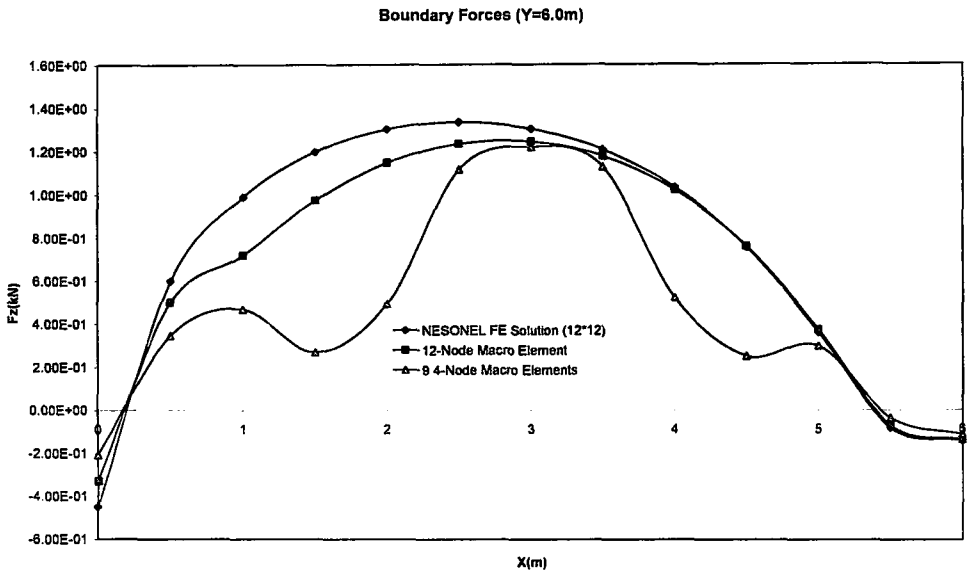


Figure 5.40 Force in Z-direction along Edge3

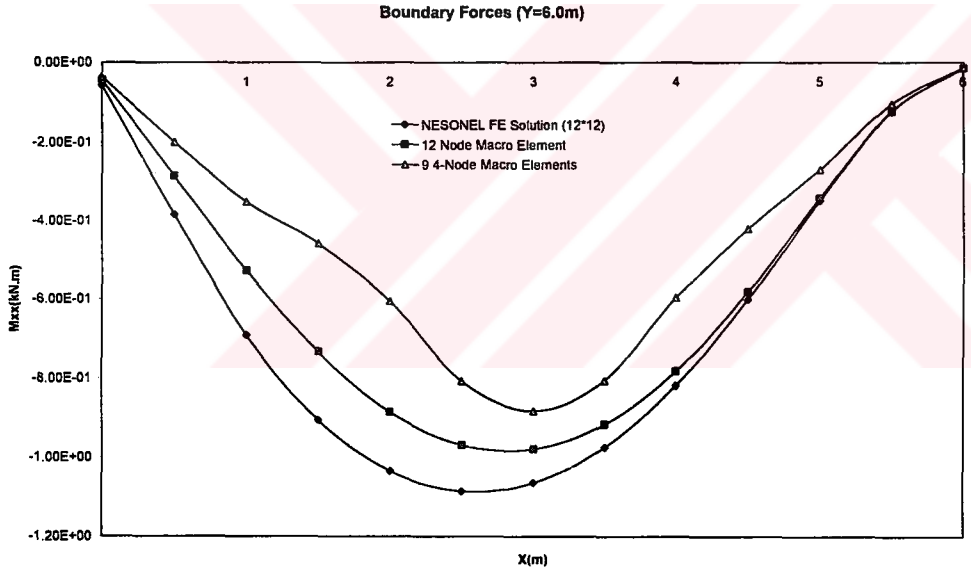


Figure 5.41 Moment about X-axis along Edge3

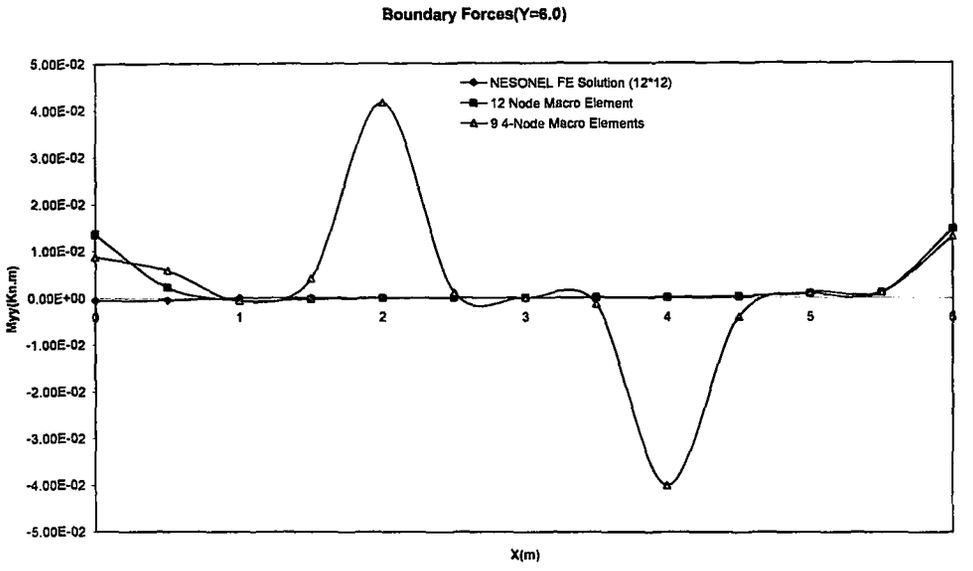


Figure 5.42 Moment about Y-axis along Edge3

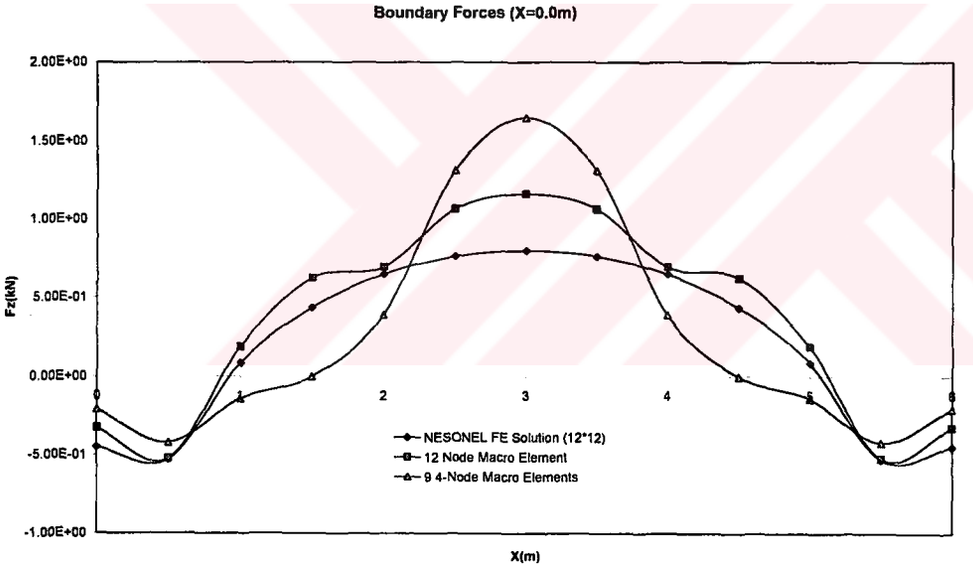


Figure 5.43 Force in Z-direction along Edge4

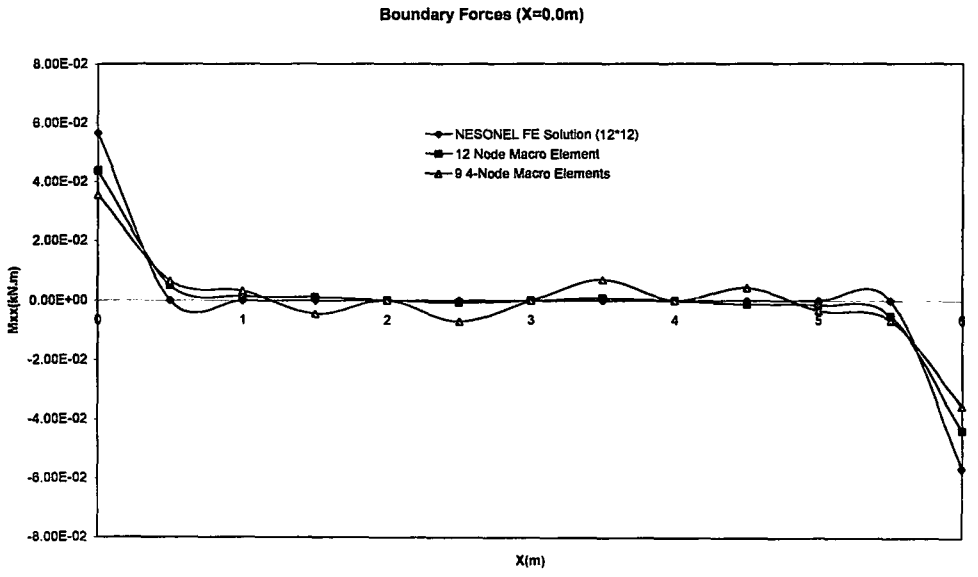


Figure 5.44 Moment about X-axis along Edge4

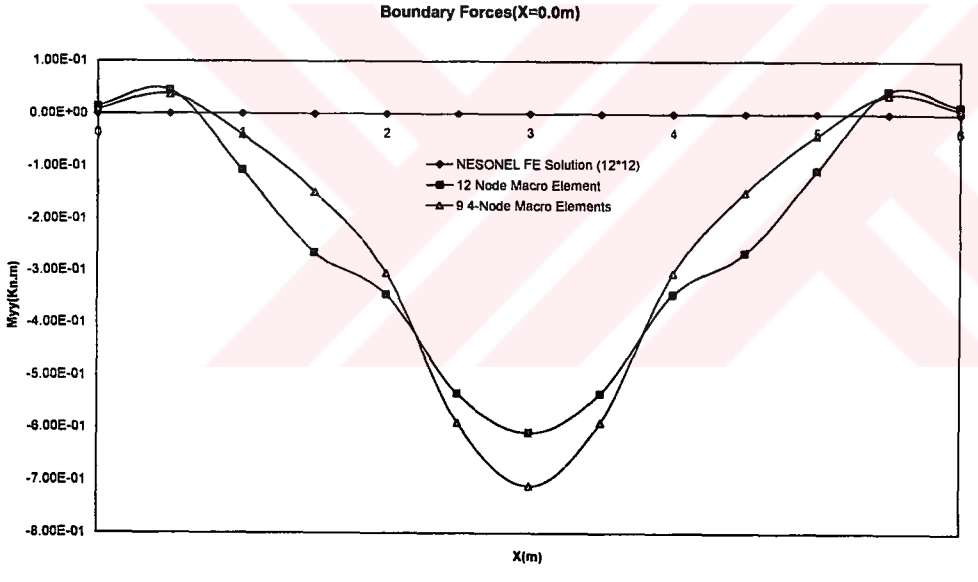


Figure 5.45 Moment about Y-axis along Edge4

Later, the variation of the nodal forces along a line parallel and slightly inside the boundary line is examined along X-direction.

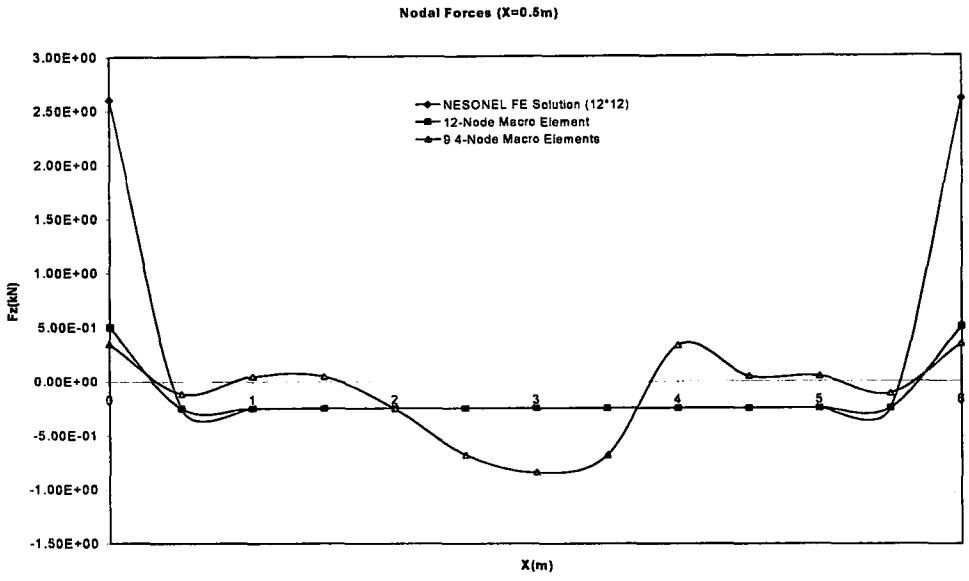


Figure 5.46 Force in Z-direction along X=0.5m

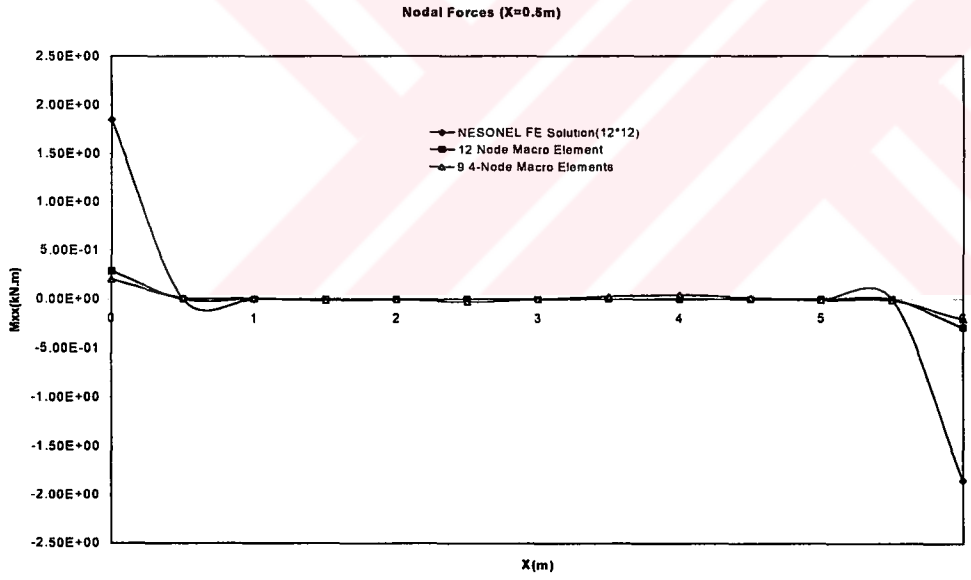


Figure 5.47 Moment about X-axis along X=0.5m

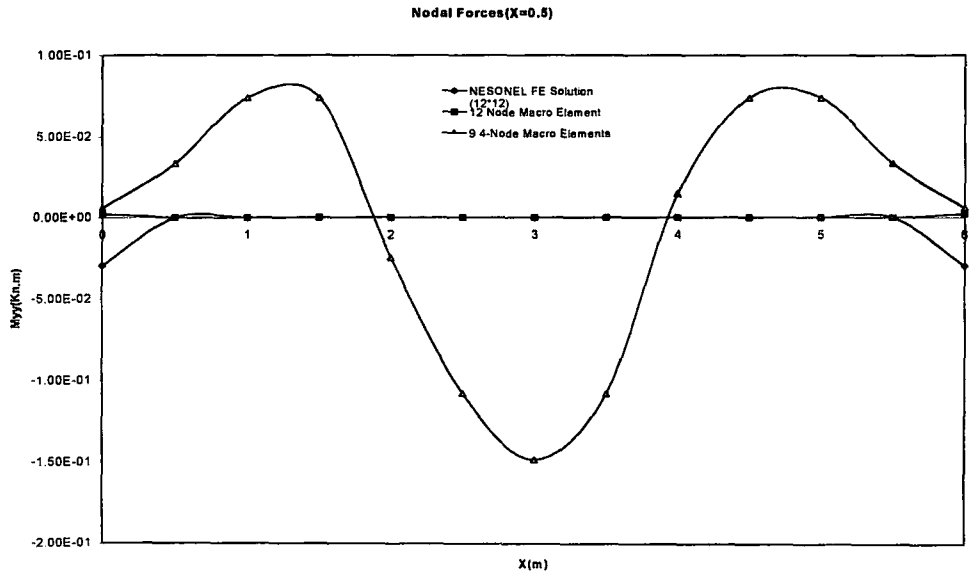


Figure 5.48 Moment about Y-axis along X=0.5m

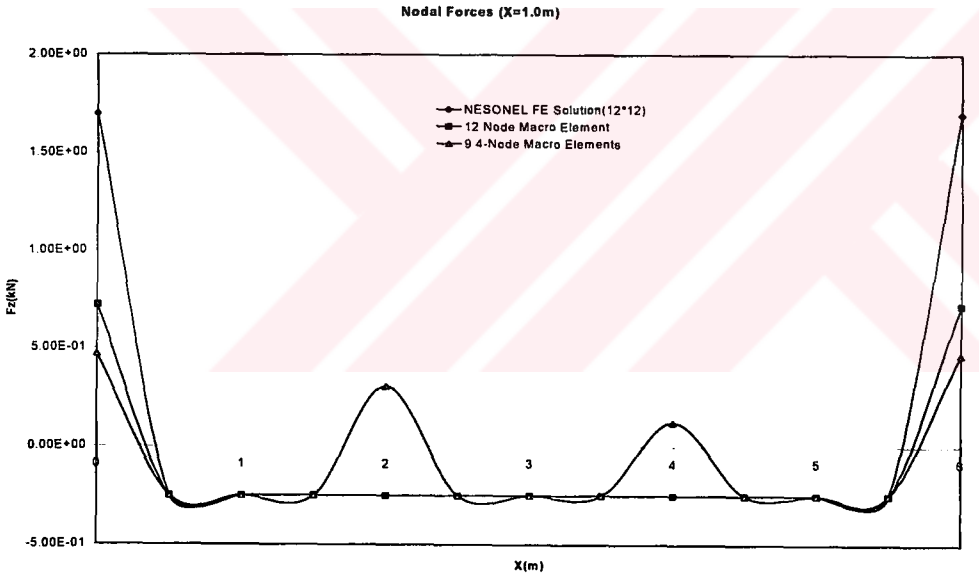


Figure 5.49 Force in Z-direction along X=1.0m

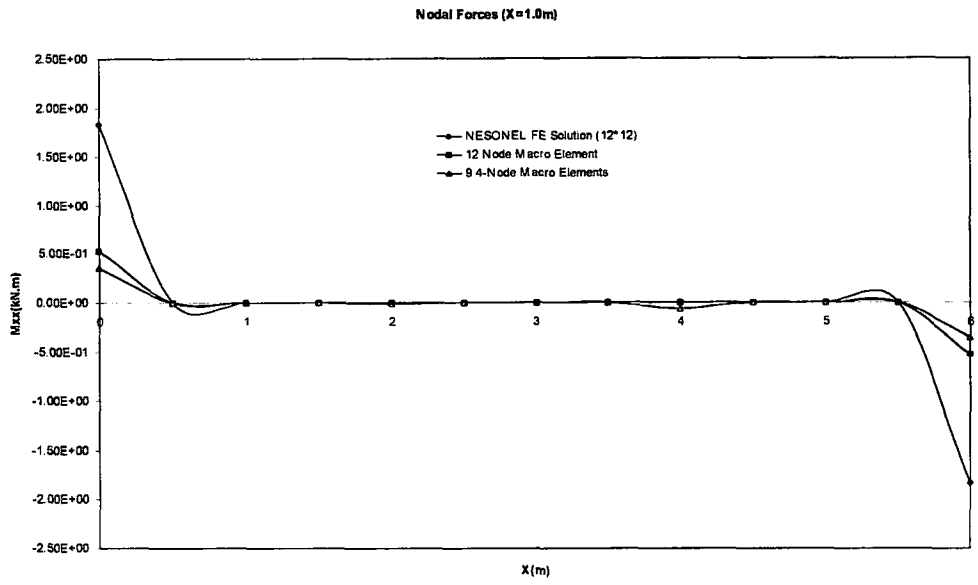


Figure 5.50 Moment about X-axis along X=1.0m

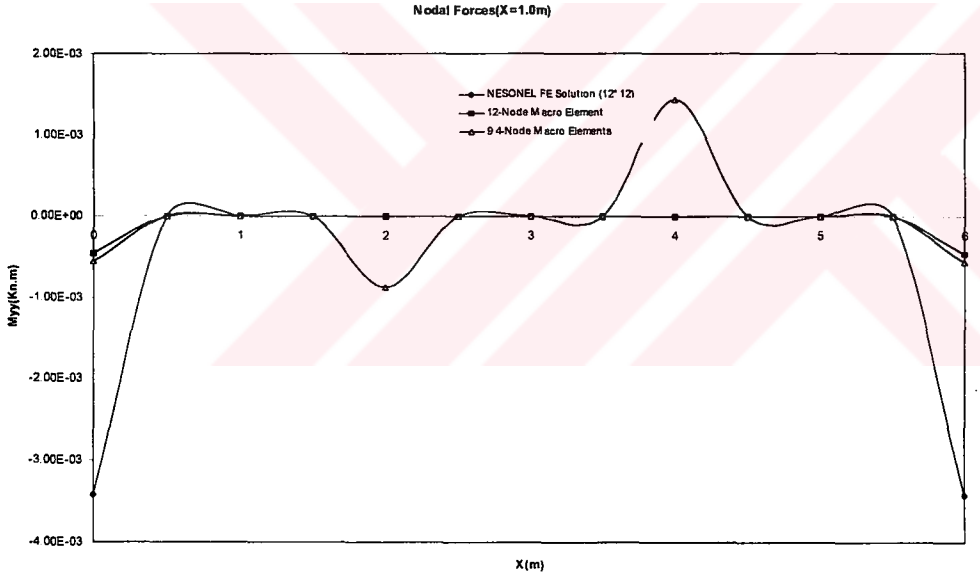


Figure 5.51 Moment about Y-axis along X=1.0m

5.4 Cantilever Plate

As a third case study, a 3x4m cantilever plate subjected to point loads at the corners of its free edge is analyzed. The plate is shown in Figure 5.52

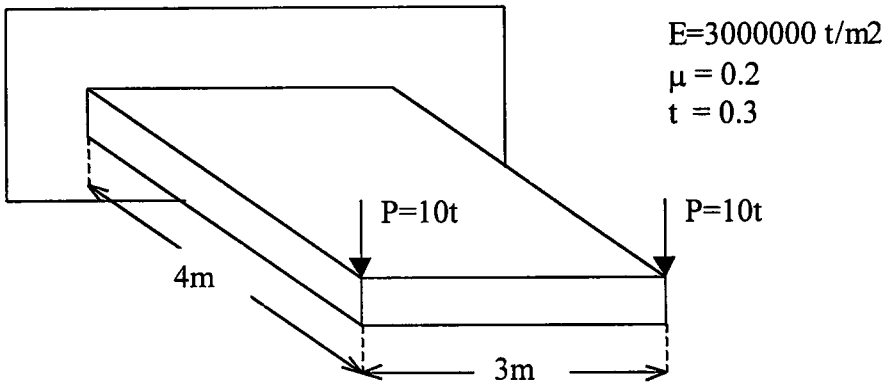


Figure 5.52 Cantilever Plate

The comparison of the results are made by using a regular finite element solution with a mesh density of 16x16, using a single 12-node macro element, and using 9 4-node macro elements. The displacements along Y = 0.5m and Y = 1.5m and the nodal force response along the supported edge are presented in the following graphs.

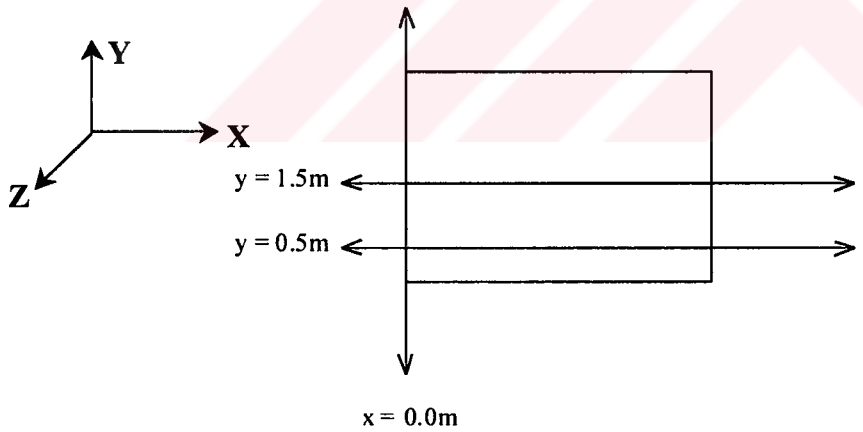


Figure 5.53 Displacement and nodal force control contours

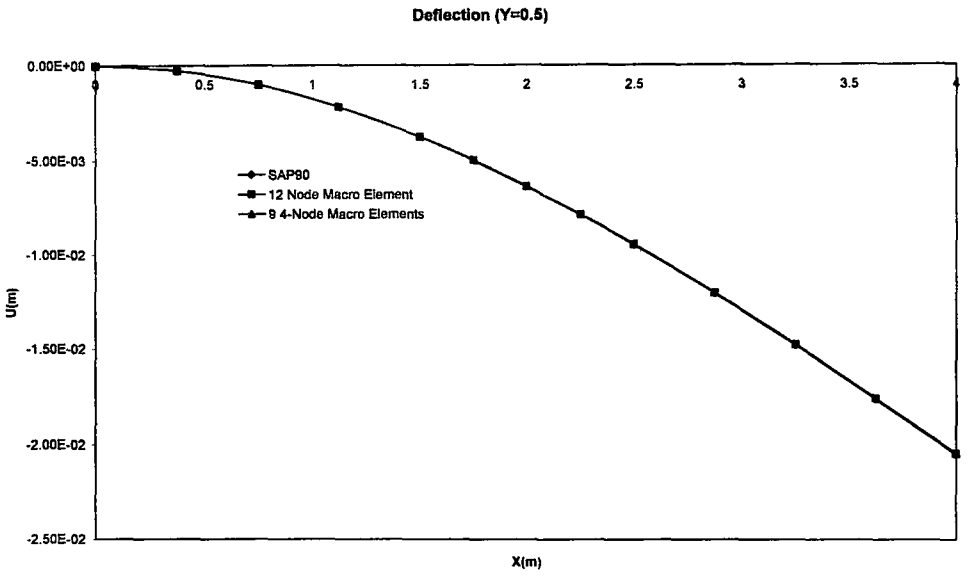


Figure 5.54 Displacement along Y=0.5m

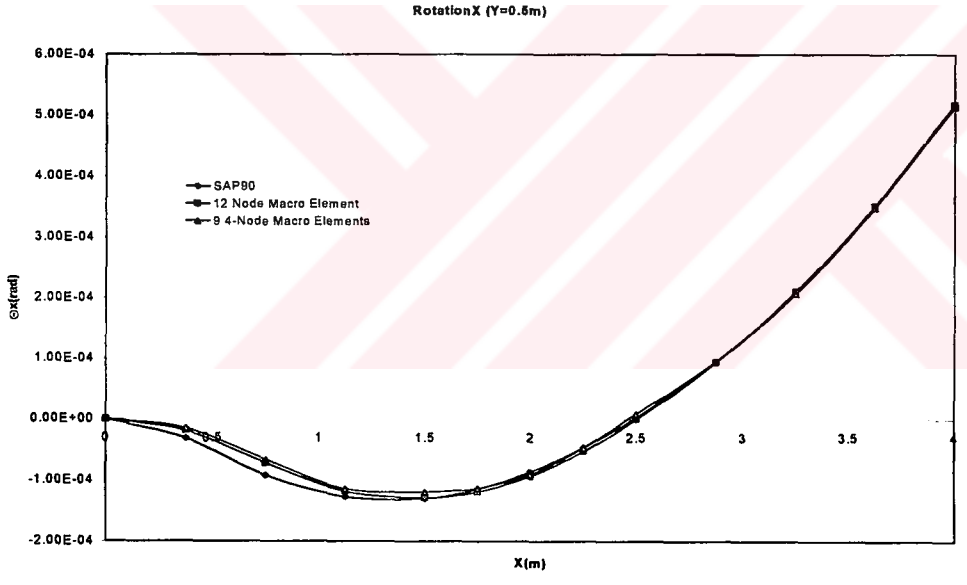


Figure 5.55 Rotation about X-axis along Y=0.5m

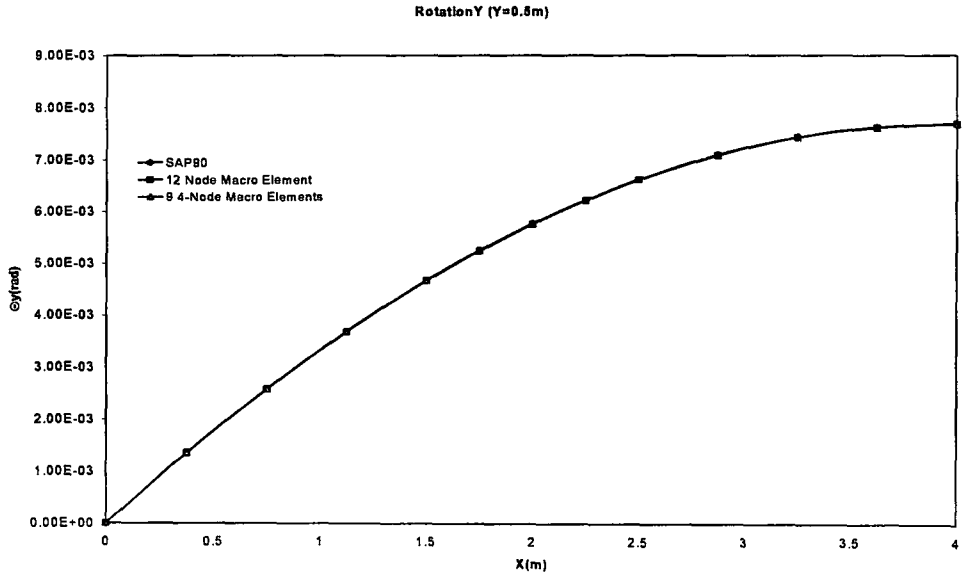


Figure 5.56 Rotation about Y-axis along Y=0.5m

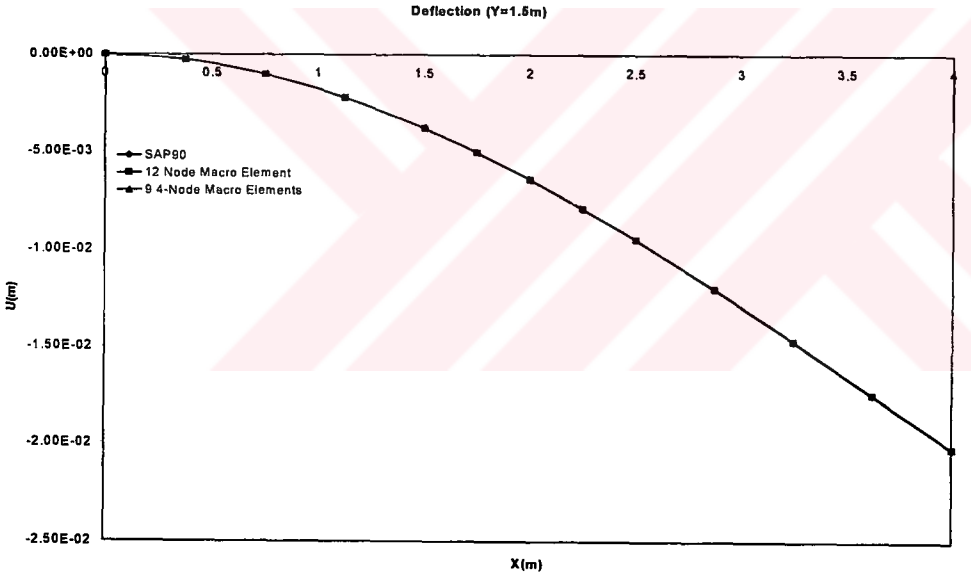


Figure 5.57 Deflection along Y=1.5m

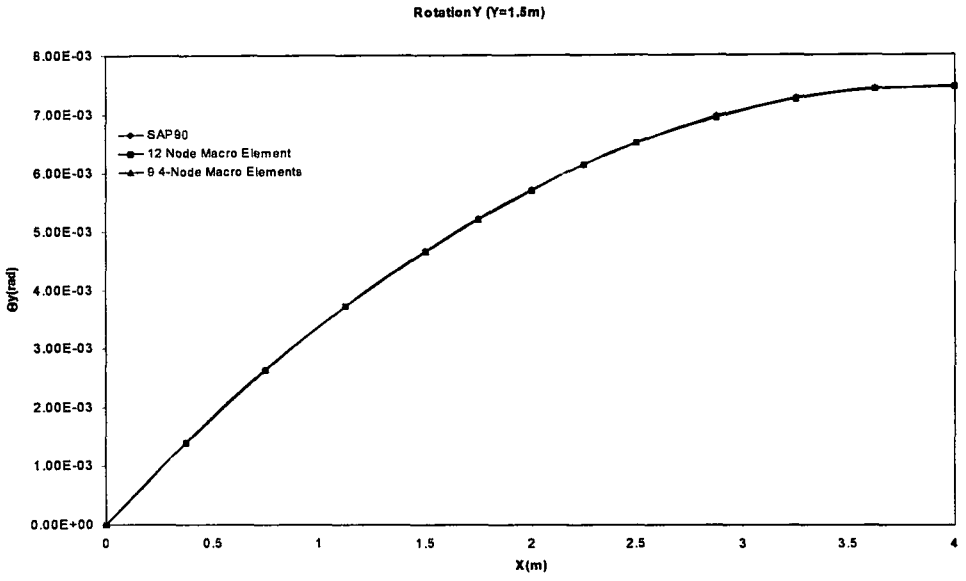


Figure 5.58 Rotation about Y-axis along Y=1.5m

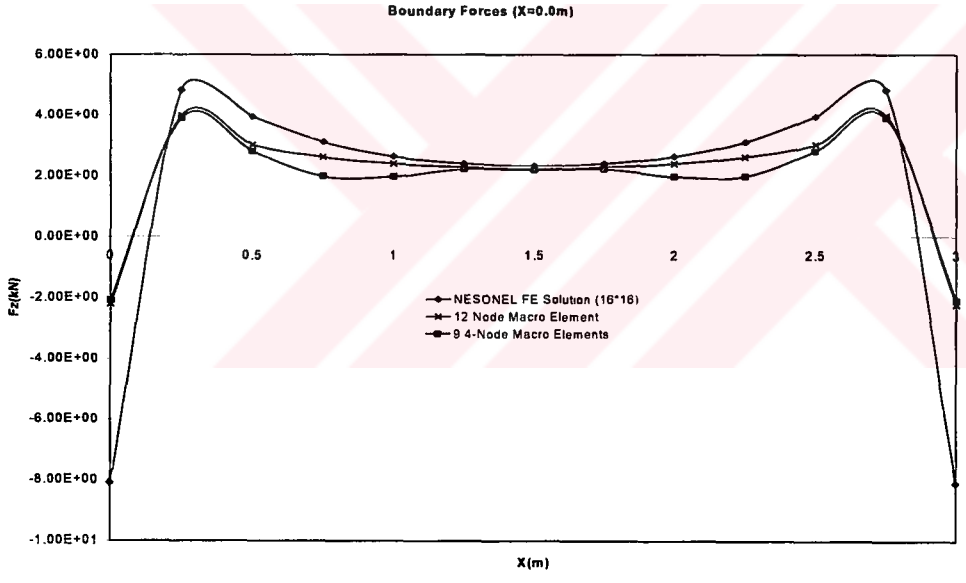


Figure 5.59 Force in Z-direction along X=0.0m

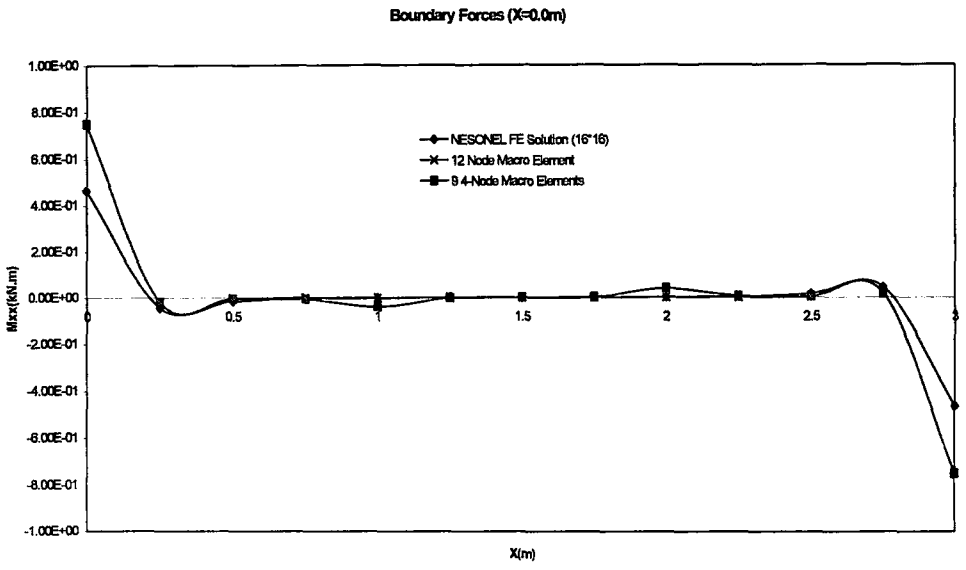


Figure 5.60 Moment about X-axis along X=0.0

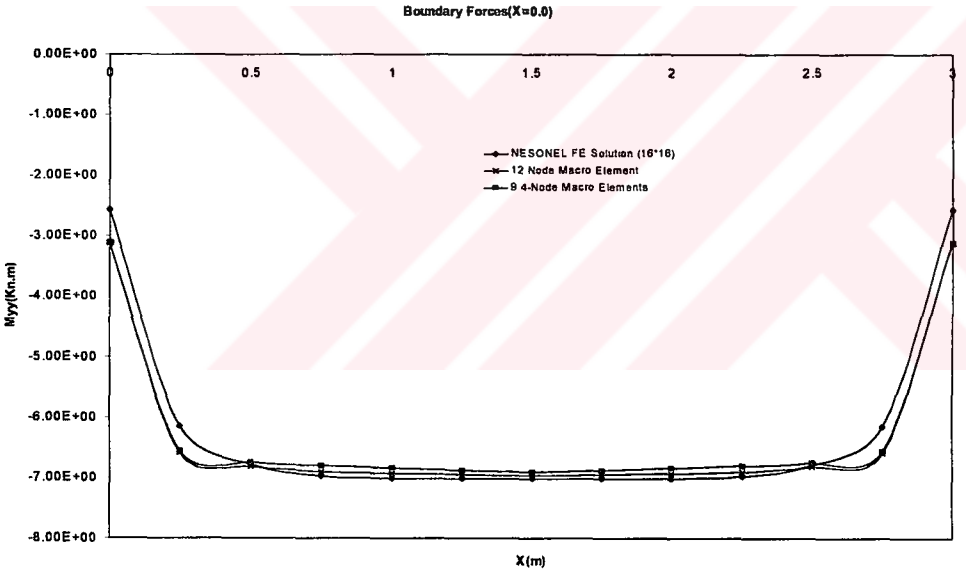


Figure 5.61 Moment about Y-axis along X=0.0

5.4 Cantilever Plate with a Square Hole Inside

Finally, the same cantilever plate with a square hole inside subjected to the same loadings as in the previous case. The plate is shown in Figure 5.62

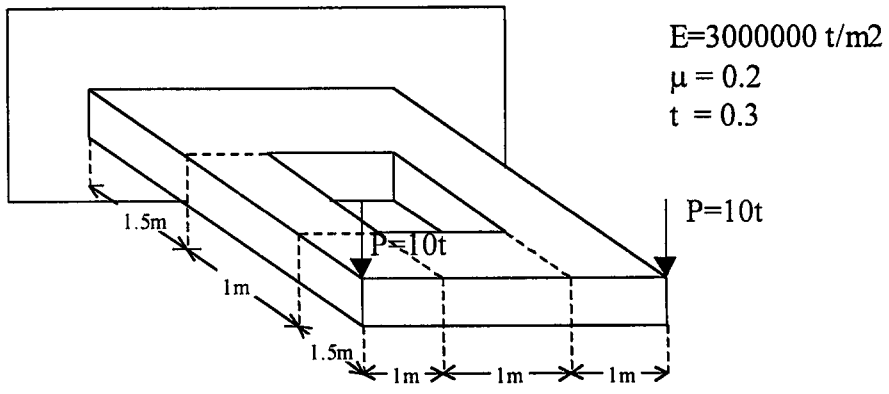


Figure 5.62 Cantilever plate with a square hole inside

The comparison of the results are made by using a regular finite element solution with a mesh density of 16x16, using a single 12-node macro element, using 8 4-node macro elements, and using 8 8-node macro elements. The displacements along Y = 0.5m and Y = 1.5m and the nodal force response along the supported edge are presented in the following graphs.

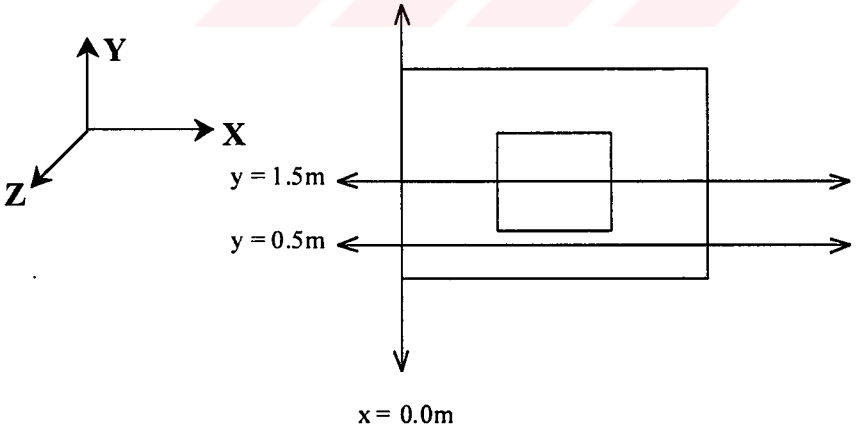


Figure 5.63 Displacement and nodal force control contours

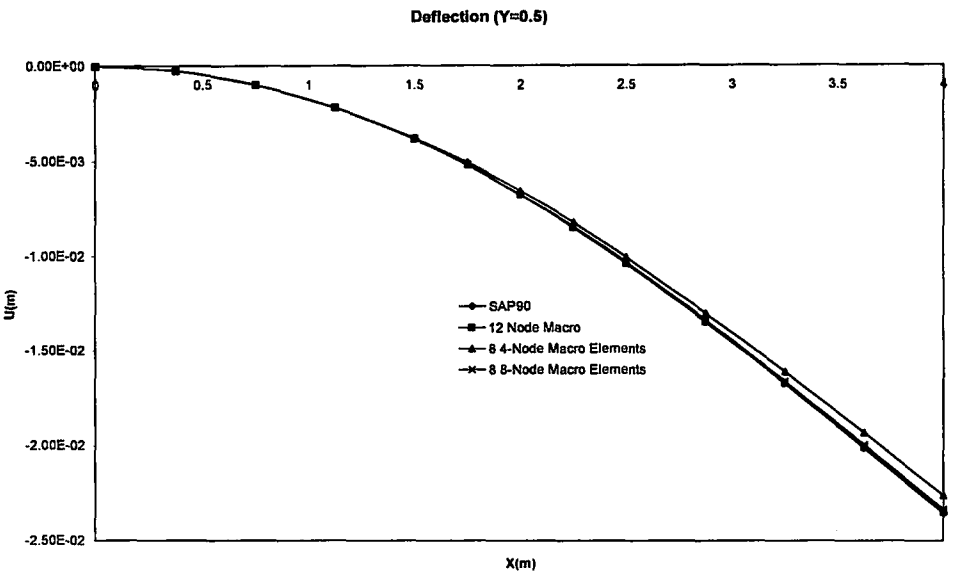


Figure 5.64 Deflection along Y=0.5m

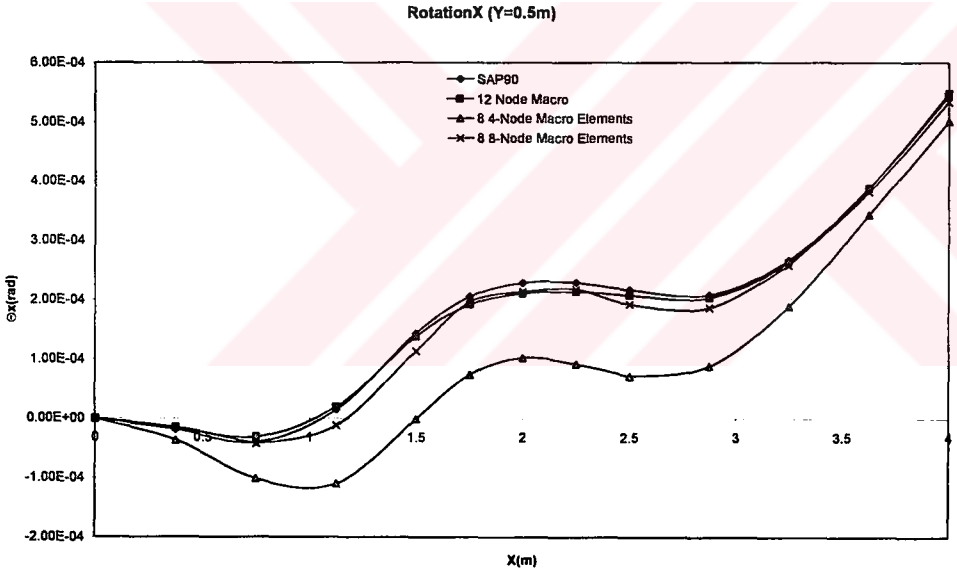


Figure 5.65 Rotation about X-axis along Y=0.5m

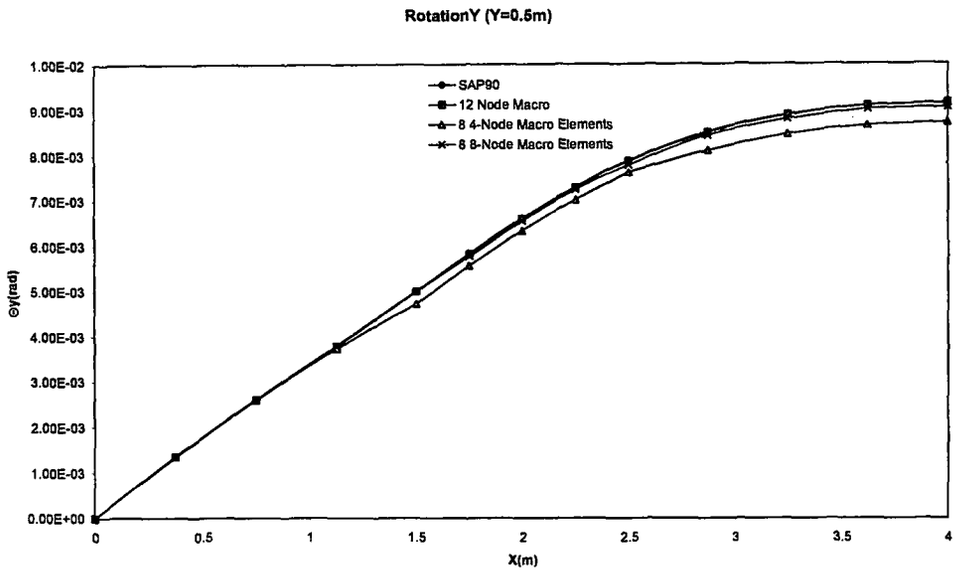


Figure 5.66 Rotation about Y-axis along Y=0.5m

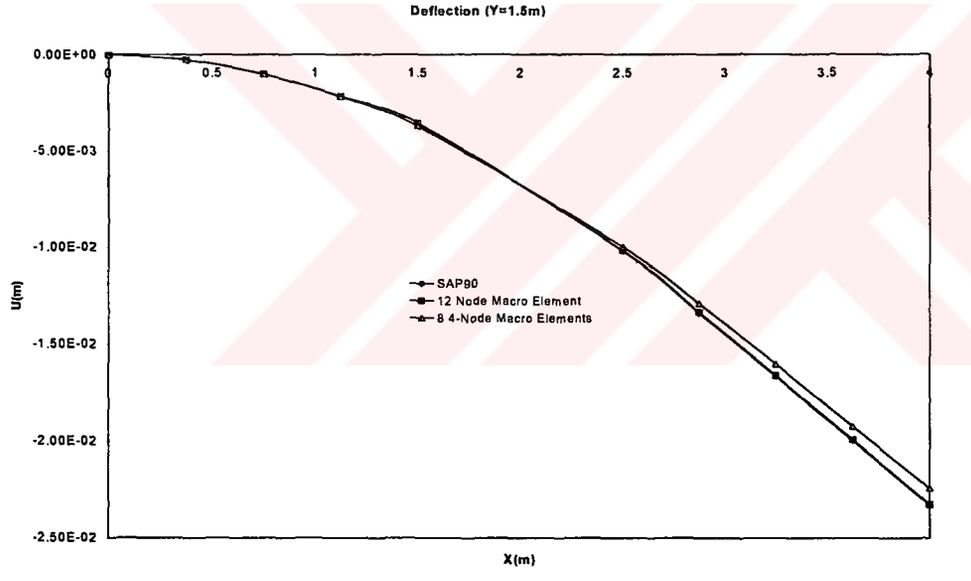


Figure 5.67 Deflection along Y=1.5m

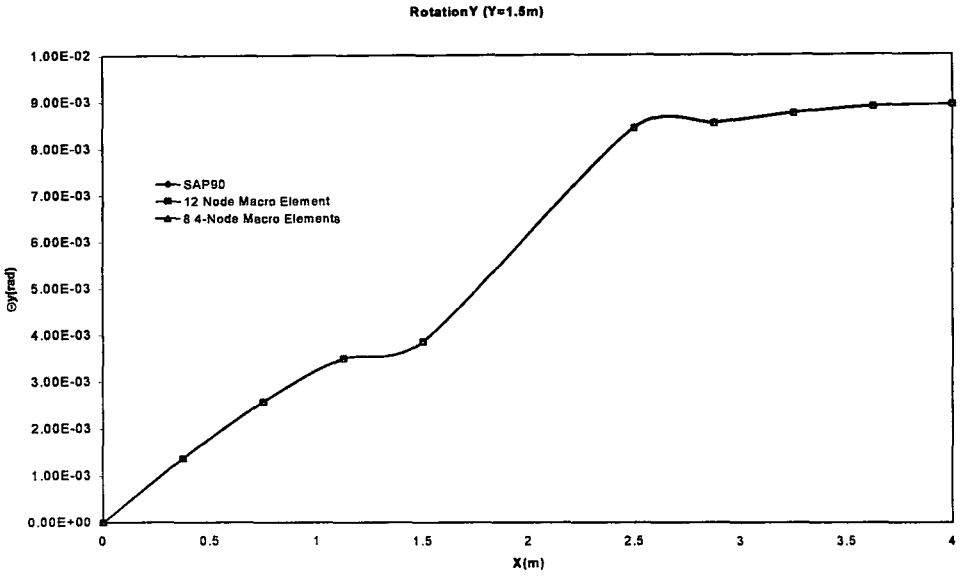


Figure 5.68 Rotation about Y-axis along Y=1.5m

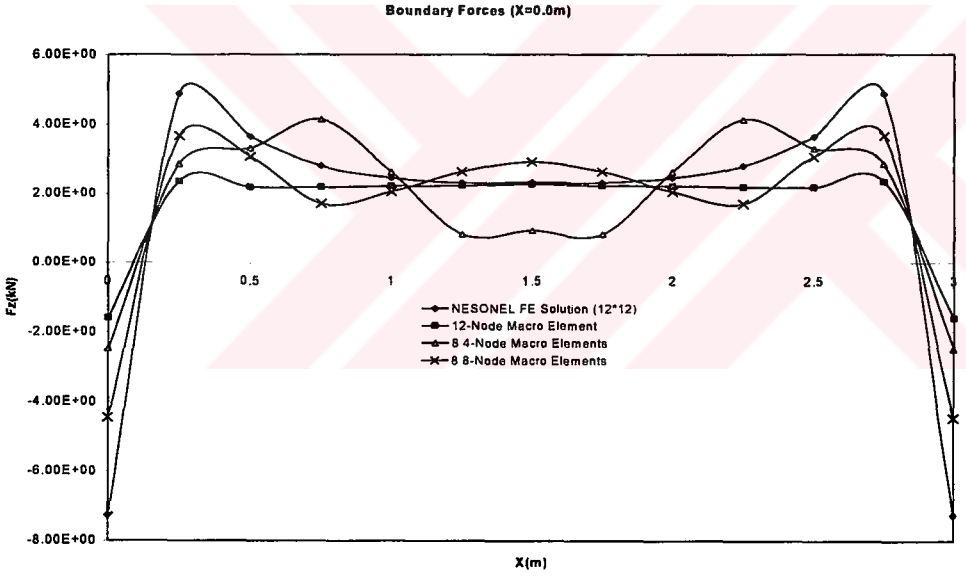


Figure 5.69 Force in Z-direction along X=0.0m

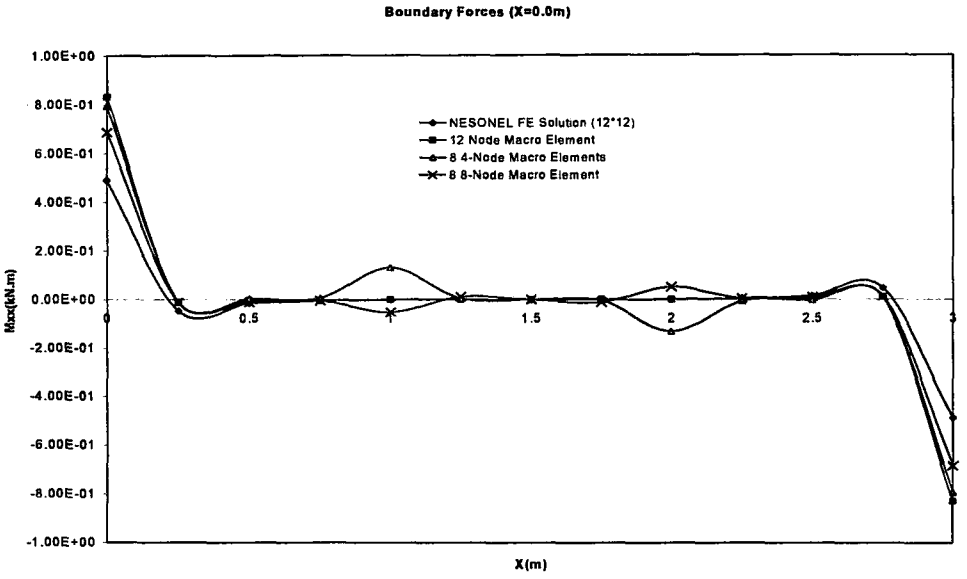


Figure 5.70 Moment about X-axis along X=0.0m

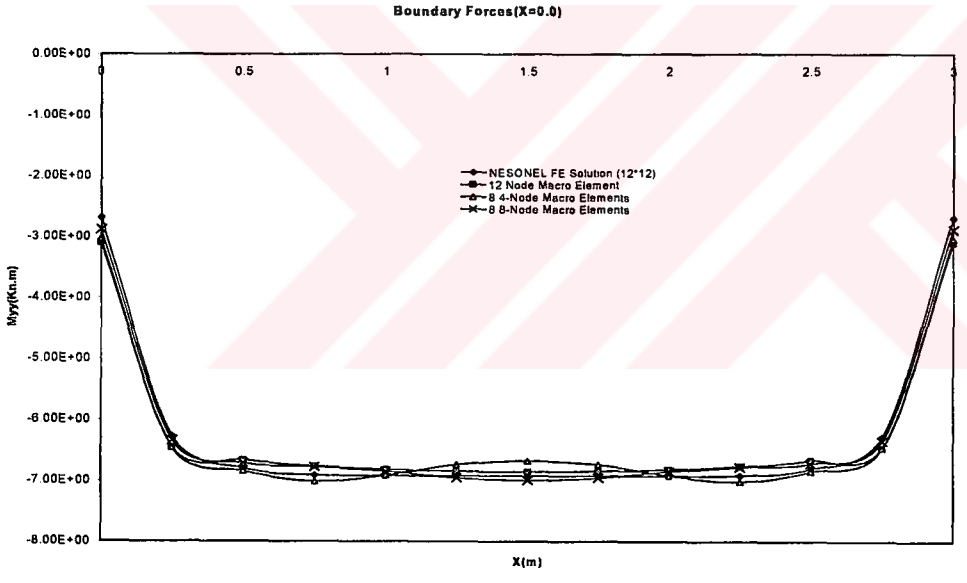


Figure 5.71 Moment about Y-axis along X=0.0m

CHAPTER 6

DISCUSSION OF RESULTS AND CONCLUSIONS

6.1 Discussion of Results

In this study, a plate/shell macro element is formulated and implemented for the analysis of planar wall and slab structures. A number of test structures are first analyzed by the finite element program SAP90[54]. Then, the same structures are modeled and analyzed by using a single 4, 8, and 12-node element and an array of 4-node or 8-node macro elements. The static and kinematic responses of the structures modeled with a varying type and number of macro elements are compared with each other and with the responses obtained from the regular finite element analysis.

A square plate which is simply supported at its corner nodes is examined as the first case study. The structure is subjected to a uniformly distributed load. First, the effect of the mesh density on the solution accuracy is tested by using a 4-node macro element. Because of the assumed and imposed linear variation of the torsional rotation along the element edge, the element has a very stiff behavior. Increasing the mesh density does not improve this deficiency. Later, the same test is repeated by using an 8-node macro element. Using three nodes on each straight boundary increases the degree of assumed displacement interpolation along the boundary. Therefore, the deformation behavior of the structure can be modeled more accurately. This numerical test shows that the mesh density does not have significant effect on the results.

Next, the effect of the number of coupling nodes on the solution accuracy is examined. For this purpose, the SAP90[54] solutions are compared with the 4, 8, and 12-node macro element solutions. The results of 8 and 12-node macro elements have a very good agreement with the finite element solutions. Moreover, they do not have a significant difference in their displacement responses. However, the 4-node macro element is not sufficient to model such a structure. It has a very stiff behavior.

The effect of macro element density on the solution accuracy is also investigated. The structure is discretised by a single 4-node macro element, 4 4-node macro elements and 9 4-node macro elements. As the macro element density increases, the solution accuracy should also increase since it is a kind of finite element, a super element. The results obtained confirm this hypothesis. Except for the single 4-node element results, the other two cases are very close to the finite element solution results.

Finally, for the displacement response comparisons, the displacement responses of a 12-node macro element, 9 4-node macro elements are compared with the finite element solution. They all give nearly the same results.

Then the nodal force responses at the boundaries are examined. The accuracy in the nodal force prediction is very important for design and for correctly determining the loads that are transferred to the edge beams or supports from the macro elements. As is observed from the boundary force responses of the macro elements, there are fluctuations at the free boundaries. Although the displacement responses have a very good agreement, there are significant differences in nodal force responses. This is due to the application of the constraint equations at the boundaries. They

cause an external disturbance that disturbs the stress response of the elements near the boundaries.

The same square plate element is restrained along the edges in order to work on common model that are used in the analysis of floor slabs in a building structure. Only the response of nodal forces are checked both on the boundaries and inside the plate domain. At the boundaries, especially at the hinge supported boundary, there are deviations from the finite element results. However, at the inside nodes the exact results are obtained by the 12-node macro element. There are fluctuations inside the plate at the interface boundaries in the model with 9 4-node macro elements. Again, this is due to the disturbances at the macro element boundaries.

The third and fourth case studies were done by using a cantilever plate. First, a simple rectangular cantilever was analyzed using a 12-node macro element, 9 4-node macro element and regular finite element solution by SAP90[54]. Very accurate results are obtained both with the 12-node and the 9 4-node macro element models. This verifies that the assumed displacement field is sufficient to model such a structure. The nodal forces along the supported edge are almost the same when the finite element solution is compared with the other solutions. Small fluctuations are observed in case of the model with 9 4-node macro elements. At the fixed end, the constraint equations are not applied since the values of the displacements are all zero. The difference between the results is due to the prescribed displacements imposed along other edges.

In order to check the efficiency of the macro element in modeling arbitrarily shaped structures, the same cantilever with a square hole inside is also analyzed. The cantilever plate is modeled using a single 12-node macro element, 8 4-node macro elements and 8 8-node macro element. If the

displacement results of the 4-node and 8-node macro element models are compared, it is seen that the prescribed variation of the torsional rotation is not adequate to model the actual variation of the torsional rotation along the element boundaries. This deficiency is corrected by modeling the same structure with higher order macro elements. The nodal forces at the supported edge are seen to be acceptably close enough for design.

6.2 Conclusions

After all these case studies and the previous work, it can be said that the planar plate/shell macro elements can model the displacement response of structural components in bending very successfully. However, because of the linear interpolation of the torsional rotation between any two adjacent coupling nodes along the element perimeter, the 4-node macro elements do not give acceptable results. They are very stiff. However, when the degree of the interpolation functions is increased by taking more coupling nodes along each boundary, very satisfactory results can be obtained. Therefore, in the analysis of structures with macro elements, elements having at least three coupling nodes along each edge must be preferred.

The macro element has some problems in predicting the stress response near the boundaries. Fluctuations, even along the free-edges, are usually observed. This is due to the application of the displacement constraints along the boundaries for the prescribed displacement variation. Although, this process does not change the internal energy of the system, it changes its distribution in the element domain. This external disturbance changes the distribution of the internal stresses. At the inner domain nodes, the disturbance dies out very quickly and the stress predictions are very close to those obtained by a regular finite element solution.

Increasing the mesh density does not have any significant effect on the solution accuracy. On the other hand, increasing the number of coupling nodes or the number of macro elements improves the behavior of the element significantly.

The macro elements are very practical tools for the analysis of the large structural systems. With the correction of the stress response predictions near the boundaries, they will become very accurate and efficient elements.

REFERENCES

[1] Cook, R.D., Malkus, D.S. and Plesha, M.E., "Concepts and Applications of Finite Element Analysis", Wiley, 1989

[2] Reddy, J.N., "An Introduction to the Finite Element Method", McGraw-Hill, 1993

[3] Zienkiewicz, O.C. and Taylor, R.L., "The Finite Element Method: Basic Formulation and Linear Problems", Vol.1, McGraw-Hill, 1989

[4] Zienkiewicz, O.C. and Taylor, R.L., "The Finite Element Method: Solid and Fluid Mechanics, Dynamics and Non-Linearity", Vol.2, McGraw-Hill, 1991

[5] Bathe, K.J., Wilson, E.L., "Numerical Methods In Finite Element Analysis", Wiley, 1976

[6] Owen, D.R.J. and Hinton, E. ,"Finite Elements In Plasticity", Pineridge Press Limited, 1980

[7] Boresi, A.P., Schmidt, R.J. and Sidebottom, O.M., "Advanced Mechanics of Materials", Wiley, 1993

[8] Timoshenko,S.P., Woinowsky-Krieger,S., "Theory of Plates and Shells", McGraw-Hill, 1959

[9] Meyer, B., "Object-Oriented Software Construction", 1997

[10] Booch, G., "Object-Oriented Analysis and Design with Applications", 1994

[11] Martin, R.C., " Designing Object Oriented C++ Applications Using Booch Method", 1995

[12] Argeso, A.H., "A Panel Macro Element for the Analysis of Structural Wall Systems", Master Thesis, METU, 1996

[13] Bahat, B.H., "Analysis of Structural Wall Systems", Master Thesis, METU, 1998

[14] Irons, B.M., "A Frontal Solution Program for Finite Element Analysis", Int. Jour. Num. Meth. Engng. , Vol.2, 1970, pp. 5-32

[15] Beer, G. and Haas, W., "A Partitioned Frontal Solver For Finite Element Analysis", Int. Jour. Num. Meth. Engng. , Vol.18, 1982, pp. 1623-1654

[16] Thompson, E. and Shimazaki, Y. "A Frontal Procedure Using Skyline Storage", Int. Jour. Num. Meth. Engng., Vol.15, 1980, pp. 889-910

[17] Hrabok, M.M. and Hrudey, T.M., "A Review and Cataloque of Plate Bending Finite Elements", Computers&Structures, Vol.19, No.3, 1984, pp. 479-495

[18] Melosh, R.J. "A Stiffness Matrix for the Analysis of Thin Plates In Bending", J. Aeronaut. Sci., Vol.28, No.34, 1961

[19] Deak, A.L. and Pian, T.H., "Application of the Smooth Surface Interpolation to the Finite Element Analysis", A.I.A.A.J., Vol.5, 1967, pp.187-189

[20] Dhatt, G. and Venkatasubbu, S., "Finite Element Analysis of Containment Vessels", Proc. First Int. conf. On. Struc. Mech. In Reac. Tech., Berlin, Germany, Vol.5, 1971, paper J. 3/6

[21] Fried, I. "Residual Energy Balancing Technique in the Generation of Plate Bending Finite Elements", Computers&Structures, Vol.4, No.4, 1974, pp. 771-778

[22] Bathe, K.J., Wilson, E.L. and Peterson, F., "SAP IV: A Structural Analysis Program for Static and Dynamic Response of Linear Systems", College of Engineering, University of California, Berkeley, Ca., USA, 1974

[23] Clough, R.W. and Felippa, C.A., "A Refined Quarilateral Element for the Analysis of Plate Bending", Proc. Conf. On Matrix Methods In Struc. Mech., 1968, Ohio, pp.399-440

[24] McNeal, R.H., "A Simple Quadrilateral Shell Element", Computers and Structures, Vol.8, 1978, pp. 175-183

[25] Cook, R., "Some Elements for Analysis of Plate Bending", ASCE, EM6, 1972, pp. 1453-1470

[26] Hughes, T., Taylor, R. and Kanoknukulachai, W., "A Simple and Efficient Finite Element for Plate Bending", Int. Jour. Num. Meth. Engng., Vol.11, 1977, pp.1529-1543

[27] Robinson, J. and Haggenmacher, G. "Lora: An Accurate Four Node Stress Plate Bending Element", Int. Jour. Num. Meth. Engng., Vol.14, 1979, pp. 296-306

[28] Batoz, J.L, Bathe, K.L. and Ho, L.W., "A Study of Three Node Triangular Plate Bending Elements", Int. Jour. Num. Meth. Engng., Vol.15, 1980, pp. 1771-1812

[29] Batoz, J.L., "An Explicit Formulation for an Efficient Triangular Plate Bending Element", Int. Jour. Num. Meth. Engng., Vol.18, 1982, pp. 1077-1089

[30] Batoz, J.L., and Tahar, M.B., "Evaluation of a New Quadrilateral Thin Plate Bending Element", Int. Jour. Num. Meth. Engng., Vol.18, 1982, pp. 1655-1677

[31] Ibrahimbegovic, A. and Wilson, E.L., "Thick Shell and Solid Finite Elements with Independent Rotation Fields", Int. Jour. Num. Meth. Engng., Vol.31, 1991, pp. 1393-1414

[32] Ibrahimbegovic, A., "Quadrilateral Finite Elements for Analysis of Thick and Thin Plates", Comp. Meth. App. Mech. Engng., Vol.110, 1993, pp.195-209

[33] MacNeal, R.H. and Harder, R.L., "A Refined Four-Noded Membrane Element with Rotational Degrees of Freedom", Computer and Structures, Vol.28, 1988, pp.75-88

[34] Irons, B.J. and Ahmad, S., "Techniques of Finite Elements", Ellis Horwood, Chichester, UK, 1980

[35] Allmann, D.J., "A Compatible Triangular Element Including Vertex Rotations for Plane Elasticity Analysis", Computers&Structures, Vol.19, 1984, pp.1-8

[36] Allmann, D.J., "A Quarilateral Finite Element Including Vertex Rotations for Plane Elasticity Analysis", Int. Jour. Num. Meth. Engng., Vol.26, 1988, pp.717-730

[37] Bergan, P.G. and Felippa, C.A., "A Triangular Membrane Element With Rotational Degrees of Freedom", Comp. Meth. Appl. Mech. Eng., Vol.50, 1985, pp.25-69

[38] Hughes, T.J.R and Brezzi, F., "On Drilling Degrees of Freedom", Comp. Meth. Appl. Mech. Eng., Vol.72, 1989, pp. 105-121

[39] Ibrahimbegovic, A., Taylor, R.L. and Wilson, E.L. "A Robust Quadrilateral Membrane Finite Element with Drilling Degrees of Freedom", Int. Jour. Num. Meth. Engng., Vol.30, 1990, pp.445-457

[40] Ibrahimbegovic, A. and Wilson, E.L., "A Unified Formulation for Triangular and Quadrilateral Flat Shell Finite Elements with Six Degrees of Freedom", Comm. Appl. Num. Methods, Vol.7, 1991, pp.1-9

[41] Belytschko, T., Lin, J.L. and Tsay, C.S., "Explicit Algorithms for the Nonlinear Dynamics and Shell", Comput. Meth. App. Mech. Eng., Vol.42, 1984, pp.225-251

[42] Engelmann, B.E. and Whirley, R.G., "A New Elastoplastic Shell Element Formulation for DYNA3D", Lawrence Livermore National Laboratory, 1990, Report UCRL-JL-104826

[43] Belytschko, T. and Leviathan, I., "Physical Stabilization of Four Node Shell Element with One-Point Quadrature", Comp. Meth. Appl. Eng., Vol.13, 1994, pp.277-286

[44] Zhu,Y. and Zacharia, T., "A New One-Point Quadrature Quadrilateral Shell Element With Drilling Degrees of Freedom", Comp. Meth. Appl. Mech. Eng., Vol.136, 1996, pp.165-203

[45] Rehak, D.R. and Bauhg, J.W., "Alternative Programming Techniques for Finite Element Program", Development Proc., IABSE Colloquium on Expert Systems In Civil Engineering, Bergamo, Italy, 1989

[46] Forde, B.W.R, Foschi, R.B and Stiemer, S.F., "Object-Oriented Finite Element Analysis", Computers and Structures, Vol.34, 1990, pp.355-374

[47] Zimmermann, T., Dubois-Pélerin, Y. and Bomme,P., "Object-Oriented Finite Element Programming: I-Governing Principles", Comp. Meth. Appl. Mech. Eng., Vol.98, 1992, pp.291-303

[48] Dubois-Pélerin, Y. and Zimmermann, T., "Object-Oriented Finite Element Programming: III- An Efficient Implementation in C++", Comp. Meth. Appl. Mech. Eng., Vol.108, 1993, pp.165-183

[49] Mackie, R.I., "Object-Oriented Programming of the Finite Element Method", Int. Jour. Num. Meth. Eng., Vol.35, 1992, pp.425-436

[50] Zeglinski, G.W. and Han, R.P.S., "Object-Oriented Matrix Classes for Use in a Finite Element Code Using C++", Int. Jour. Num Meth. Eng., Vol.37, 1994, pp.3921-3937

[51] Besson, J. and Foerch, R., "Large Scale Object-Oriented Finite Element Code Design", Comp. Meth. Appl. Mech. Eng., Vol.142, 1997, pp. 165-187

[52] Scholz, S.P., "Elements of an Object-Oriented FEM++ Program in C++", Computers and Structures, Vol.43, No.3, 1992, pp. 517-529

[53] Kaliakin, V.N. "A Simple Coordinate Determination Scheme for Two Dimensional Mesh Generation", Computers and Structures, Vol.43, 1992, pp.505-516

[54] Habibullah, A. and Wilson, E.L., "SAP90- A General Structural Analysis Program User's Guide", Computers and Structures, Inc., Berkeley, California, 1989

APPENDIX A

INPUT FILE FORMAT

There are 13 different control identifiers. Although it is possible to write any kind of explanation between the identifiers, inside the identifier the given format must be strictly applied.

1- NODES Data Block

Nodes identifier must be the first data block. It is used to define the global nodes of the structural system. After the NODES identifier, the number of nodes must be entered in the following format

nNodes=...

The following lines are for the Node ID and its coordinates. The complete format is shown below.

```
NODES
nNodes=2
1       X=0.0  Y=0.0  Z=0.0
2       X=2.0  Y=0.0  Z=0.0
```

It should be noted that the node ID's are valid during the initial geometry definition. The nodes will be renumbered during the mesh generation and solution phase. Hence, the numbering of the nodes must be in order.

2- RESTRAINTS Data Block

RESTRAINTS identifier is used to define nodal restraints. Even though the definition of edge restraints is possible, the restraints for the global nodes must be defined separately. The second line is used to determine the number of nodal restraints. Afterwards the restraint data must be determined. This data section has the following form.

NodeId R= r_{ux}, r_{uy}, r_{uz}, r_{rx}, r_{ry}, r_{rz}

NodeId determines the node to be restrained. r_{ux}, r_{uy}, r_{uz}, r_{rx}, r_{ry}, r_{rz} defines whether a restraint in translation in x,y,z direction and rotation in x,y,z directions respectively exists. Their value can be either TRUE or FALSE.

The format of RESTRAINTS data block is given below:

```
RESTRAINTS
nRes=2
1     R=1,1,1,1,1,1
2     R=1,1,0,0,0,1
```

A node without inactive degrees of freedom need not be specified.

3- EDGES Data Block

The EDGES data block is for the definition of the lines used to define polygons. The second line of this block determines the number of edges

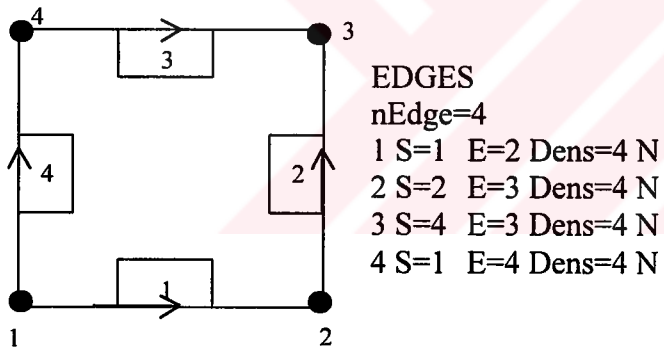
with the identifier "nEdge=". The following edge data should have the following form:

EdgeId S=StartNode E=EndNode Dens=MeshDensity R=r_{ux},r_{uy},r_{uz},r_{rx},r_{ry},r_{rz}

EdgeId's are used in the definition of polygons and they are not valid during solution phase. As NodeId's, they must be in order. MeshDensity property is used during mesh generation and it determines the number of elements that belong to that edge. R is used for restraining the nodes belonging to the edge. If no inactive degrees of freedom exist for an edge, then the identifier "N" indicating "none" must be entered.

One important point during the definition of the edge connectivity is the direction of the edges. They must be from left to right or from bottom to top.

An example to EDGE data block is given below:



EDGES
nEdge=4
1 S=1 E=2 Dens=4 N
2 S=2 E=3 Dens=4 N
3 S=4 E=3 Dens=4 N
4 S=1 E=4 Dens=4 N

Figure A.1 Edge data input format

4- POLYGONS Data Block

This data block is for defining the polygons, its material type and the element type. The number of polygons to be defined in this data block is

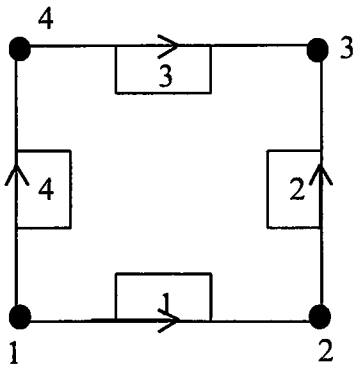
determined by "nPoly" identifier. Then the following form of poygon definition must be entered.

PolygonId C=EdgeI, EdgeJ, ... M=MaterialId eType=ElementCode

The polygonId's must be in order if this polygon will be used as a macro element. Otherwise no limitations exist for the polygon identification numbers. The connectivity information for a polygon is determined by defining its edges in a counter-clockwise manner. Elements belonging to a polygon will have the same material property that is defined in MaterialId. ElementType identifier determines the mechanical element used in the solution. For the time being the program has 7 different mechanical elements whose Element Codes and definitions given below:

Element Code	Definition
1001	Discrete Kirchhoff Quadrilateral Plate Bending Element
1003	A.Ibrahimbegovic's Thick Plate Bending Element
1011	4 Node Membrane Element
1012	4 Node Membrane Element with Drilling Dof
1201	Macro Plate Bending Element
1202	Macro Membrane Element
1203	Macro Thick Plate Element

The form of polygon data block is shown below:



POLYGONS
nPoly=1
1 C=1,2,3,4 M=1 eType=1001

Figure A.2 Polygon data input format

5- JOIST BEAM Data Block

The program "NESONEL" gives the facility of adding stiffeners to two-dimensional elements. For the definition the start, end coordinates and the material property of the joist beam is required.

After defining the number of joist beams by using "nBeam" identifier, the beam information data line has the following form:

BeamId XS=... YS=... ZS=... XE=... YE=... ZE=... M=MaterialId

XS,YS,ZS are the X,Y,Z coordinates of the starting point and XE,YE,ZE are the X,Y,Z coordinates of the end point of the beam respectively. Material Id is used for determining the material property of the beam.

6- MACRO ELEMENT Data Block

NESONEL provides the chance of using macro elements with different number of global nodes, more than one macro element and arbitrary shaped macro elements. For this purpose, the initial geometry and the mesh

densities must be previously defined at NODES, EDGES and POLYGONS data blocks.

First the number of macro elements must be defined by the "nElem" identifier. The macro element information has the following structure.

ElemId P=PolygonI, PolygonJ, ... gNodes=NodeI, NodeJ, ...

The element identification numbers must be in order for the assemblage of panel loads and panel macro elements. A macro element can involve different number of polygons for the shape and global node definitions. The element codes of the polygons used in macro element definitions must be one of the macro element codes. Global nodes are the solution points of the macro element used in the structural system solution. The global nodes must be defined in counter clockwise direction.

An example to this data block is presented below:

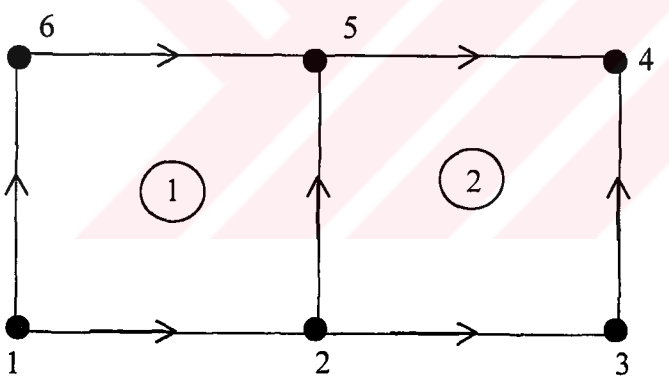


Figure A.3 Macro element input format

4-Node Macro Element:

MACRO ELEMENT
nElem=1
1 P=1,2 gNodes=1,3,4,6

6-Node Macro Element:

MACRO ELEMENT
nElem=1
1 P=1,2 gNodes=1,2,3,4,5,6

7- PANEL JOIST BEAM Data Block

Each macro element can have different number of stiffeners. This data block is used to add joist beams to the macro elements. The second data line is for designating the number of macro elements to be stiffened. The identifier for this data line is "nType". The following line is for the definition of the first data type. Two properties, the macro element ID that the beams belong to and the number of beams must be entered by using the identifiers "nPanel and "nBeam". Then the joist beam data that has the same format with the ones in JOIST BEAM data block must be determined.

The format of this data block is given below:

PANEL JOIST BEAM
nType=1
nPanel=1 nBeam=1
1 XS=... YS=... ZS=... XE=... YE=... ZE=... M=MaterialId } 1st type

8- MATERIAL Data Block

This data block is used to designate the material and section properties of the mechanical elements. The second data line is for the number of different material types and the identifier "nMat" is used for this purpose. The following data fields could be filled either by two-dimensional element or by beam element material properties. They have the following form:

Two-dimensional element:

MaterialId E = e Mu = μ t = th

Beam element:

MaterialId E = e G = g I1 = I_1 I2 = I_2 J = j t = th

where

μ = Poisson ratio

e = Elastic modulus

g = Shear modulus

I_1= Moment of Inertia in Global Y-direction

I_2= Moment of Inertia in Global Z-direction

J = Torsional constant

t = thickness of the element

9- POINT LOAD Data Block

The second line is used for determining the number of point loads by using the identifier "nLoad". The following lines are used to define the coordinates, the direction and the magnitude of the loading. The data line has the following form.

LoadId X=x Y=y Z=z M = F_x,F_y,F_z,M_x,M_y,M_z

x,y,z are the X,Y,Z coordinates and F_x,F_y,F_z,M_x,M_y,M_z are the directions and magnitude of loadings respectively.

10- NODAL LOAD Data Block

A point load could be applied directly to the global nodes. This prevents the time loss while locating the point load. After determining the number of loads, the NodeId and the magnitude should be entered. An example to this data block is shown below:

```
NODAL LOAD
nLoad=1
3  M=0,0,-10
```

11- LINE LOAD Data Block

In order to define a line load, its start and end coordinates must be entered. Format of this data block is:

```
LINE LOAD
nLoad=…
LoadId XS=… YS=… ZS=… XE=…YE=… ZE=… M= Fx,Fy,Fz,Mx,My,Mz
```

12- SURFACE LOAD Data Block

The surface loads are directly applied to the previously defined polygons. They have three different types. Gravity Loads, Pressure Load and Snow Load. They are determined by "Type" identifier. Format of this data block is given below:

```
SURFACE LOAD
nLoad=1
1 Type=1 M=-1
```

PolygonId Type=Icode

Icode	Definiton:
1	Gravity Load
2	Pressure Load
3	Snow Load

13- PANEL LOAD Data Block

The loadings on macro elements must be defined separately, and this data block is used for this purpose.

The second line of this data block determines the number of different loading blocks. The identifier "nType" is used.

For each data blocks, the loading type must be defined. The following line is used for determining the macro element Id and the number of loadings. the other lines have the same structure of the chosen type.

Two examples for this data block is given below:

```
PANEL LOAD
nType=2
POINT LOAD
nPanel=1 nLoad=1
1 X=0.0  Y=0.0  Z=0.0  M=0,0,-10
LINE LOAD
nPanel=1 nLoad=1
1 XS=0.0 YS=0.0 ZS=0.0  XE=2.0  YE=2.0  ZE=0.0  M=0,0,-1
```

```
PANEL LOAD
nType=2
POINT LOAD
nPanel=1 nLoad=1
1 X=0.0  Y=0.0 Z=0.0  M=0,0,-10
POINT LOAD
nPanel=2 nLoad=1
1 X=2.0  Y=2.0 Z=0.0  M=0,0,-10
```

APPENDIX B

BOOCH NOTATION

Static Model Notation

Class

A class is a set of objects that share a common structure and a common behaviour.

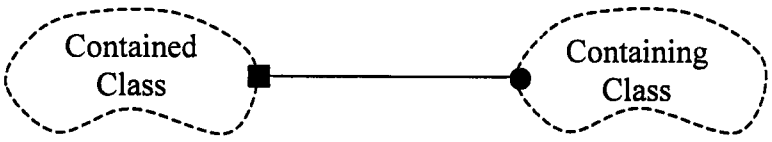
- Class Symbol:

- Abstract Class Symbol:

Class Relationships

- Containment by Value: The lifetime of the object is controlled by the containing object. The containing class has the contained class in its state.

  Symbol:

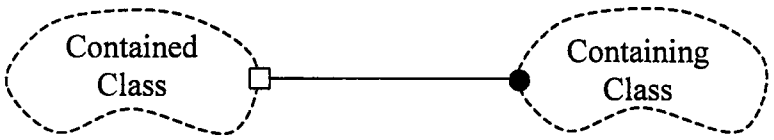


C++ Implementation:

```
Class Element {
private:
List NodeList;
...
};
```

- Containment by reference: The lifetime of the two objects are independent with the restriction that the contained class must exist as long as it is being contained. the containing class has the pointer for the contained class in its state.

  Symbol:



C++ Implementation:

```
Class Element {
   private:
        List NodeList;
   ...
};
```

- Inheritance: Expresses generalisation/specialisation relationship

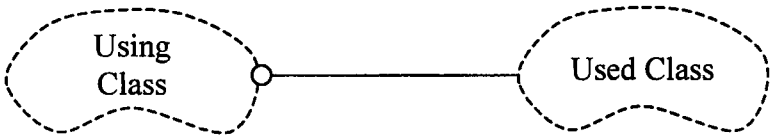
  Symbol:

C++ Implementation

```
Class Element : public MechanicalEntity {
   private:
   ...
};
```

- Uses relationship: Indicates that the used class is somehow known to the used class either being created by the using object, passed to the using object through its interface, or exist in the same lexical scope of the using object.

  Symbol:



C++ Implementation:

```
Class Element {
   private:
   ...
   public:
   void      assemble(Matrix);
 };
```

- Containment with cardinality: A simple class can contain unspecified number of items.

  Symbol: