Title

A Simple and Efficient Finite Element for General Shell Analysis

Author

Kanok-Nukulchai, Worsak

Publication Date

1978

Report no.
UC SESM 78-1

STRUCTURAL ENGINEERING AND STRUCTURAL MECHANICS

A SIMPLE AND EFFICIENT FINITE ELEMENT FOR GENERAL SHELL ANALYSIS

by

WORSAK KANOKNUKULCHAI

JANUARY 1978

DEPARTMENT OF CIVIL ENGINEERING
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA

Division of Structural Engineering and Structural Mechanics
Department of Civil Engineering, University of California
Berkeley, California.

Report No. 78-1

A SIMPLE AND EFFICIENT FINITE ELEMENT FOR GENERAL SHELL ANALYSIS

by

WORSAK KANOK-NUKULCHAI

Graduate student and research assistant

This report was prepared under contract number

DOT - HS - 6 - 01443

Faculty Investigators

J.L. SACKMAN
R.L. TAYLOR

sponsored by

National Highway Traffic Safety Administration
Department of Transportation
Washington, D.C.

January 1978

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

This report was prepared under the contract no. DOT-HS-6-01443; sponsored by the National Highway Traffic Safety Administration of the Department of Transportation, administrated by Dr. L. Ovenshire. The project is under the supervision of Professors R.L. Taylor and J.L. Sackman of the Department of Civil Engineering, University of California at Berkeley, and Professor T.J.R. Hughes of the Division of Engineering and Applied Science, California Institute of Technology, Pasadena, California.

Acknowledement is also extended to his colleagues, A. Curnier, J. Rollins and Dr. H. Hilber for their continued interest.

A SIMPLE AND EFFICIENT FINITE ELEMENT FOR GENERAL SHELL ANALYSIS.

WORSAK KANOK-NUKULCHAI

Division of Structural Engineering and Structural Mechanics,
Department of Civil Engineering, University of California
Berkeley, California, U.S.A.

ABSTRACT

A simple, efficient and versatile finite element is introduced for shell applications. The element is developed based on a degeneration concept, in which the displacements and rotations of the shell mid-surface are independent variables. Bilinear functions are employed in conjunction with a reduced integration for the transverse shear energy. Several examples are tested to demonstrate the effectiveness and versatility of the element. The numerical results indicate that the shell element performs accurately for both thick and thin shell situations.

A SIMPLE AND EFFICIENT FINITE ELEMENT FOR GENERAL SHELL ANALYSIS.

WORSAK KANOK-NUKULCHAI

Division of Structural Engineering and Structural Mechanics,
Department of Civil Engineering,University of California
Berkeley, California, U.S.A.

INTRODUCTION

The importance of shell structures and their generic analysis complexities has naturally lead to a reliance on the finite element method for the solution to many types of shell problems. This, of course, requires a finite element representation of the shell behavior and over the last 20 years many elements have been developed and employed in a multitude of programs. A history of shell elements is traced in [1,2] and only a brief summary will be included here. The paper's main theme is the development of a new shell element: the (B)ilinear (D)egenerated (S)hell. The BDS element attempts to rectify problems inherent to most shell elements: (a) the limited scope of problems which can be solved and, (b) the high formulation cost of computing the element stiffness. The former restricts most elements to one class of shells, either thin or thick shells, depending on the parent theory used for developing the element. The element stiffness formulation may not be a cost factor for most linear problems but is of paramount importance in nonlinear formulations. Thus the quest for the ideal shell element that is universally applicable and cheap to use was a motivating force in the present study.

The approach taken in this paper explores an avenue that was successfully exploited by Hughes et.al. [3] for developing a simple bilinear plate element. The element in [3] was based on a one-point quadrature for the tranverse shear strain energy. The solution, despite its simplicity, is surprisingly accurate for both thick and thin plates.

SHELL ELEMENT STRATEGIES

A shell element can be classified as a 3-D continuum, a classical shell, or a degenerated shell. Figure 1 portrays a skeletal outline of these classifications. To complement Figure 1, a brief paragraph will describe the highlights as they relate to the main theme of this paper. The details of previous shell formulations are contained in the references and the reader is referred to them for a detailed understanding of the development of previous shell elements.

3-D Continuum Elements

The three dimensional field equations can be processed to form a 3-D continuum

* Graduate student and research assistant

element, Figure 1. This, of course, produces an element that is ignorant of the usual kinematic constraints, or assumptions, of most shell problems. Popular among these is: the thickness of the shell is small compared to the other dimensions. Coupled to this is that the shell generally carries load via bending and membrane action. The 3-D formulation is now at a distinct disadvantage: several layers of elements or higher order elements are required to portray bending. Thus ecomonic considerations usually curtail the usefullness of this element. In addition, a 3-D element is found to fail at a moderate length to thickness aspect ratio due to displacement locking through the thickness. Wilson [4] suggested the use of incompatible displacement modes to improve the basic 3-D thick shell performance. Dovey [2] followed with a study of both the reduced integration technique and the idea of adding incompatible modes. These modifications do produce the desired effect, a more flexible shell behavior hence better performance. However, the convergence of these modified 3-D elements can not be guaranteed [2].

Classical Shell Elements

The classical shell element is derived by reducing the 3-D field equations to a particular class of shell equations using analytical integration over the thickness while employing shell assumptions. Common assumptions take advantage of one or more facets of the shell geometry such as the rotation of the cross-section is simply the slope of the shell. This applies only when the shell is relatively thin and its shear strain is negligible. As a result the normal to the reference surface remains normal. This is the *Kirchhoff-Love hypothesis.* Since most shell elements of classical type invoke this assumption, its implications will be reviewed. The Kirchhoff-Love hypothesis leads to displacement equations of equilibrium that are a coupled set of two second-order differential equations inplane and a fourth-order differential equation in the transverse direction of the shell. Therefore, a shell element based on this theory needs C^1 continuity, thus, higher order interpolation functions are required than for the 3-D continuum. Nodal variables must include at least three displacements and two derivatives of the transverse displacement. The inplane, membrane, interpolation functions are usually of lower order than the transverse, bending, function. This can create gaps or overlaps between the edges of two non-planar elements. Many shell elements and shell theories also lack the presence of rigid body modes. Despite these shortcomings many elements are reported to work satisfactorily in the linear infinitesimal-displacement regime [4,5,6].

Degenerated Shell Elements

The degeneration concept described in Figure 1 directly discretizes the 3-D field equation in terms of mid-surface nodal variables. This usually employs a shell assumption, i.e., *the straight normal at any point on the mid-surface remains straight.* The formulation also includes the transverse shear effect , thus no Kirchhoff-Love hypothesis is presumed. The equilibrium equations in terms of independent variables (e.g., displacements and rotations) are second-order differential equations, therefore, the elements require only C^0-continuous shape functions.

Many authors [7,8,9,10] who developed shell elements based on this concept obtained unsatisfactory results when these elements are applied in the thin shell regime. The difficulty can be traced to the transverse shear energy which is $O[(L/h)^2]$ higher than the remaining terms, where L/h denotes the element length to thickness aspect ratio. Thus as the thickness approaches zero, the computed shear stiffness completely dominates, and no effect of the bending stiffness remains with a finite computer word length. As a result, the element produces an excessively stiff solution which does not reflect the correct bending behavior. Physically, this phenomenon is known as a *shear-locking* [3].

Several techniques have been studied in an attempt to solve the shear-locking problem. Wempner [10] introduced a *discrete Kirchhoff hypothesis* which enforced the Kirchhoff hypothesis only at the mid-side of the elements. The poor performance of this element was due, presumably, to too many constraints being enforced. The correction employed in [10] was to reduce the number of constraints. All shear energy was suppressed and good results were obtained. The element with no shear energy reverts to being good for only thin shells hence this solution is wanting. In addition the technique used in [10] leads to either unsymmetric coefficient matrices to be solved for the nodal displacements or an increased number of unknowns if Lagrange multiplier methods are used to satisfy the constraints. Zienkiewicz et.al. [11] retained the transverse shear energy but used reduced integration for quadratic and cubic isoparametric serendipity elements [12] and obtained good results for some thin and thick shells. In this formulation, the transverse shear strain energy in the thin shell situation is a penalty function for the Kirchhoff-Love hypothesis and requires under integration to avoid the locking effect [13]. For certain boundary conditions, however, this element shows a tendency to lock in thin shell analysis due to the additional constraints imposed by the boundary conditions.

THE BILINEAR DEGENERATED SHELL ELEMENT

The BDS element (Figure 2) evolves from an eight-node three dimensional brick. The mid-surface , enclosed by four straight sides, forms a hyperbolic paraboloid and, as the name implies, the concept of degeneration is used. Due to the simplicity of the bilinear mid-surface geometry and the displacement field, the strain energy can be integrated analytically over the shell thickness. The integration over the reference mid-surface can then be performed numerically. This not only simplifies the derivation but also saves computer effort in formulating the element stiffness. The rigid body modes are present since this is an isoparametric element. The transverse shear strain energy is retained, consequently, the element is applicable to either thick or thin shells. Only one-point numerical integration is used for this shear strain energy at the center of the element to avoid the shear-locking effect. The element, when applied to plates is equivalent to the plate element presented in [3].

Several examples are tested using the BDS element. The results show that the element is capable of performing accurately for both thin and thick shells. The details of the formulation are now presented.

Geometry and Displacement Field

The shell element, Figure 2, is defined by the natural, curvilinear coordinates, {r,s,t} such that a bi-unit cube is uniquely mapped into the shell element. As the thickness of the shell element is being input in the direction normal to the mid-surface at each node, the position at any point in the element can be uniquely given in terms of nodal coordinates and thicknesses as

$$\mathbf{x}(r,s,t) = \sum_{I=1}^{4} N^I(r,s)\{\mathbf{x}^I + \frac{1}{2}\, t\, h^I\, \hat{\mathbf{e}}_{z_3}{}^I\} \tag{1}$$

where $\mathbf{x}^I$ are the coordinates of the mid-surface, h^I the thickness and $\hat{\mathbf{e}}_{z_3}{}^I$ the normal unit vector, all at a node I. The interpolation function N^I is bilinear, i.e.,

$$N^I(r,s) = \frac{1}{4}(1+r^I r)(1+s^I s) \tag{2}$$

where r^I, s^I are the natural coordinates of node I.

At any point on the mid-surface, an orthogonal set of local coordinates, { $\mathbf{z}$ }, is constructed such that its unit vectors are:

$$\hat{\mathbf{e}}_{z_3}(r,s) = \mathbf{x}_{,r}(r,s) \times \mathbf{x}_{,s}(r,s) \;/\; |\mathbf{x}_{,r} \times \mathbf{x}_{,s}| \tag{3}$$

$$\hat{\mathbf{e}}_{z_2}(r,s) = \hat{\mathbf{e}}_{z_3}(r,s) \times \mathbf{x}_{,r}(0,0) \;/\; |\hat{\mathbf{e}}_{z_3} \times \mathbf{x}_{,r}(0,0)| \tag{4}$$

$$\hat{\mathbf{e}}_{z_1}(r,s) = \hat{\mathbf{e}}_{z_2}(r,s) \times \hat{\mathbf{e}}_{z_3}(r,s) \tag{5}$$

The first is normal, while the others are tangent to the midsurface at a point (r,s,0) on the mid-surface (Figure 2).

The displacement vector at any point (r,s,t) in the element can be given in the form:

$$\mathbf{u}(r,s,t) = \sum_{I} N^I(r,s)\{\mathbf{u}^I + \mathbf{u}_\alpha^I(t)\} \tag{6}$$

where $\mathbf{u}^I$ is the nodal displacement vector on the mid-surface, and $\mathbf{u}_\alpha^I$ is the relative nodal displacement vector produced by a normal rotation at the node. The vector $\mathbf{u}_\alpha^I$ is to be expressed explicitly in terms of the rotation vector, α^I, about each of the global axes at the node. Using the shell assumption that *straight normals to the reference mid-surface remain straight after deformation,* the displacement vector based on the local $\mathbf{z}$ coordinates , produced by the normal rotation α' about local axes, is $\mathbf{w}_\alpha$ and expressed by

$$\mathbf{w}_\alpha(t) = \frac{1}{2} t\, h \begin{Bmatrix} \alpha_2' \\ -\alpha_1' \\ 0 \end{Bmatrix} \tag{7}$$

For infinitesimal rotations, the usual transformations from $\mathbf{w}_\alpha^I$ to $\mathbf{u}_\alpha^I$ and α'^I to α^I, in view of (7) lead to

$$\mathbf{u}_\alpha^I(t) = \frac{1}{2}\, t\, h^I \, \Phi^I \alpha^I \tag{8}$$

where

$$\Phi^I = \begin{bmatrix} 0 & \theta_{33}^I & -\theta_{23}^I \\ -\theta_{33}^I & 0 & \theta_{13}^I \\ \theta_{23}^I & -\theta_{13}^I & 0 \end{bmatrix} \tag{9}$$

in which θ_{ij} denotes the direction cosine from global to local coordinates, i.e.,

$$\theta_{ij} = (\hat{\mathbf{e}}_{x_i} . \hat{\mathbf{e}}_{z_j}) \tag{10}$$

Substituting (8) into (6) yields the expression of the displacement vector at any point in the shell element in terms of nodal variables:

$$\mathbf{u}(r,s,t) = \sum_I N^I(r,s)\{\mathbf{u}^I + \frac{1}{2}\, t\, h^I\, \mathbf{\Phi}^I \boldsymbol{\alpha}^I\} \tag{11}$$

Stresses - Strains

At any point in the shell element, the local strain components of interest are

$$\boldsymbol{\epsilon}(r,s,t) \equiv \begin{Bmatrix} \epsilon_{11} \\ \epsilon_{22} \\ \epsilon_{12} \\ \epsilon_{13} \\ \epsilon_{23} \end{Bmatrix} = \begin{Bmatrix} w_1/z_1 \\ w_2/z_2 \\ w_1/z_2 + w_2/z_1 \\ w_1/z_3 + w_3/z_1 \\ w_2/z_3 + w_3/z_2 \end{Bmatrix} \tag{12}$$

The symbol $(.)/z_i$ defines a derivative with respect to local curvilinear coordinates. The derivation of these strain components can be achieved by using the second-order tensor transformation of the displacement gradients, i.e.,

$$\begin{bmatrix} w_1/z_1 & w_2/z_1 & w_3/z_1 \\ w_1/z_2 & w_2/z_2 & w_3/z_2 \\ w_1/z_3 & w_2/z_3 & w_3/z_3 \end{bmatrix} = \mathbf{\Theta}^T \begin{bmatrix} u_{1,x_1} & u_{2,x_1} & u_{3,x_1} \\ u_{1,x_2} & u_{2,x_2} & u_{3,x_2} \\ u_{1,x_3} & u_{2,x_3} & u_{3,x_3} \end{bmatrix} \mathbf{\Theta} \tag{13}$$

where $(.)_{,x_i}$ denotes a partial derivative with respect to the global cartesian coordinates and $\mathbf{\Theta} \equiv [\theta_{ij}]$ is the transformation matrix defined in (10).

For a linear isotropic elastic material, the local stress components are obtained from the usual shell constitutive equations which assume a state of plane stress in each lamina. Accordingly

$$\boldsymbol{\sigma}(r,s,t) = \mathbf{D}\, \boldsymbol{\epsilon}(r,s,t) \tag{14}$$

where $\boldsymbol{\sigma}(r,s,t) \equiv [\sigma_{11}\ \sigma_{22}\ \sigma_{12}\ \sigma_{13}\ \sigma_{23}]^T$ is the local stress vector and

$$\mathbf{D} = \begin{bmatrix} \bar{\lambda}+2\mu & \bar{\lambda} & 0 & 0 & 0 \\ \bar{\lambda} & \bar{\lambda}+2\mu & 0 & 0 & 0 \\ 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & \mu/\kappa_s & 0 \\ 0 & 0 & 0 & 0 & \mu/\kappa_s \end{bmatrix} \tag{15}$$

in which κ_s is a shear deformation correction (1.2 is used for BDS), μ is the shear modulus, and $\bar{\lambda}$ is the plane-stress reduced Lame constant, i.e., $\bar{\lambda} = \nu E/(1-\nu^2)$, where E is the modulus of elasticity and ν is Poisson ratio.

Derivation of the Element Stiffness

The standard form of the element stiffness, as derived by the general finite element procedure, is [20]

$$\mathbf{K}^{IJ} = \int_V (\mathbf{B}^I)^T \mathbf{D} \, \mathbf{B}^J \, dV \tag{16}$$

in which $\mathbf{B}^I$ is the standard strain matrix relating the local strain vector to the nodal variables $\delta^I \equiv [\mathbf{u}^I, \alpha^I]^T$ such that

$$\epsilon(r,s,t) = \sum_I \mathbf{B}^I \delta^I \tag{17}$$

It is convenient to split the stiffness in (16) into two parts : the transverse shear effect and the bending and membrane effects. This will allow use of an appropriate order of numerical integration for each part. Accordingly, let

$$\mathbf{K}^{IJ} = \mathbf{K}_m^{IJ} + \mathbf{K}_s^{IJ} \tag{18}$$

where

$$\mathbf{K}_m^{IJ} = \int_V (\mathbf{B}_m^I)^T \mathbf{D}_m \, \mathbf{B}_m^J \, dV \tag{19}$$

$$\mathbf{K}_s^{IJ} = \int_V (\mathbf{B}_s^I)^T \mathbf{D}_s \, \mathbf{B}_s^J \, dV \tag{20}$$

and $\mathbf{B}_m, \mathbf{B}_s, \mathbf{D}_m, \mathbf{D}_s$ are defined according to

$$\epsilon_m \equiv [\epsilon_{11} \; \epsilon_{22} \; \epsilon_{12}]^T = \sum_I \mathbf{B}_m^I \delta^I \tag{21}$$

$$\epsilon_s \equiv [\epsilon_{13} \; \epsilon_{23}]^T = \sum_I \mathbf{B}_s^I \delta^I \tag{22}$$

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_m & \underline{0} \\ \underline{0} & \mathbf{D}_s \end{bmatrix} \tag{23}$$

To formulate the matrices $\mathbf{B}_m$ and $\mathbf{B}_s$, substitute (11) into (13) and the result into (12) to yield

$$\begin{Bmatrix} \epsilon_m(r,s,t) \\ \epsilon_s(r,s,t) \end{Bmatrix} = \sum_I \begin{bmatrix} \mathbf{B}_{1m}^I & \mathbf{B}_{2m}^I + t \, \mathbf{B}_{3m}^I \\ \mathbf{B}_{1s}^I & \mathbf{B}_{2s}^I + t \, \mathbf{B}_{3s}^I \end{bmatrix} \begin{Bmatrix} \mathbf{u}^I \\ \alpha^I \end{Bmatrix} \tag{24}$$

in which

$$\mathbf{B}_{1m}^{I} = \mathbf{L}_m(N^I)\ \Theta^T \tag{25a}$$

$$\mathbf{B}_{1s}^{I} = \mathbf{L}_s(N^I)\ \Theta^T \tag{25b}$$

$$\mathbf{B}_{2m}^{I} = \frac{1}{2}\ h^I\ N^I\ \mathbf{L}_m(t)\ \Theta^T\ \Phi^I \tag{25c}$$

$$\mathbf{B}_{2s}^{I} = \frac{1}{2}\ h^I\ N^I\ \mathbf{L}_s(t)\ \Theta^T\ \Phi^I \tag{25d}$$

$$\mathbf{B}_{3m}^{I} = \frac{1}{2}\ h^I\ \mathbf{L}_m(N^I)\ \Theta^T\ \Phi^I \tag{25e}$$

$$\mathbf{B}_{3s}^{I} = \frac{1}{2}\ h^I\ \mathbf{L}_s(N^I)\ \Theta^T\ \Phi^I \tag{25f}$$

The operators $\mathbf{L}_m(.)$ and $\mathbf{L}_s(.)$ are gradient operators with respect to local coordinates which are defined as

$$\mathbf{L}_m(f) = \begin{bmatrix} \sum_k \theta_{k1} f_{,x_k} & 0 & 0 \\ 0 & \sum_k \theta_{k2} f_{,x_k} & 0 \\ \sum_k \theta_{k2} f_{,x_k} & \sum_k \theta_{k1} f_{,x_k} & 0 \end{bmatrix} \tag{26}$$

and

$$\mathbf{L}_s(f) = \begin{bmatrix} \sum_k \theta_{k3} f_{,x_k} & 0 & \sum_k \theta_{k1} f_{,x_k} \\ 0 & \sum_k \theta_{k3} f_{,x_k} & \sum_k \theta_{k2} f_{,x_k} \end{bmatrix} \tag{27}$$

Physically, $\mathbf{B}_1^I$ is the strain contribution of the inplane displacements at node I, and $\mathbf{B}_2^I + t\mathbf{B}_3^I$ is the strain contribution of the rotations at node I, which also includes the curvature effect.

Taking into account the way the local coordinates $\{\mathbf{z}\}$ are constructed, it can be shown via an orthogonality condition that

$$\mathbf{L}_m(t) = \underline{0} \tag{28}$$

which results in

$$\mathbf{B}_{2m}^{I} = \underline{0} \tag{29}$$

Substituting (24) into (19), in view of (21) and (29), results in

$$\mathbf{K}_m^{IJ} = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1}\begin{bmatrix} (\mathbf{B}_{1m}^{I})^T \mathbf{D}_m \mathbf{B}_{1m}^{J} & t(\mathbf{B}_{3m}^{I})^T \mathbf{D}_m \mathbf{B}_{1m}^{J} \\ t(\mathbf{B}_{1m}^{I})^T \mathbf{D}_m \mathbf{B}_{3m}^{J} & t^2(\mathbf{B}_{3m}^{I})^T \mathbf{D}_m \mathbf{B}_{3m}^{J} \end{bmatrix} |J(r,s,t)|\, drdsdt \tag{30}$$

where $|J|$ is the determinant of the Jacobian matrix obtained from (1). To be consistent with the shell assumption of straight normals, $|J(r,s,t)|$ can be approximated by $|J(r,s,0)|$. Since $\mathbf{B}_1$ and $\mathbf{B}_3$ are functions of r and s only, the integral in (30) can be analytically integrated with respect to t which leads to

$$\mathbf{K}_m^{IJ} = \int_{-1}^{1}\int_{-1}^{1}\begin{bmatrix} 2(\mathbf{B}_{1m}^{I})^T \mathbf{D}_m \mathbf{B}_{1m}^{J} & \underline{0} \\ \underline{0} & \frac{2}{3}(\mathbf{B}_{3m}^{I})^T \mathbf{D}_m \mathbf{B}_{3m}^{J} \end{bmatrix} |J(r,s,0)|\, drds \tag{31}$$

Similarly, the shear part in (20) can be obtained as

$$\mathbf{K}_s^{IJ} = \int_{-1}^{1}\int_{-1}^{1}\begin{bmatrix} 2(\mathbf{B}_{1s}^{I})^T \mathbf{D}_s \mathbf{B}_{1s}^{J} & 2(\mathbf{B}_{1s}^{I})^T \mathbf{D}_s \mathbf{B}_{2s}^{J} \\ 2(\mathbf{B}_{2s}^{I})^T \mathbf{D}_s \mathbf{B}_{1s}^{J} & 2(\mathbf{B}_{2s}^{I})^T \mathbf{D}_s \mathbf{B}_{2s}^{J} + \frac{2}{3}(\mathbf{B}_{3s}^{I})^T \mathbf{D}_s \mathbf{B}_{3s}^{J} \end{bmatrix} |J(r,s,0)|\, drds \tag{32}$$

The numerical integration is then used to evaluate the remaining integrals in (31) and (32). A two-by-two Gaussian quadrature is employed for the integral in (31) and, as mentioned before, a one-point Gaussian quadrature for the integral in (32). This will ensure singularity of the $\mathbf{K}_s^{IJ}$ which is necessary in order to avoid the shear-locking phenomenon.

Torsional Effect

In using a bilinear-element assemblage to approximate curved shells, convergence is spoiled by a weakly-restrained torsional mode after the mesh reaches some state of refinement. The reason can be explained as follows: The BDS element employs six degrees of freedom per node, however, no stiffness corresponding to the torsional-rotation degree of freedom exists locally in the formulation. All the resistance to this rotation at each node I comes directly from the coupling of the α^I -rotations of the non-planar elements surrounding node I. When the finite-element mesh is refined, angles of the kinks between these elements are close to 2 π and the coupling effect is reduced. This weak coupling only generates a minute amount of stiffness for the torsional rotation. Therefore, any slight disturbance in the generalized load corresponding to this degree of freedom can amplify the torsional mode to an unrealistic amount, which affects the global solution.

This problem is common to many shell elements which use six global degrees of freedom per node. In the past, a fictitious torsional spring was added either locally at the element level, or in some pseudo-normal direction defined at each node. This technique often is found unsatisfactory, especially for a flexible system in which an unrealistic amount of strain energy in the spring can be produced by a rigid body motion.

In a degenerated shell, the rotation of the normal and the mid-surface displacement field are independent. The idea then is to derive an additional constraint between the torsional rotation of the normal, α_3', and the rotation of the mid-surface, $1/2(\partial w_2/\partial z_1 - \partial w_1/\partial z_2)$.

The deviation of the associated rotation from the mid-surface slope, Figure 3(a), is governed by the transverse shear strain energy, i.e.,

$$\pi_s = \kappa_s \, \mu \, h \int_S [\, \alpha_2'(r,s) + \frac{\partial w_3}{\partial z_1}(r,s,0) \,]^2 \, dA \tag{33}$$

In fact, the transverse shear stiffness $\mathbf{K}_s$ of Equation 20 can be obtained directly from the stationary value of π_s. Similarly, the deviation of the torsional rotation of the normal from that of the mid-surface, Figure 3(b), is assumed to have a governing strain energy,

$$\pi_t = \kappa_t \, \mu \, h \int_S [\, \alpha_3'(r,s) - \frac{1}{2} \{ \frac{\partial w_2}{\partial z_1}(r,s,0) - \frac{\partial w_1}{\partial z_2}(r,s,0) \} \,]^2 \, dA \tag{34}$$

where κ_t is a parameter to be determined. If $\kappa_t \, \mu \, h$ is chosen to be large relative to the factor $E \, h^3$ (which appears in the bending energy), Equation 34 will play the role of penalty function and results in the desired constraint:

$$\alpha_3' \approx \frac{1}{2} [\, \frac{\partial w_2}{\partial z_1} - \frac{\partial w_1}{\partial z_2} \,] \tag{35}$$

at the Gauss points. A one-point numerical integration should be used in evaluating the penalty integral in order to avoid an over-constrained situation similar to shear locking.

To derive a torsional stiffness from Equation 34, the local variables are expressed in terms of global nodal variables, $\boldsymbol{\delta}^I$, by bilinear shape functions. This gives Equation 34 in the form,

$$\pi_t = (\boldsymbol{\delta}^I)^T \mathbf{K}_t^{IJ} \boldsymbol{\delta}^J \tag{36}$$

where the torsional stiffness,

$$\mathbf{K}_t^{IJ} = \kappa_t \, \mu \, h \int_{-1}^{1}\int_{-1}^{1} \begin{bmatrix} (\mathbf{R}_m^I)^T \mathbf{R}_m^J & (\mathbf{R}_m^I)^T \mathbf{R}_n^J \\ (\mathbf{R}_n^I)^T \mathbf{R}_m^J & (\mathbf{R}_n^I)^T \mathbf{R}_n^J \end{bmatrix} |J(r,s,0)| \, dr ds \tag{37}$$

and

$$\mathbf{R}_m^I = \frac{1}{2} [\sum_k \theta_{k2} N^I_{,x_k} \quad -\sum_k \theta_{k1} N^I_{,x_k} \quad 0] \, \Theta^T \tag{38}$$

$$\mathbf{R}_n^I = N^I \, [\theta_{13} \quad \theta_{23} \quad \theta_{33}] \tag{39}$$

Since two penalty functions are included in the thin shell situation, the penalty factors, $\kappa_s \, \mu \, h$ and $\kappa_t \, \mu \, h$, should have magnitude of the same order. The result displayed in Figure 5 indicates that the converged solution is insensitive to κ_t, as long as κ_t is large enough (>0.1) to sufficiently restrain the troublesome torsional modes. This insensitivity demonstrates that the addition of a torsional stiffness will not degrade the behavior of the system after the torsional effect is deleted.

All numerical examples in this study employ a value of $\kappa_t = 10$. It should be noted that in example 2 when the real hyperbolic paraboloid shell geometry is exactly represented, an identical solution is obtained with or without the torsional stiffness.

NUMERICAL EXAMPLES

To demonstrate the effectiveness and the versatility of the element, several examples of different shell situations are analyzed.

Thin cylindrical shell roof: The thin cylindrical shell in Figure 4 is tested with meshes of 4,16 and 64 elements. The shell is supported at both ends on rigid diaphragms. This example shows a situation where both membrane and bending actions are significant.

Initially the problem is solved without the addition of a torsional stiffness $\mathbf{K}_t$. The solution fails to converge as is evident in Figure 6, graph 8. A parameter study of κ_t is then carried out for both thick and thin shells, using different element meshes. The results are displayed in Figure 5. The adding of the torsional stiffness with sufficient amount of κ_t to restrain the free torsional mode produces a unique and converged solution.

The convergence characteristics are compared with other elements in Figure 6. The present solution converges to a complete (non-shallow) shell solution. This is also true for many of the shell elements which possess no geometric slope continuity at the element boundaries [23]. The BDS element proves to be highly competitive ---- only some of the higher order elements perform better for the same number of degrees of freedom. The bilinear element, however, requires much less time to formulate due to its simplicity.

Figures 7 and 8 compare displacements to shallow-shell exact solutions. The corresponding deep-shell solutions are not available. The stress-resultants are also shown to be very accurate in Figure 9.

Clamped hyperbolic paraboloid shell: In this example, Figure 10, the element assemblage exactly represents the real shell geometry. The shell is subjected to a uniform normal pressure. The deflection along the center is plotted in Figure 11 and compared to the exact solution reported in [19]. Figure 12 shows the rapid convergence of the solution compared to the relatively slow convergence in [14]. In that reference, the element was specially designed for the shallow hypar shell in which a cubic transverse displacement and linear inplane displacements were used.

Thick cylindrical shell: The problem consists of a long cylindrical shell of uniform thickness, Figure 13, resting on a line suport and subjected to gravity loading. Figure 14 shows the deformed mid-surface which checks well with the simple ring analytical solution. In comparison solutions obtained by the 2-D plane-strain continuum element (Q-4) and the present element but with exact numerical integration are shown. As expected, they are both too stiff.

Arch dam: This example provides a good test of the element in the case of an arbitrary , thick shell with varying thickness. Figure 15 shows a doubly curved arch dam known to be type 5 [15,16]. Details of the geometry can be found in [15] and [16]. In Figure 16, the down stream displacement is plotted and compared with the solutions from a thin shell element, a 3-D continuum element, and a higher-order element [12]

It is remarkable to note that accurate results are attainable with a very crude mesh (16 elements) using this low order element, and are comparable to those of the finer mesh (32 elements) using the higher-order element of [12].

Curved box girder: The last example shows a more complex structure of a curved box girder as detailed in Figure 17. The present element is used for the flanges, curved webs and diaphragm of the box girder. For webs, a 1-by-2 Gaussian quadrature, Figure 18, is employed for the inplane μ -energy term to avoid the poor beam-bending behavior which is analogous to the transverse shear-locking in the shell situation. The 2-point integration along the web thickness is necessary to include the equivalent effect of the analytical integration in a classical beam. The diaphragm is treated as a membrane by deleting the bending stiffness.

The vertical deflection and normal stresses in both flanges are plotted in Figures 19,20 and 21. The solutions are compared with both the experiment and the higher-order finite-element solutions reported in [17]. The present element proves to perform surprisingly well in all aspects.

CONCLUSIONS

The attractiveness of the BDS element is attributed to its efficiency, effectiveness and versatility . The simplest element geometry is chosen so that the element can serve as a convenient basis for unlimited forms of shell geometry. The degeneration concept, coupled with a reduced integration technique, produces a shell element which performs accurately in both thick and thin shell situations. With slight modification, the element is also capable of analyzing other types of thin structures, such as a curved web of the curved box girder.

REFERENCES

[1] S.F. Pawsey, 'The analysis of moderately thick to thin shells by the finite element method,' SESM Report No. 70-12, Dept. of Civil Engineering, U.C. Berkeley, August 1970.

[2] H.H. Dovey, 'Extension of three dimensional analysis to shell structures using the finite element idealiztion,' SESM Report No. 74-2, Dept. of Civil Engineering, U.C. Berkeley, January 1974.

[3] T.J.R. Hughes, R.L. Taylor and W. Kanoknukulchai,' A simple and efficient finite element for plate bending', *Int.J.num.Meth.Engng.*, 11,1529-1543(1977).

[4] E.L. Wilson, R.L. Taylor, W.P. Doherty and J. Ghaboussi,' Incompatible displacement models', *Numerical and Computer Methods in Structural Mechanics,* ed. S.J. Fenves et.al., Academic Press, New York (1973).

[5] R.W. Clough and R.J. Johnson,'A finite element approximation for the analysis of thin shell', *Int.J.Solids Struct.,4,43-60(1968).*

[6] S.W. Key and Z.E. Beisinger,'The analysis of thin shells with transverse shear strains by the finite element method', *Proc. Conf.Matrix Meth.Struct.Mech. Wright-Patterson Air Force Base, Ohio,* AFFDL-TR-68-150, 667-710(1968).

[7] B.M. Irons and K.J. Draper,'Inadequecy of nodal connections in stiffness solution for plate bending', *AIAA J.* 3,5,965(1965).

[8] R.J. Melosh,'A flat triangular shell element stiffness matrix', *Proc.Conf.Matrix Meth.Struct.Mech., Wright-Patterson Air Force Base, Ohio* ,AFFDL-TR-66-80, 503-514(1965).

[9] S. Utku, 'Stiffness matrices for thin triangular elements of non-zero Gaussian curvature', *AIAA J.* ,5,9,1659-1667(1967).

[10] G.A. Wempner, J.T. Oden and D.A. Kross, 'Finite-element analysis of thin shell', *J.Engng.Mech.Div.* ASCE 94,EM6,1273-1293(1968).

[11] O.C. Zienkiewicz, R.L. Taylor and J.M. Too, 'Reduced integration technique in general analysis of plates and shells', *Int.J.num.Meth.Engng* ,3,275-290(1971).

[12] S. Ahmad, B.M. Irons and O.C. Zienkiewicz, 'Analysis of thick and thin shell structures by curved finite elements', *Int.J.num.Meth.Engng.* ,2,419-451(1970).

[13] O.C. Zienkiewicz and E. Hinton, 'Reduced integration, function smoothing and non-conformity in finite element analysis', *J.Franklyn Inst.* 302,443-461(1976).

[14] J.J. Connor and C. Brebbia, 'Stiffness matrix for shallow rectangular shell element', *J.Engng.Mech.Div.* ASCE 93,EM5,43-65(1967).

[15] J. Ergataudis, B.M. Irons and O.C. Zienkiewicz, ' Three-dimensional analysis of arch dams and their foundations', *Proc.Symp.Arch dams* Inst. civ. Engng., London, 37-50(1968).

[16] O.C. Zienkiewicz, C.J. Parekh and I.P. King, 'Arc dams analyzed by a linear finite element solution program', *Proc.Symp.Arch dams* Inst.civ.Engng.,London,19-23(1968).

[17] A.R.M. Fam and C. Turkstra,' Model study of horizontally curved box girder', *J.Engng.Str.Div.* ,ASCE 102,ST5,1097-1108(1976).

[18] A.C. Scordelis and K.S. Lo, 'Computer analysis of cylindrical shells', *ACI J.* 61,1,539-561(1964).

[19] S. Chetty and H. Tottenham, 'An investigation into the bending analysis of hyperbolic paraboloid shells', *Indian Concrete Journal* ,July,1964.

[20] O.C. Zienkiewicz, *The Finite Element Method,* McGraw-Hill, London (1977).

[21] G.R. Cowper, G.M. Lindberg and M.D. Olson, 'A shallow shell finite element of triangular shape', *Int.J. Solids Struct.* 6, 1133-1156 (1970).

[22] R.L. Taylor, *private communications.*

[23] K. Forsberg, *An Evaluation of Finite Difference and Finite Element Techniques for Analysis of General Shells,* Symposium on high speed computing for elastic structures, IUTAM, Mechanics, Liege, 1970.

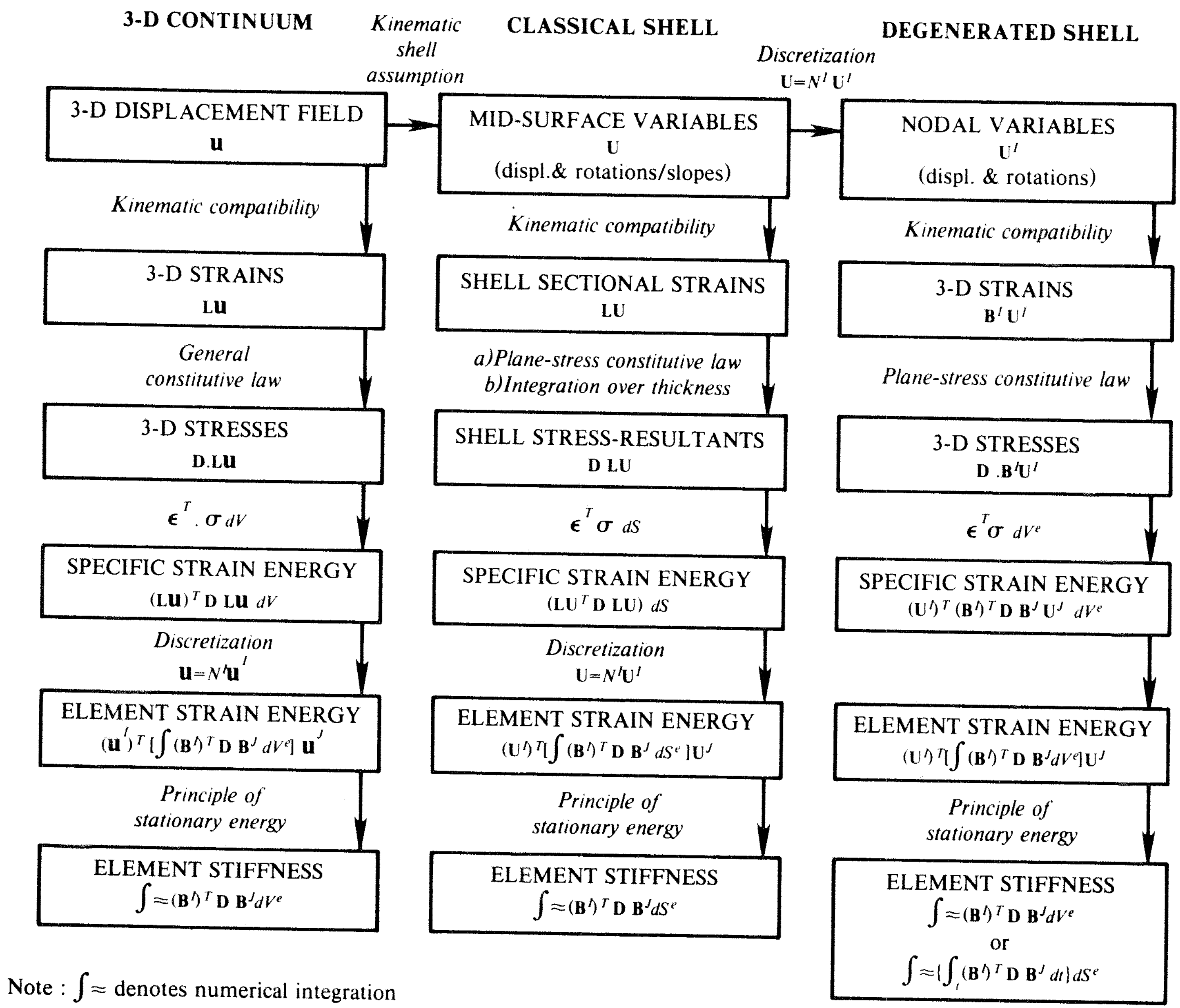


Figure 1. Strategies of shell element derivation.

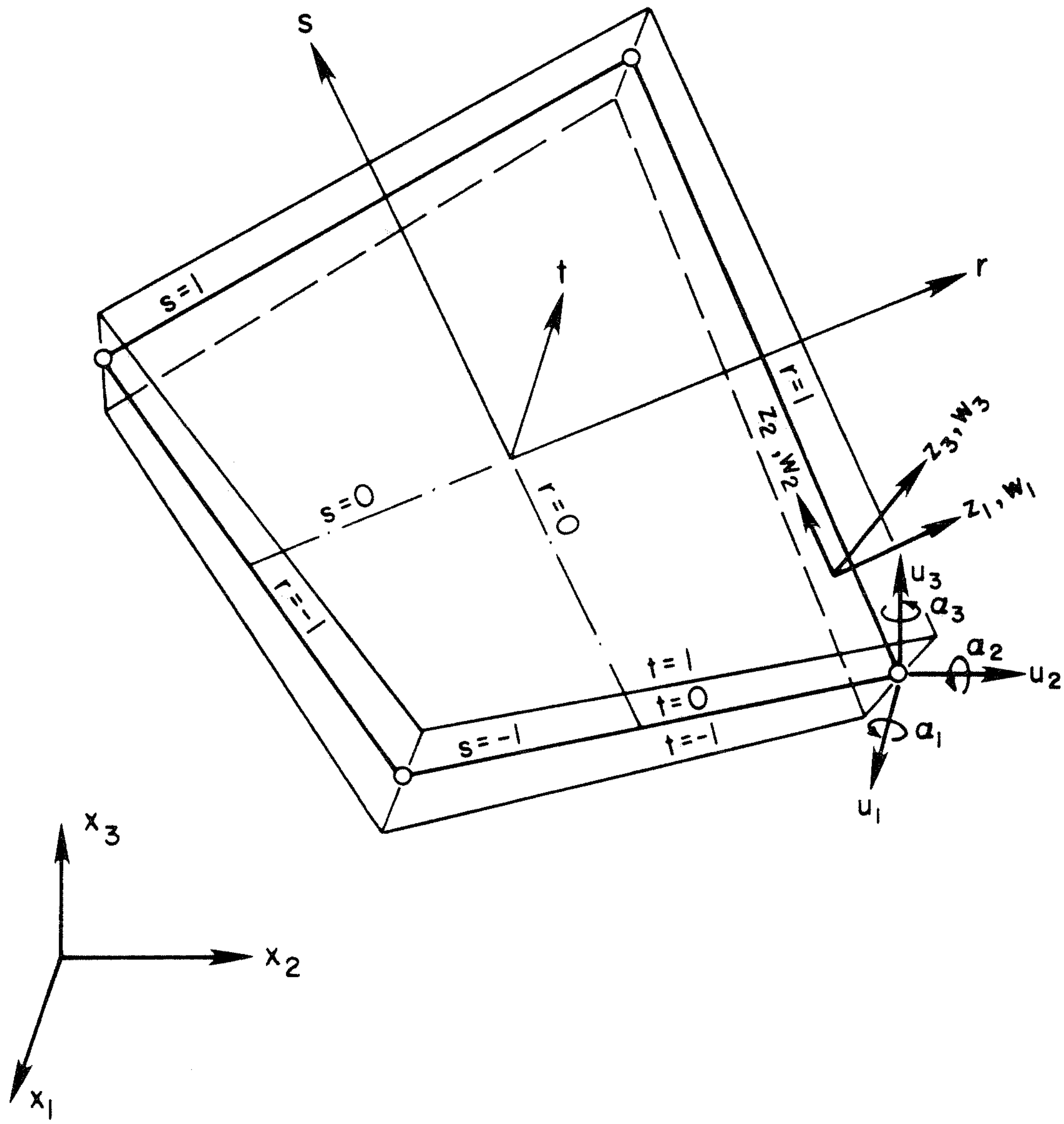


Figure 2. Bilinear shell element.

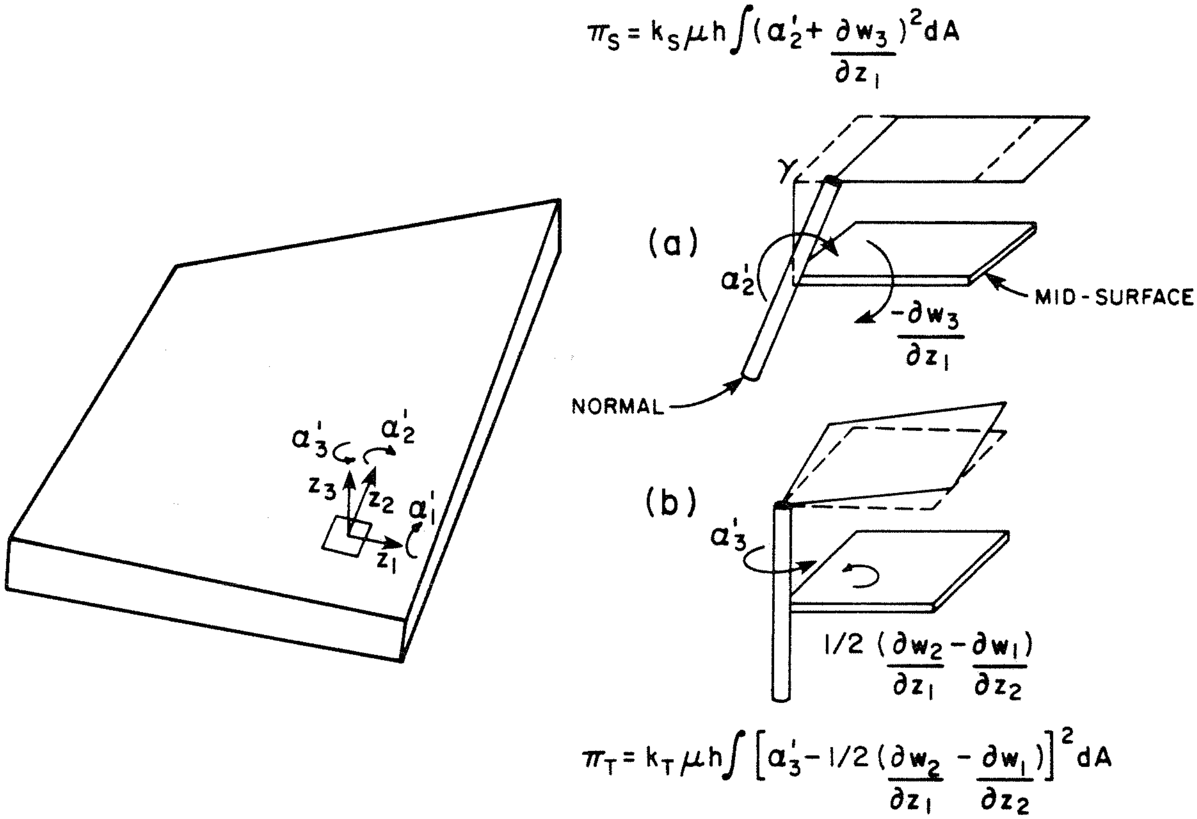


Figure 3. Penalty functions: (a) transverse shear (b) torsion.

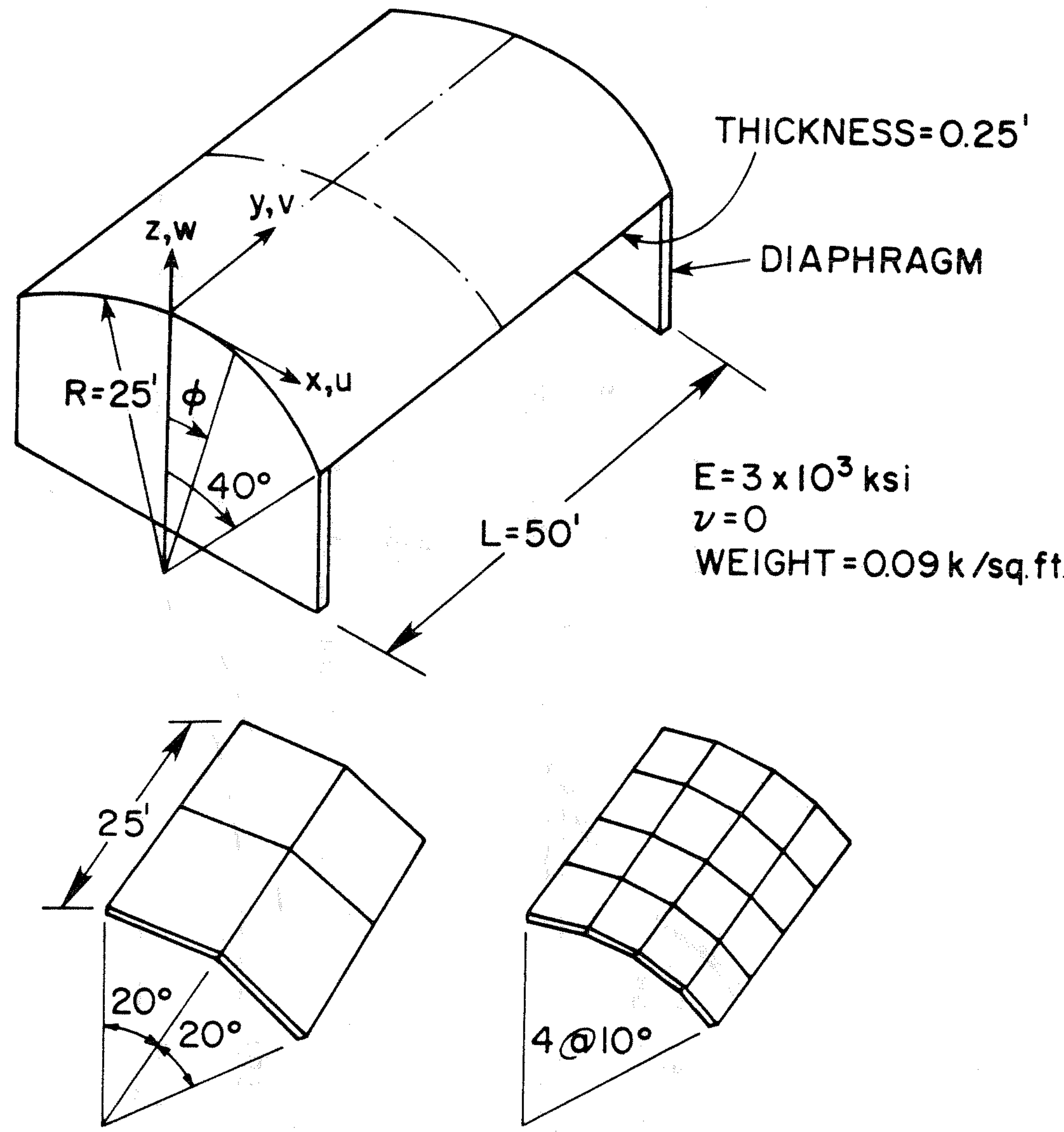


Figure 4. Thin cylindrical shell. Geometry and meshes of 4 and 16 elements.

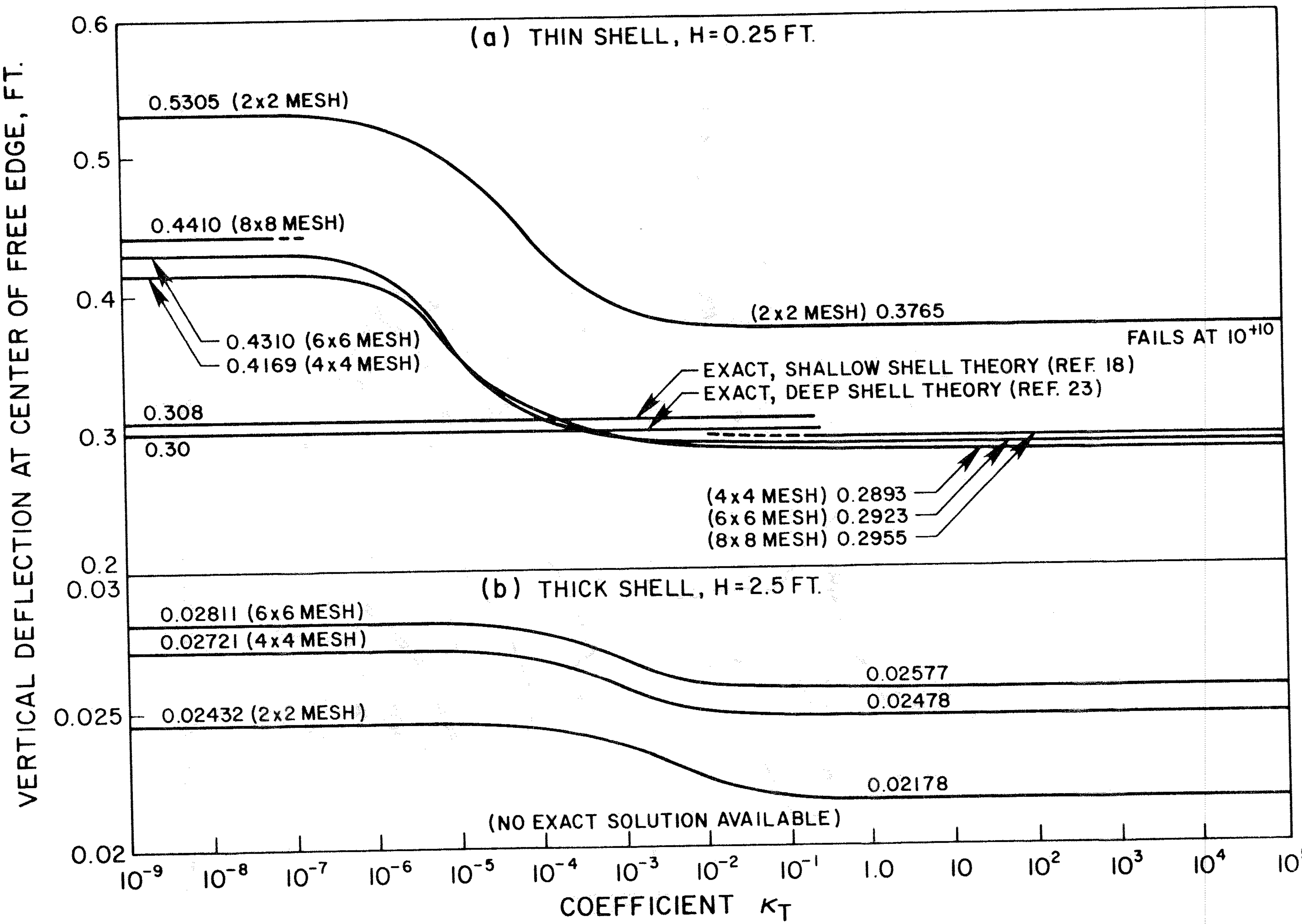


Figure 5. Effect of torsional stiffness to solutions of thin and thick cylindrical shell roofs.

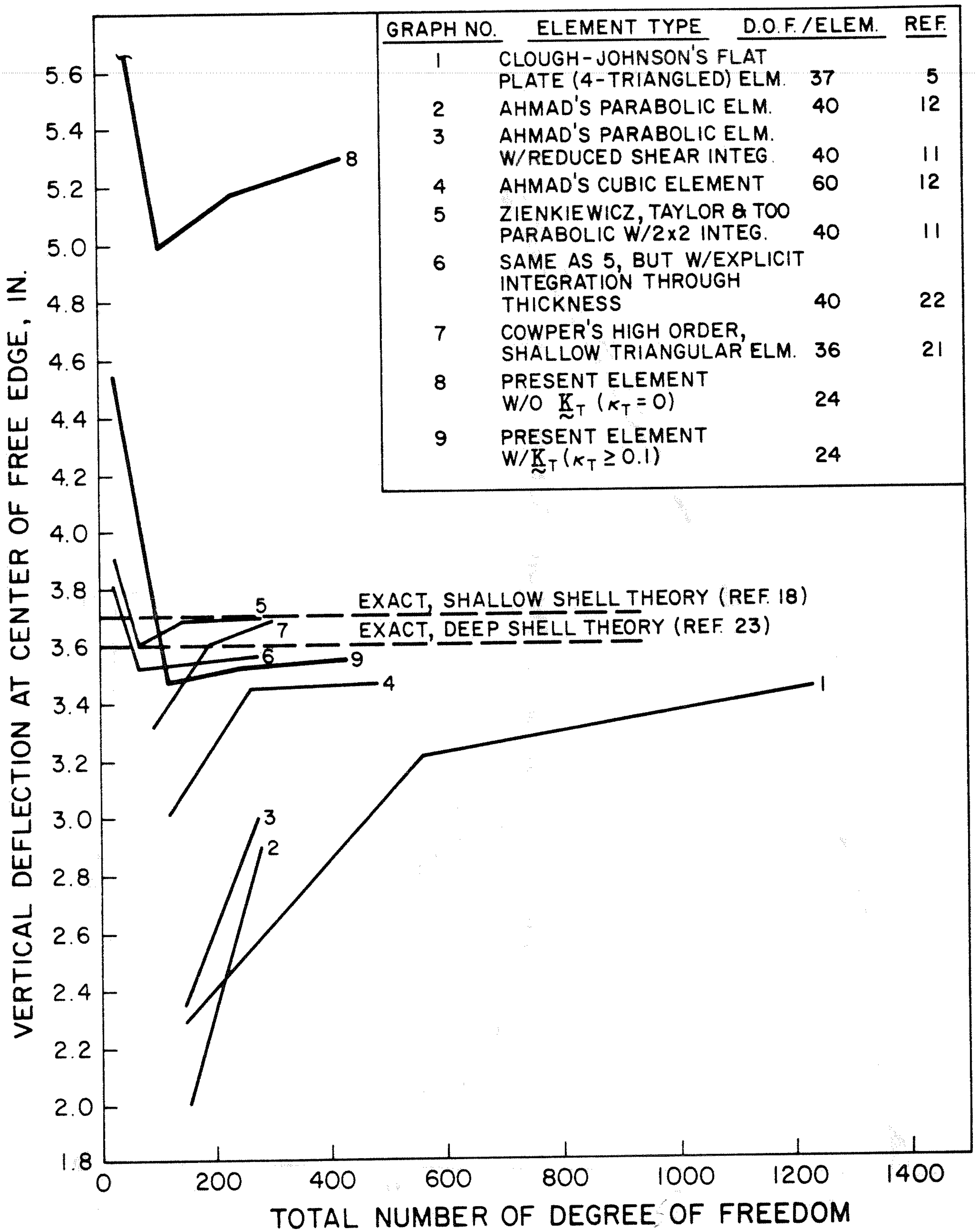


Figure 6. Convergence of deflection versus degrees of freedom, thin cylindrical shell roofs.

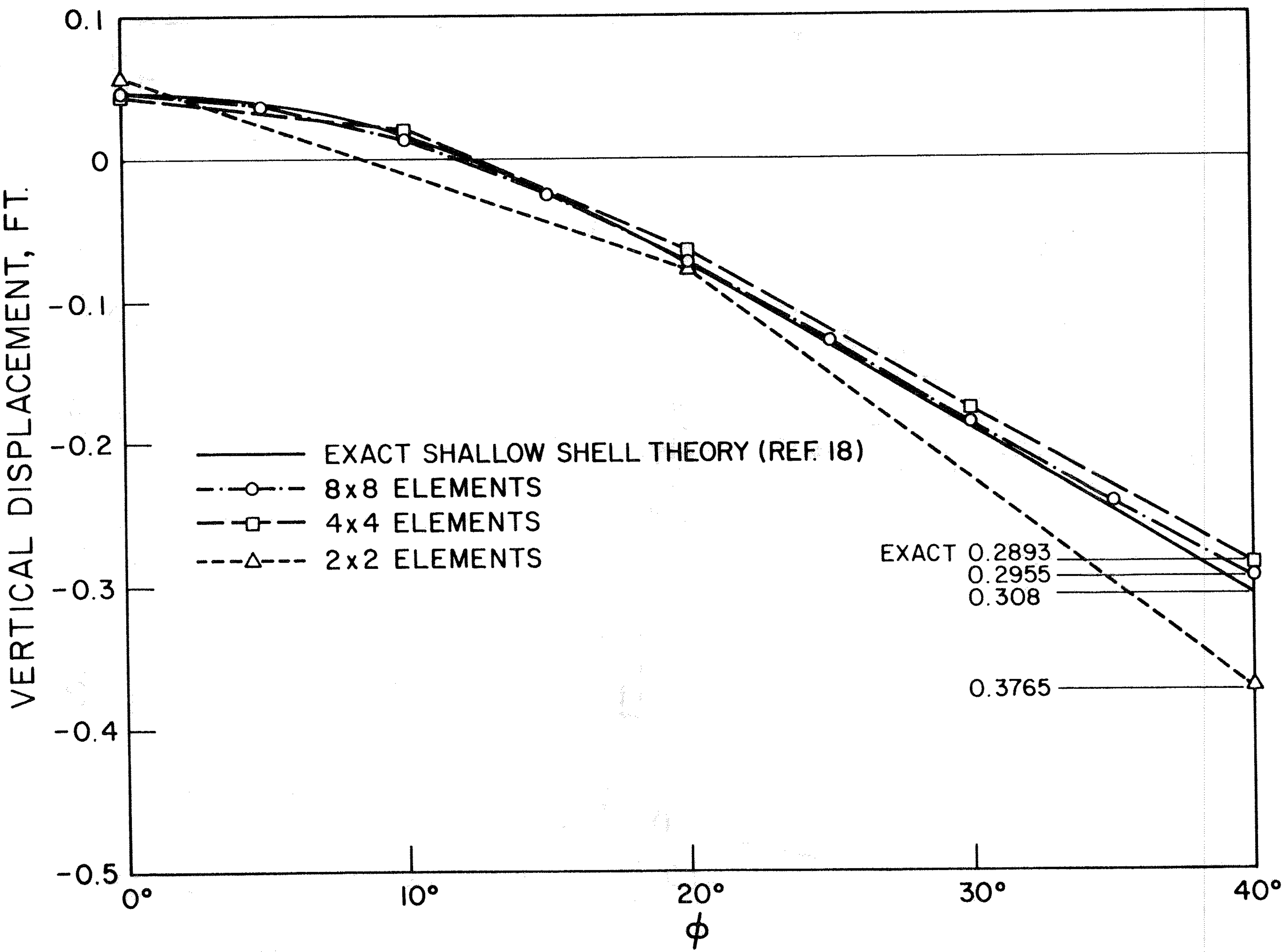


Figure 7. Cylindrical shell roof, vertical displacement at y = 1/2.

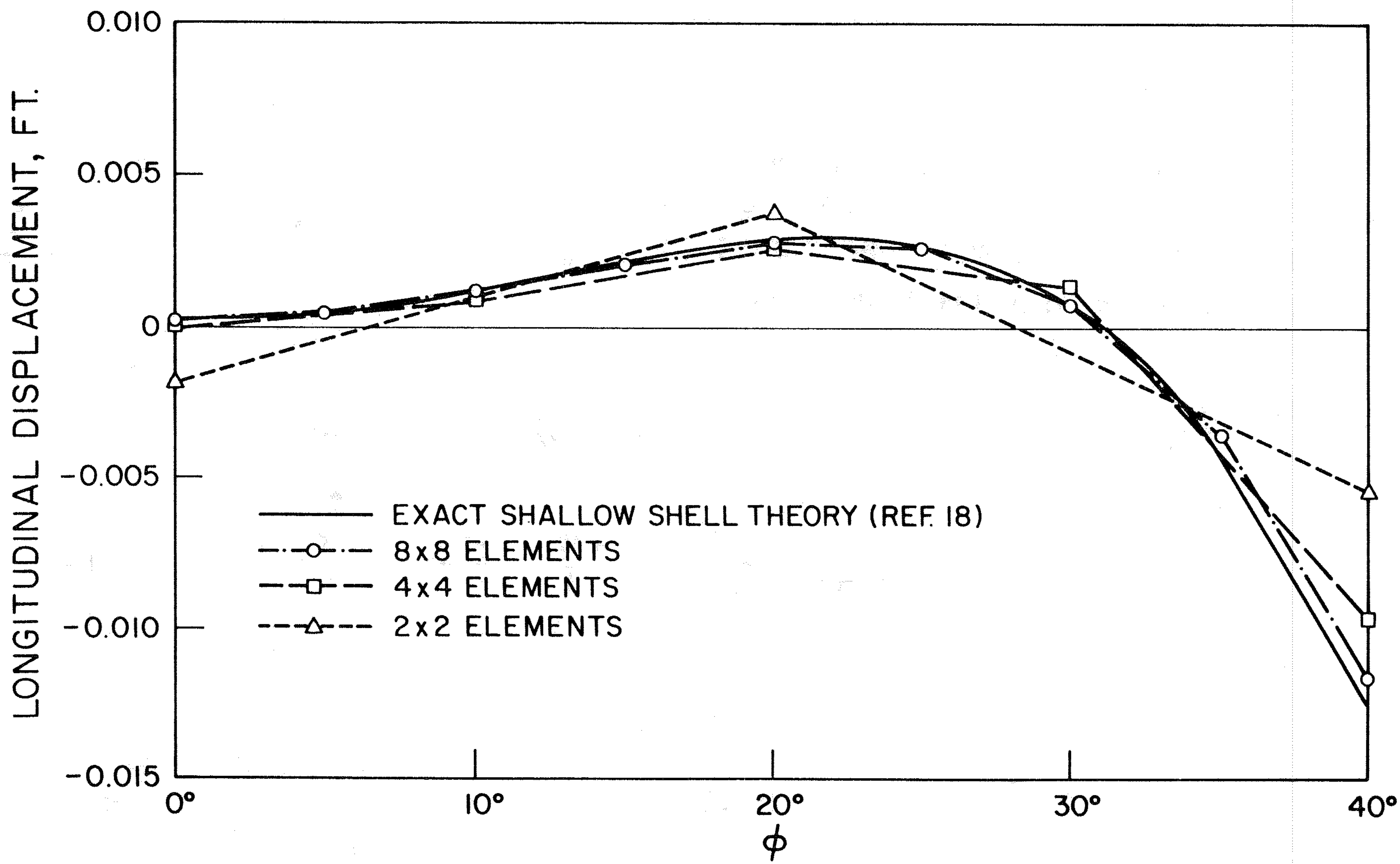


Figure 8. Cylindrical shell roof, longitudinal displacement at diaphragm (y = 0).

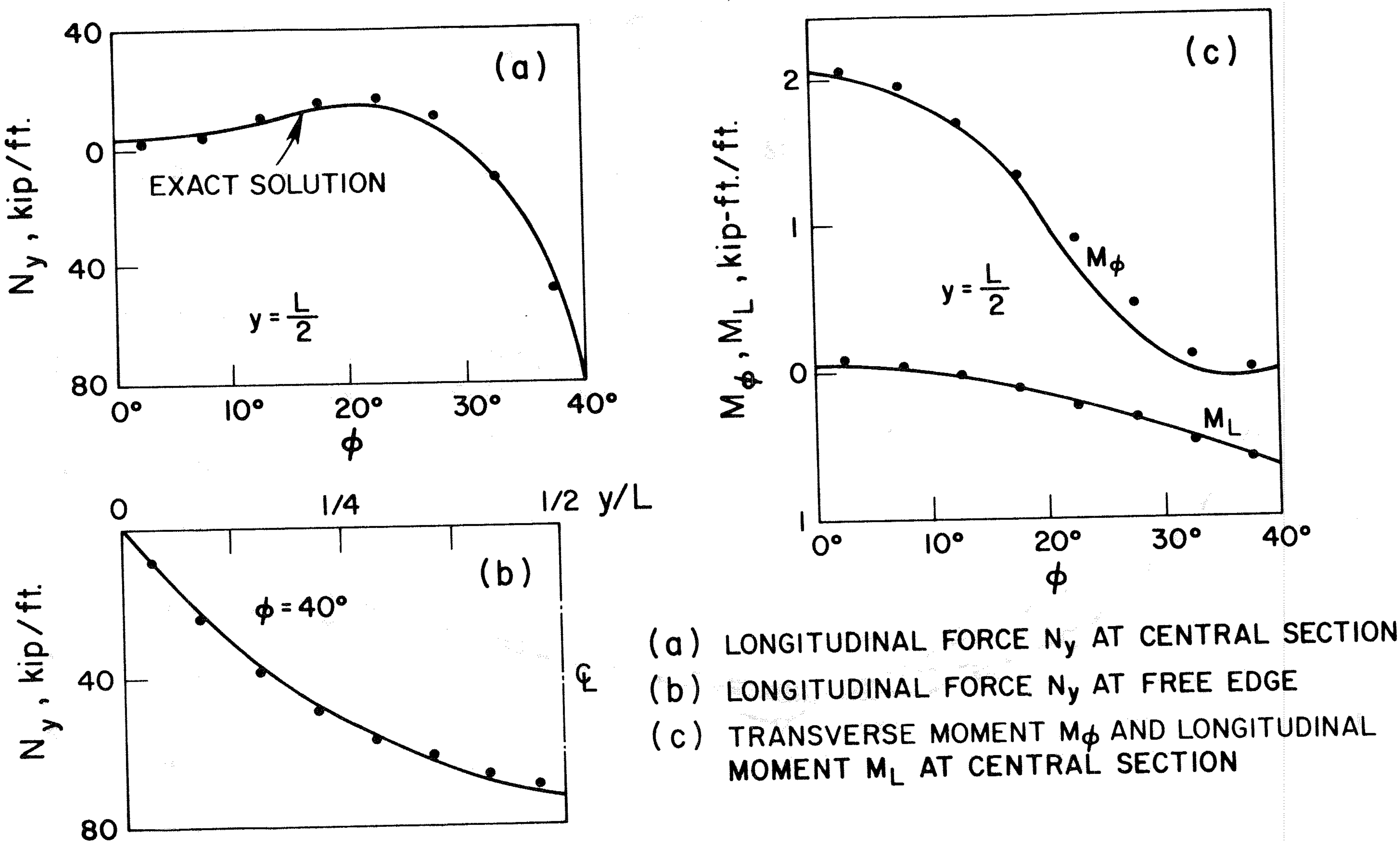


Figure 9. Stress resultants for 8-by-8 element mesh. All stresses are obtained by the interpolation of values at Gauss points.

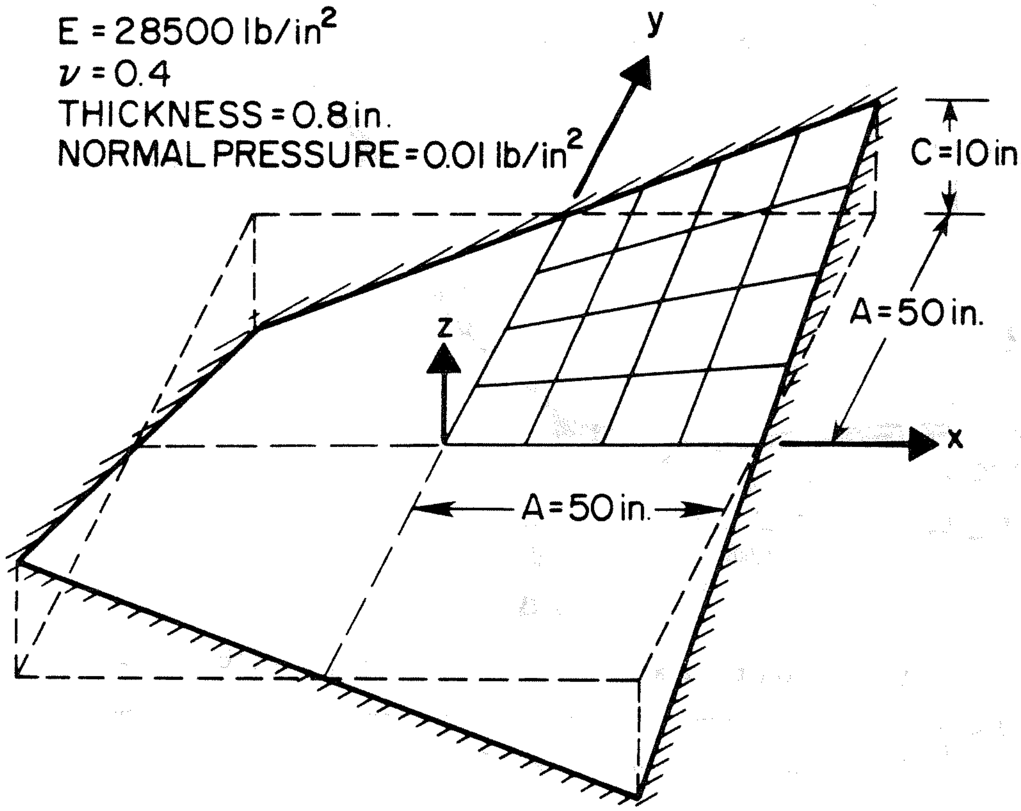


Figure 10. Clamped hyperbolic shell.

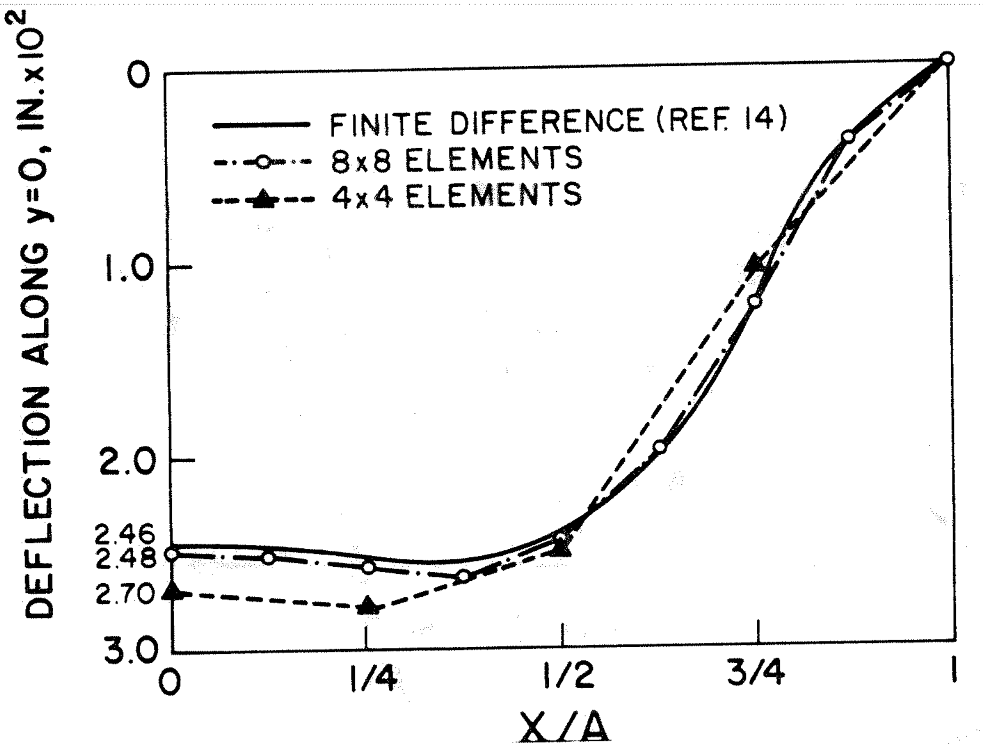


Figure 11. Deflection at center line of hyperbolic shell.

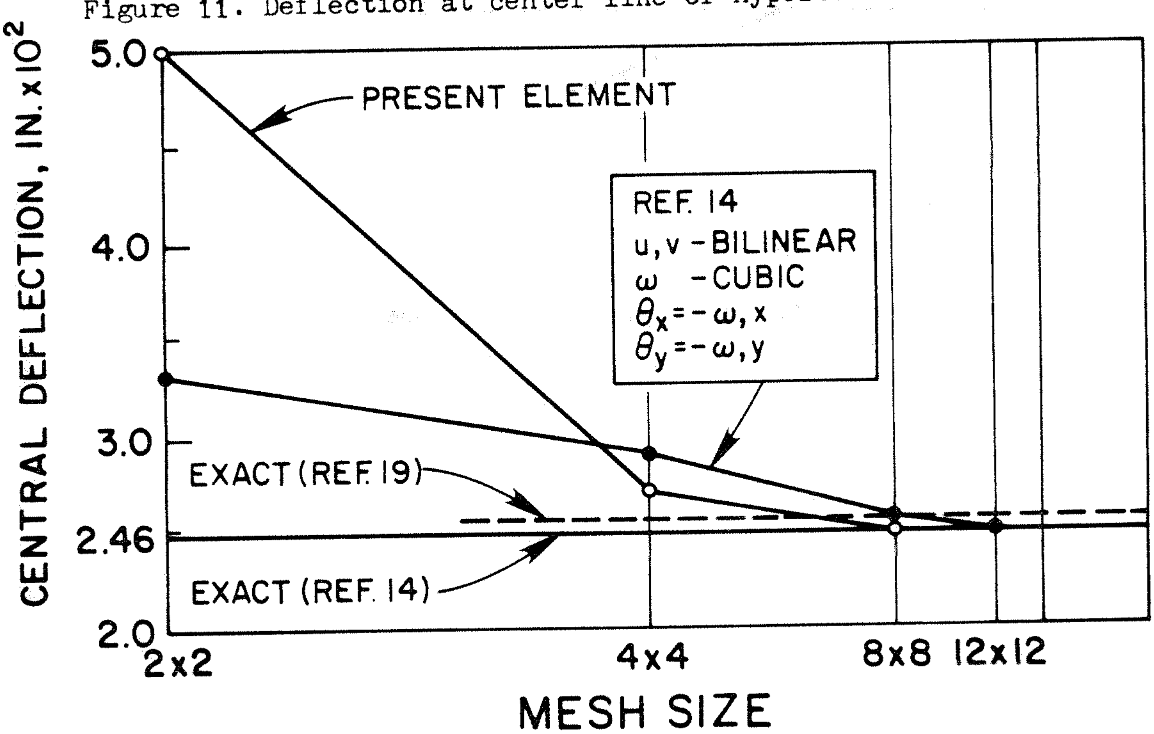


Figure 12. Clamped hyperbolic shell, convergence of central deflection.

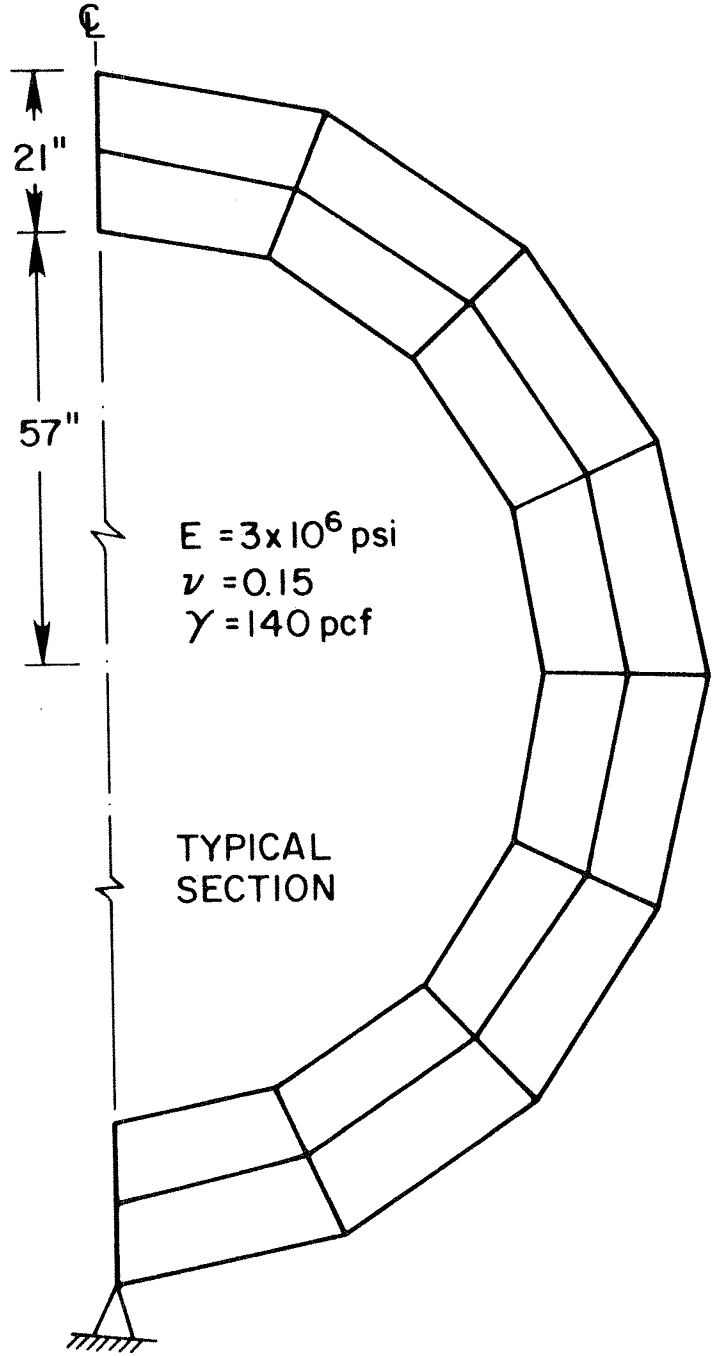


Figure 13. Long cylindrical thick shell on line support.

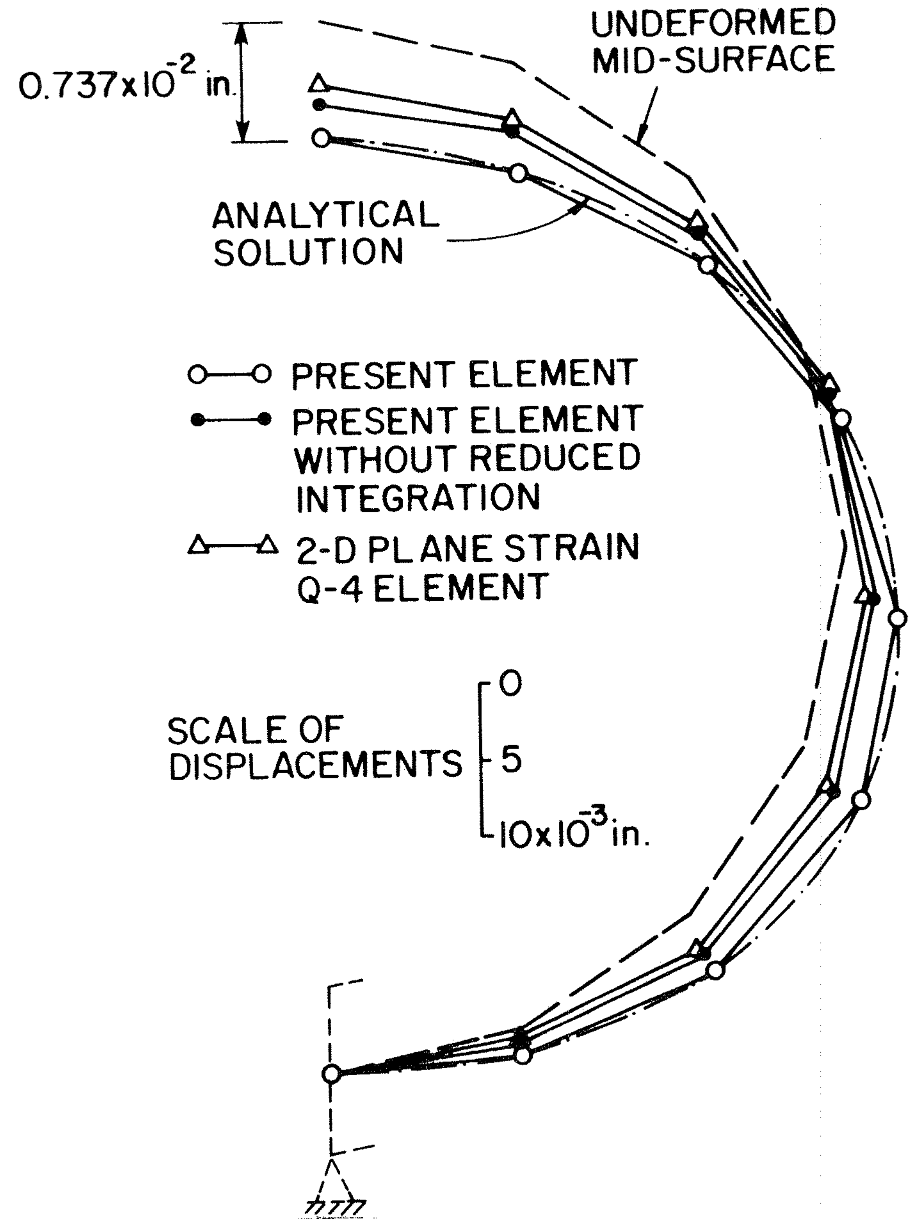


Figure 14. Deformed mid-surface of cylindrical shell due to gravity loading.

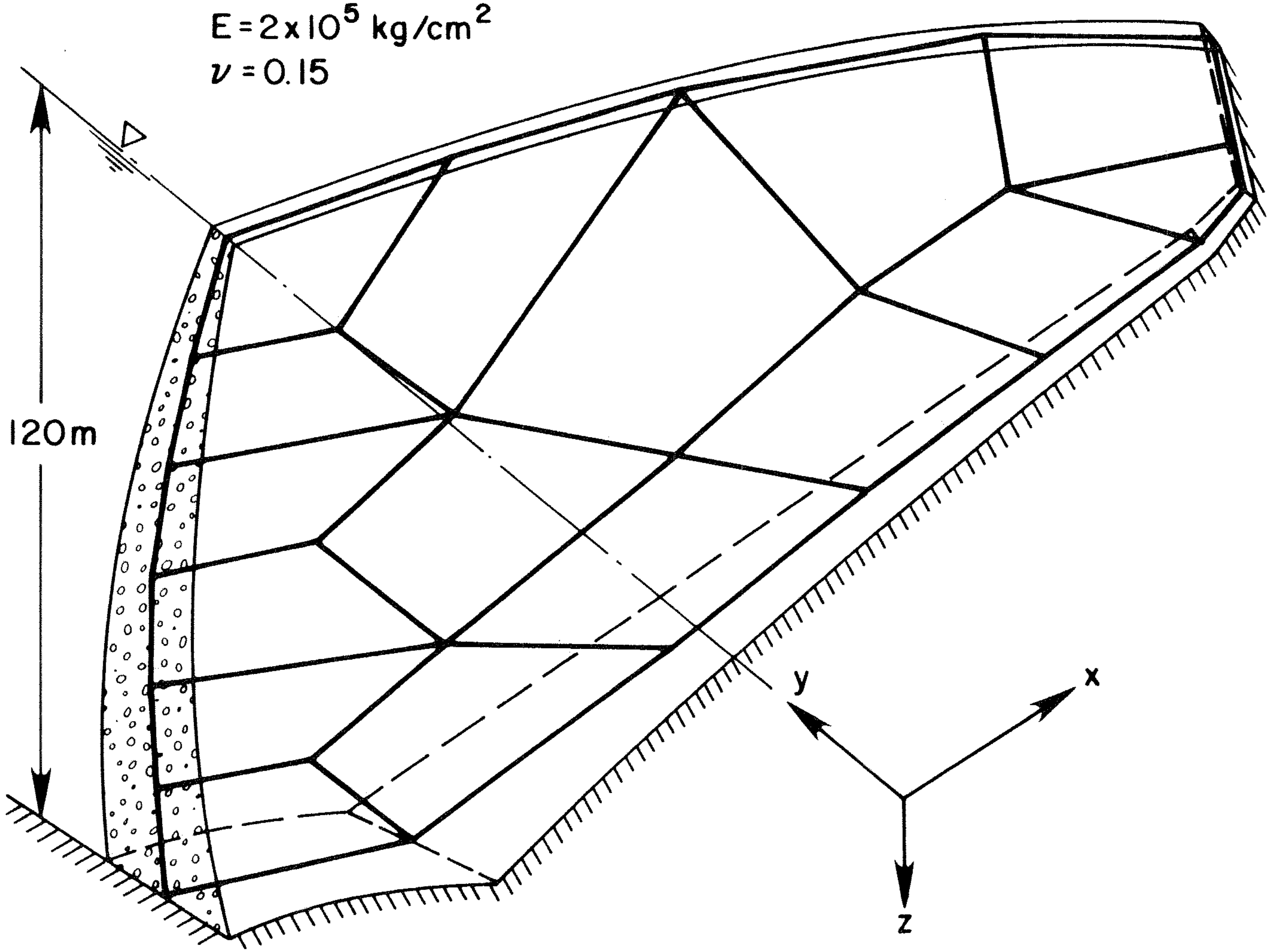


Figure 15. Arch dam type 5, 16-element mesh.

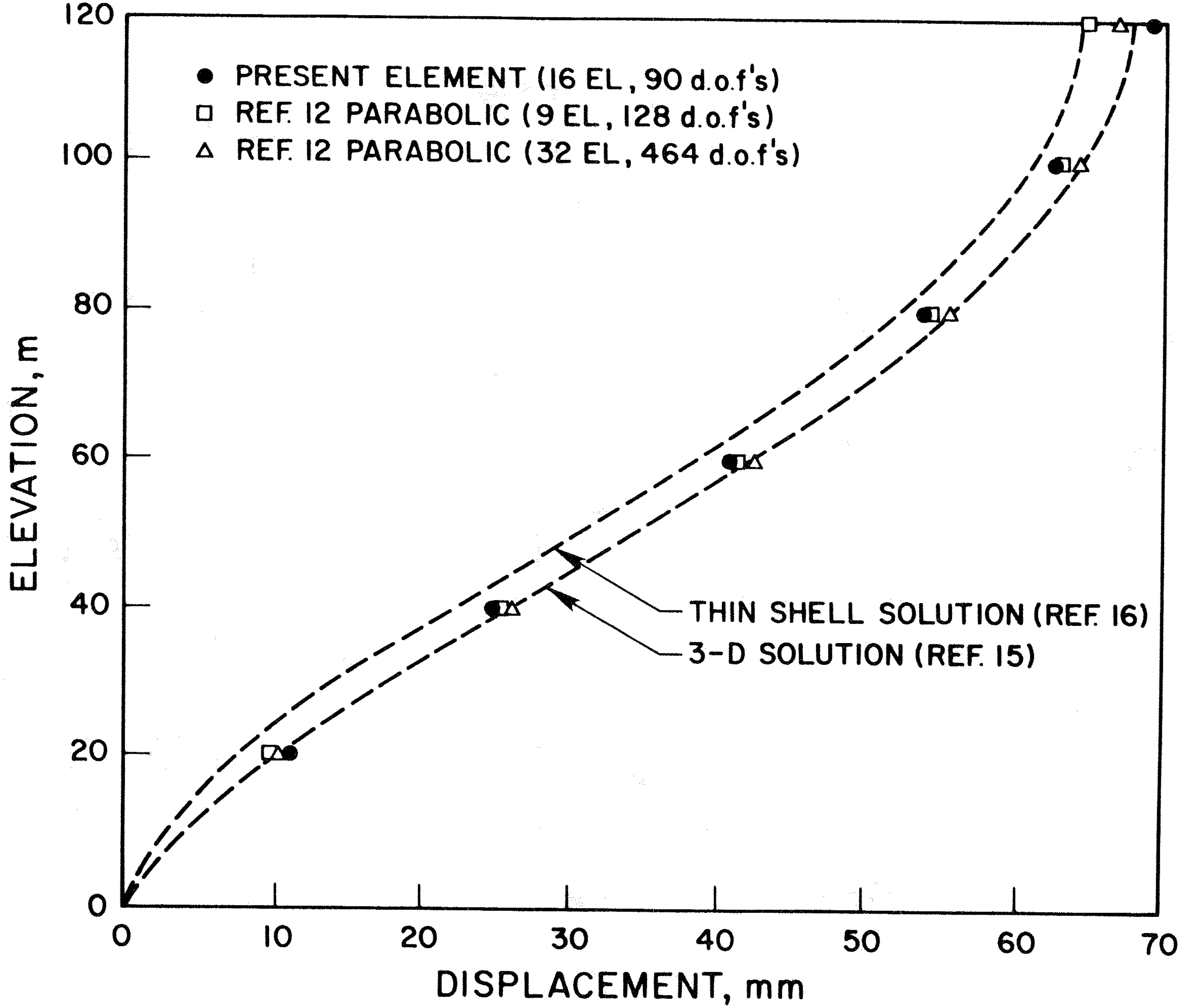


Figure 16. Downstream displacements on crown section of dam type 5.

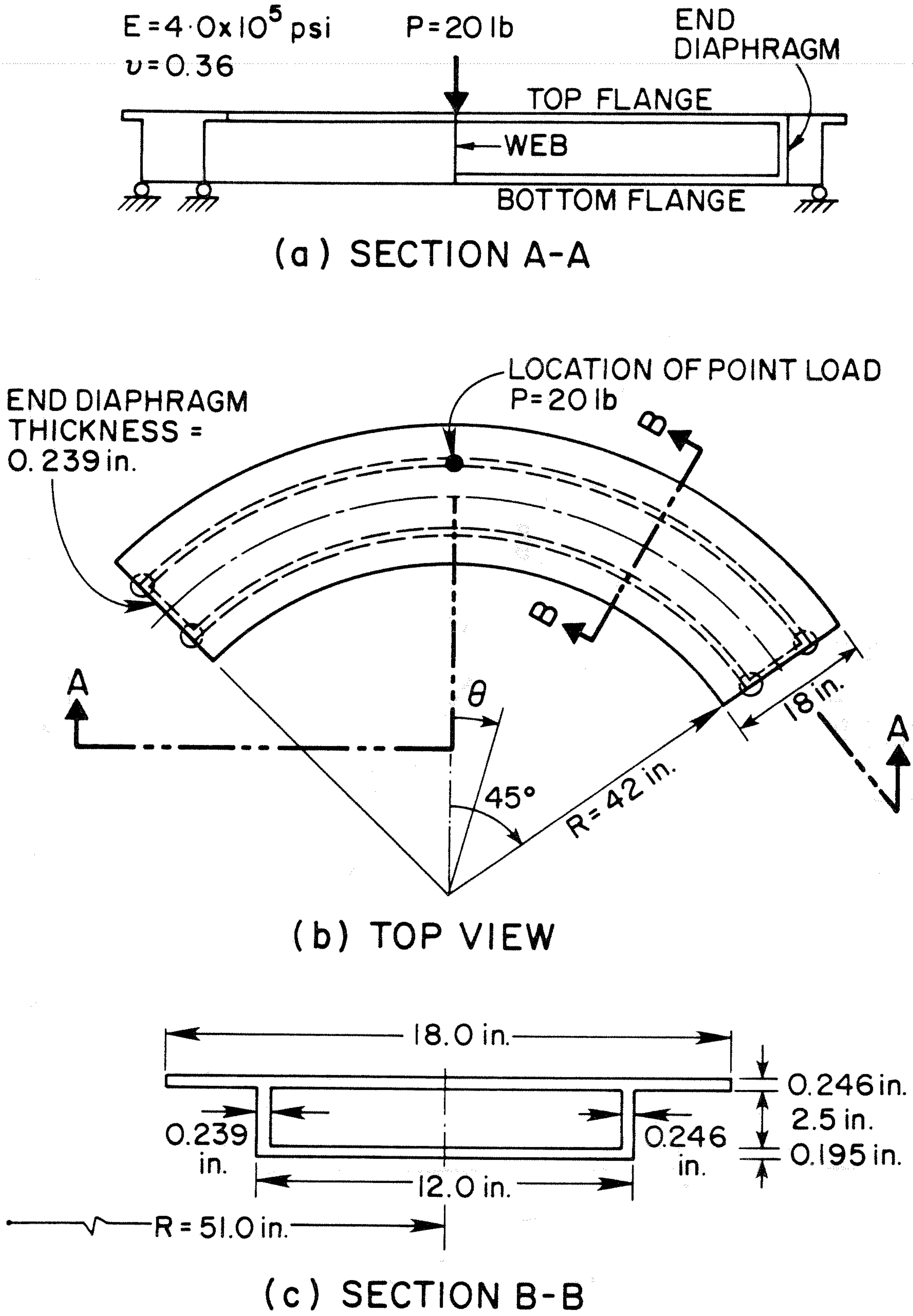
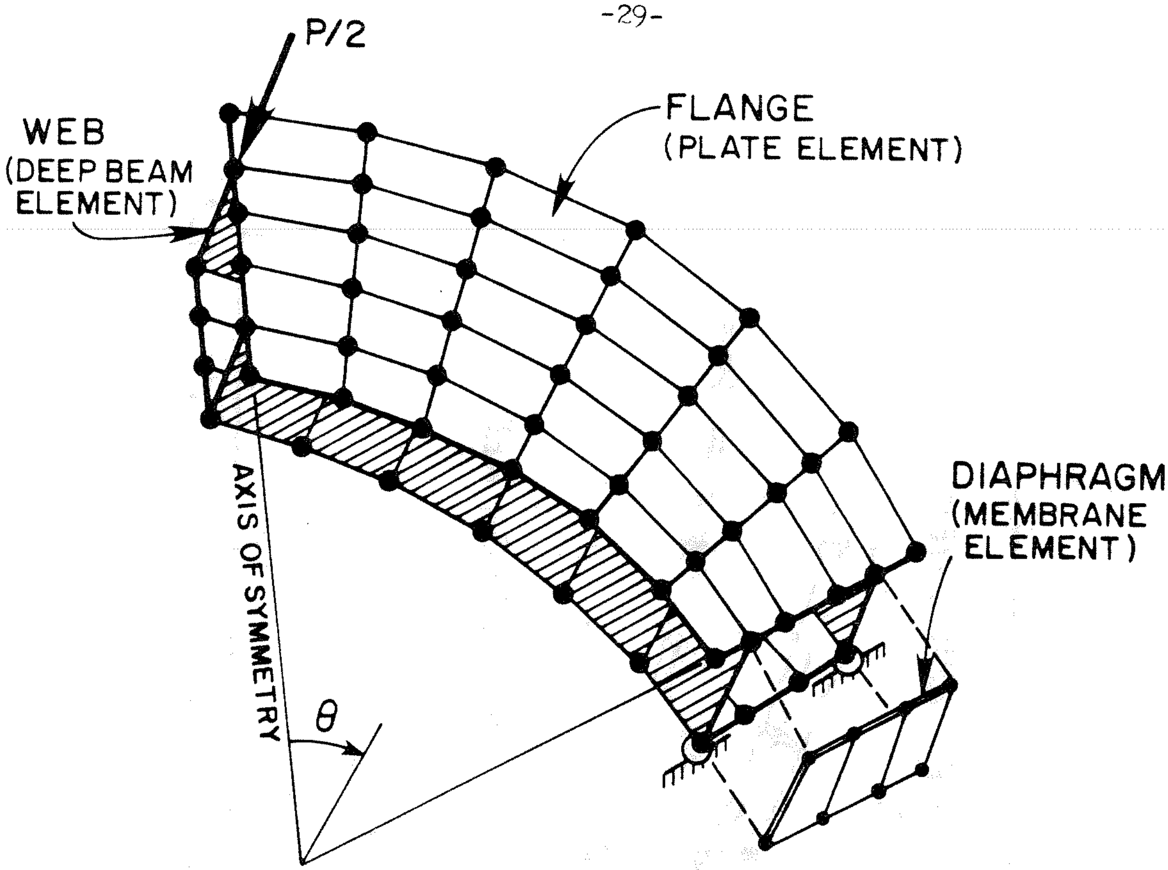


Figure 17. A horizontally curved box girder.



(a) FINITE ELEMENT MESH

TYPE	DOMINANT ACTION	GAUSSIAN QUADRATURE		
		NORMAL STRESSES	IN-PLANE SHEAR	TRAVERSE SHEAR
FLANGE (PRESENT ELEMENT)		2x2	2x2	1x1
WEB (PRESENT ELEMENT W/ IN-PLANE SHEAR MODIFICATION)		2x2	2x1	1x1
DIAPHRAGM (PRESENT ELEMENT W/ OUT-OF-PLANE EFFECT NEGLECTED)		ALL 2x2 OUT-OF-PLANE EFFECT IGNORED		

(b) MODIFICATION OF THE ELEMENT FOR WEB MEMBERS AND DIAPHRAGM

Figure 18. Curved box girder, finite element idealization.

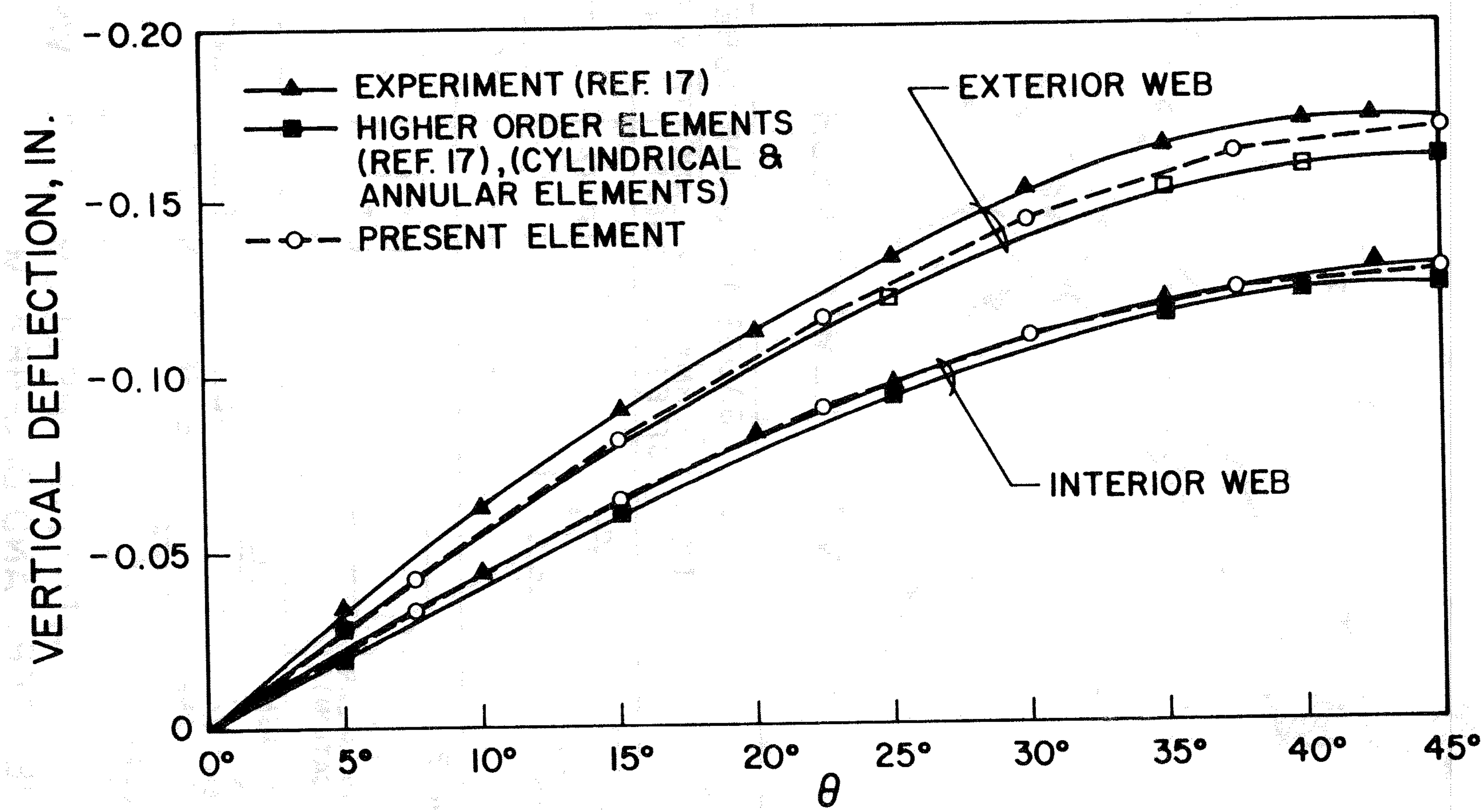


Figure 19. Vertical deflection along half span of box girder.

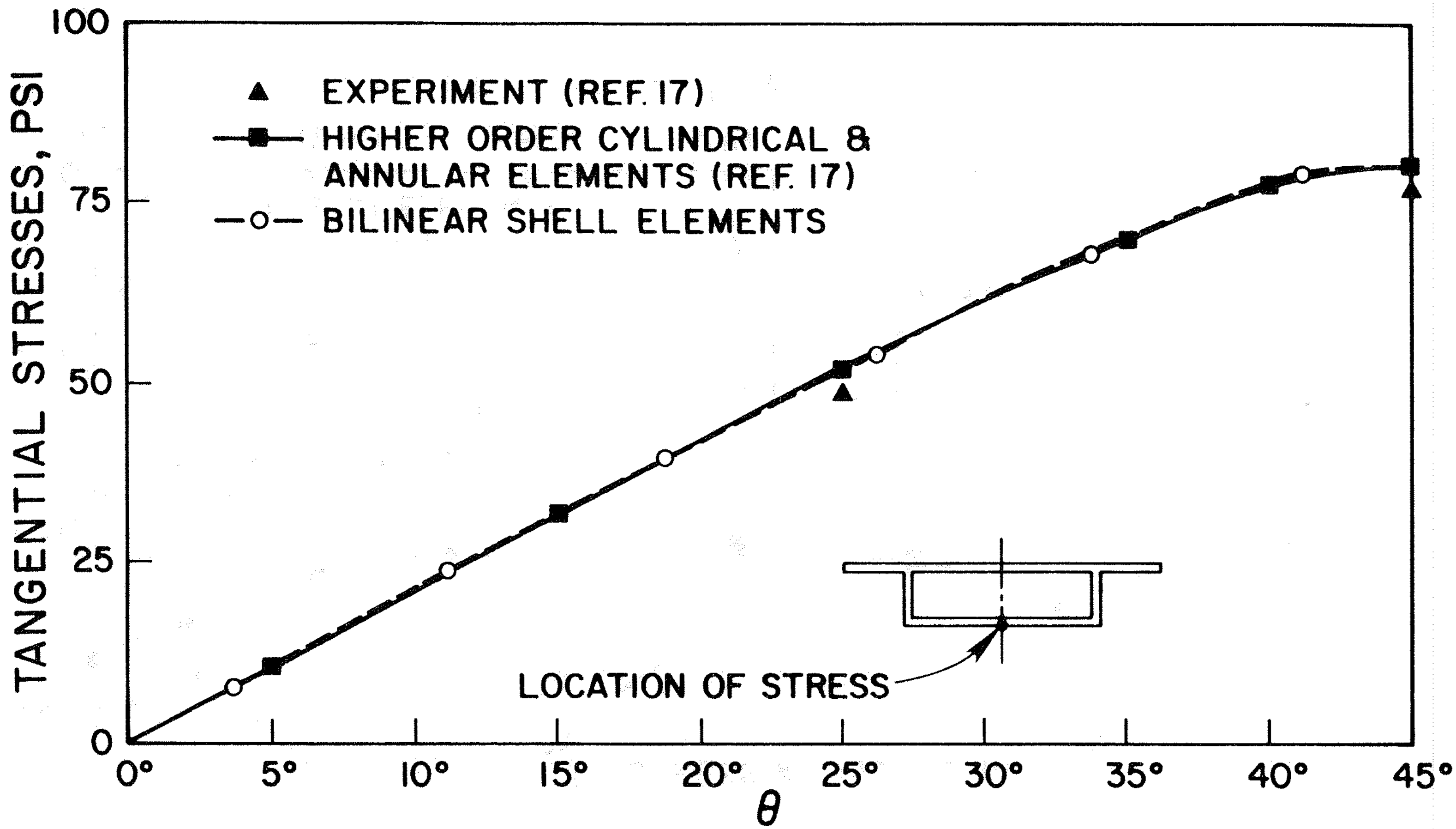


Figure 20. Tangential stresses on outer face of the bottom flange of box girder.

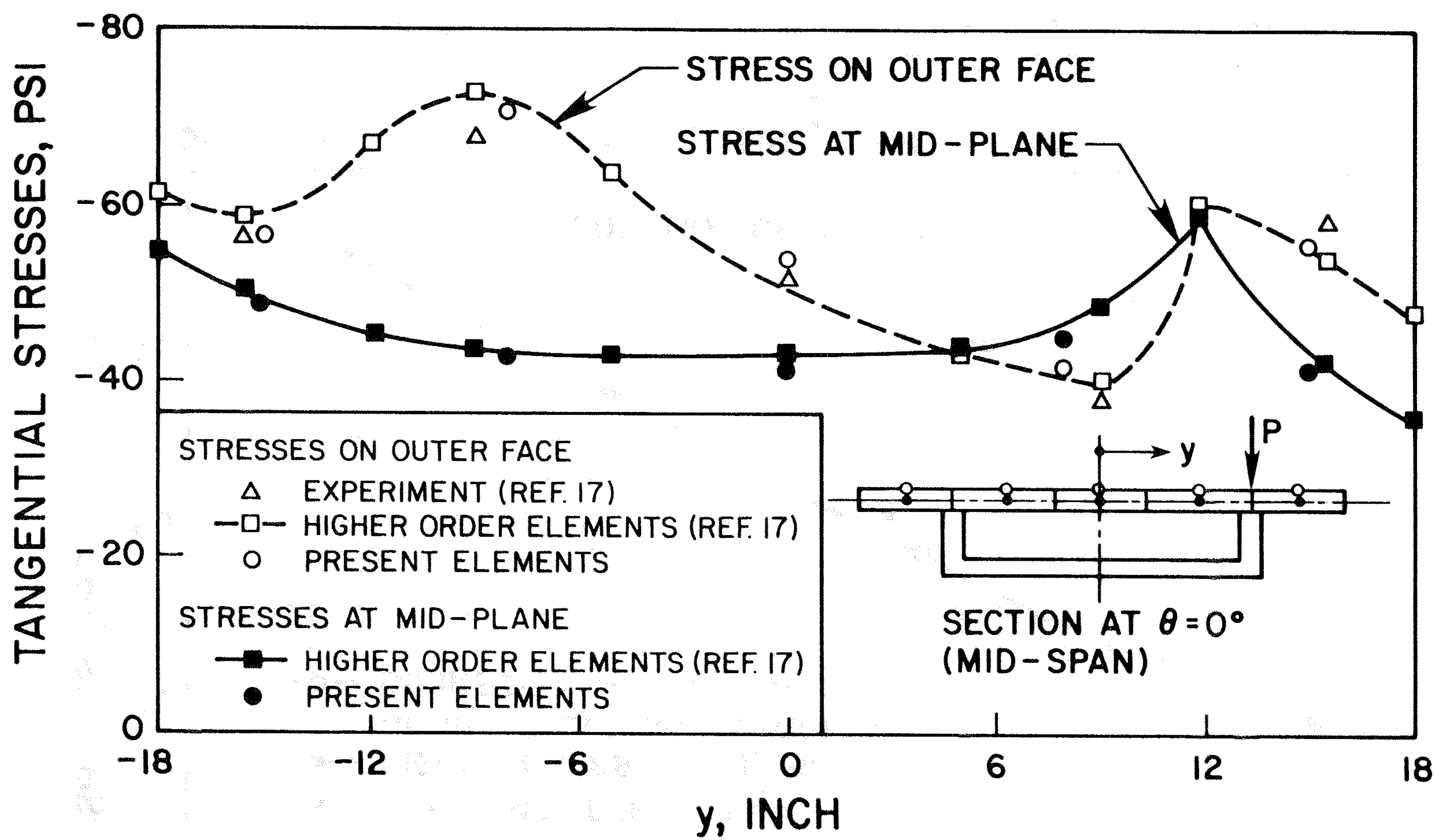


Figure 21. Tangential stresses in top flange at mid-span of box girder.

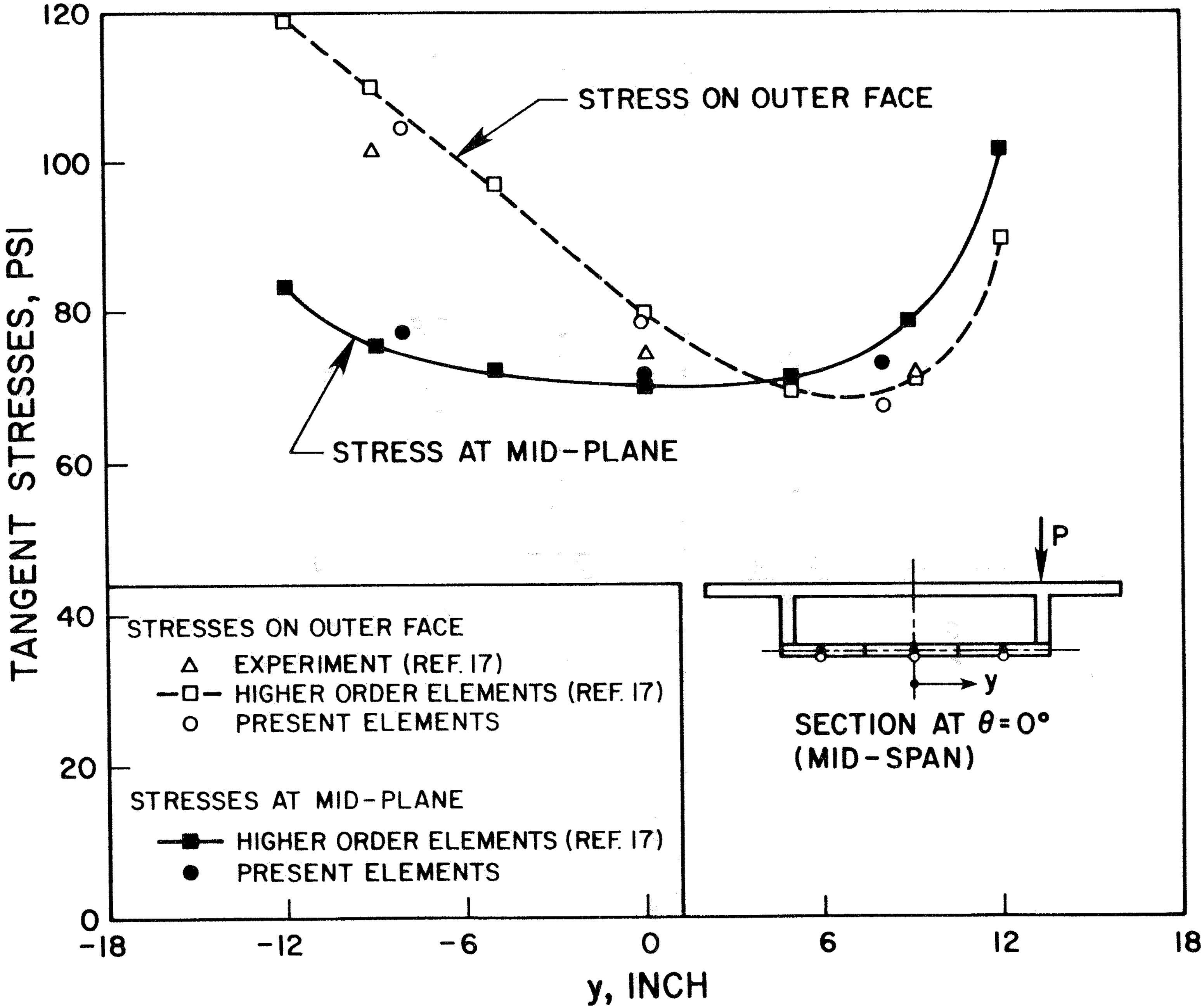


Figure 22. Tangential stresses in bottom flange at mid-span of box girder.

APPENDIX A

Shell Stress-Resultants

The usual shell stress-resultants: membrane forces, shears and couples, are defined as follow

$$\mathbf{N}(r,s) \equiv \begin{Bmatrix} N_{11} \\ N_{22} \\ N_{12} \\ N_{13} \\ N_{23} \end{Bmatrix} = \int_{-h/2}^{h/2} \sigma(r,s,t)\ dz_3 \tag{A1}$$

$$\mathbf{M}(r,s) \equiv \begin{Bmatrix} M_{11} \\ M_{22} \\ M_{12} \end{Bmatrix} = \int_{-h/2}^{h/2} z_3\, \sigma(r,s,t)\, dz_3 \tag{A2}$$

The equation (24) is substituted in (14) and the result in (A1) and (A2) which are then integrated to yield

$$\mathbf{N}_m \equiv \begin{Bmatrix} N_{11} \\ N_{22} \\ N_{12} \end{Bmatrix} = \sum_I h(r,s)\ \mathbf{D}_m\ \mathbf{B}_{1m}^I\ \mathbf{u}^I \tag{A3}$$

$$\mathbf{N}_s \equiv \begin{Bmatrix} N_{13} \\ N_{23} \end{Bmatrix} = \sum_I h(r,s)\ \mathbf{D}_s\ [\mathbf{B}_{1s}^I\ \mathbf{B}_{2s}^I] \begin{Bmatrix} \mathbf{u}^I \\ \alpha^I \end{Bmatrix} \tag{A4}$$

$$\mathbf{M} = \sum_I \frac{h^2}{6}(r,s)\ \mathbf{D}_m\ \mathbf{B}_{3m}^I\ \alpha^I \tag{A5}$$

APPENDIX B

Element Manual : Bilinear Degenerated Shell (BDS) Element

ELEMENT DESCRIPTION

a) Geometry : 4-node bilinear quadrilateral mid-surface, uniform or non-uniform thickness.

b) Material : linear elastic.

c) Applications : infinitesimal displacement analysis of
    -shells (thick, thin, non-uniform),
    -membranes,
    -deep beams, shear panels,
    -and any thin structures with user's assigned selective integration scheme.

CODING ASPECT

a) Main program : FEAP74 (by Prof. R.L. Taylor, University of California ,Berkeley).

b) Element identification : ELM04

c) Modular subroutines : BMATRIX, CROSSP, JACOBI, LOAD, LOCALX, TRIMUL, TORSION and SPACKD.

d) Input data cards :

FLOAD, WTABL
(2F10.0) omitted if LCODE = 0

E ,ν ,H,LCODE, κ_l,ZB, ρ ,IBS
(7F10.0,I5)

MA, 'ELM04' Alphanumerical information
(Standard card in MATERIAL section of FEAP)

Notation :

E - Modulus of Elasticity

ν - Posson ratio

H - Uniform thickness (For non-uniform thickness, leave H blank and input nodal thicknesses in VECT(NUMNP,1) array - see section 6 of FEAP Manual)

LCODE- 1 Dead load
2 Uniform load
3 Normal pressure
4 Water pressure

κ_t - coefficient of torsional stiffness (> 0.1 recommended)

ZB - ZB.ne.0 : all out-of-plane stiffness deleted.

ρ - Material density.

IBS - Selective integration scheme; three digits corresponding to numbers of quadrature points for (bending,inplane shear,tranverse shear) see Note. IBS = 441 for shell (default) and IBS = 421 for deep beam.

FLOAD- Load intensity according to

LCODE	Meaning of FLOAD	Direction
1	Specific weight	Global x_3
2	Uniform load/projected area	Global x_3
3	Normal pressure	Local z_3
4	Water pressure	Local z_3

WTABL- x_3 -coordinate of water table for LCODE=4 only

Note

For the case of a deep beam application, the order of element nodal connection must be such that sides 1-4 and 2-3 represent the beam thicknesses at both ends of the beam element. In this case, H becomes the width of the beam. Integration scheme IBS = 421 should be employed. It denotes a two-by-two quadrature being used in evaluating the stiffness due to bending effect (λ -energy), a one-by-two quadrature (the two on the line joining midsides 1-2 and 3-4) the stiffness due to inplane shear effect (μ -energy), and a one-by-one the stiffness due to transverse shear effect.

```
C                      ******************************
C                      *                            *
C                      *     A P P E N D I X   C    *
C                      *                            *
C                      ******************************
      SUBROUTINE ELMT04(N,MA,NDM,NDF,NEL,NEL1,NST,NSIZV,NVEC,MCT,DM,D,         ELM   1C
     1    XYZ,IX,F,P,S,U,VECT,ISW)                                             ELM   2C
C
C      *** BILINEAR SHELL ELEMENT - BY WARSAK KANOKNUKULCHAI -6/77 ***
C      ***************************************************************
C
C     GEOMETRY    - 4-NODED QUADRI-LATERAL IN 3-D SPACE (HYPERBOLIC
C                   PARABOLOID)
C     VARIABLES   - 6 DOF/NODE,3 DISPLACEMENTS AND 3 ROTATIONS
C     INTERPOLATION FUNCTION
C                 - BILINEAR FUNCTION FOR GEOMETRY AND ALL VARIABLES
C                   (ISO-PARAMETRIC ELEMENT)
C     CAPACITIES  - 1) LINEAR ELASTIC ISOTROPIC MATERIAL
C                 - 2) THICK/THIN SHELL WITH USER,S ASSIGNED QUADRATURES
C                   ON BENDING,INPLANE SHEAR AND TRANVERSESHEAR ENERGIES
C                   3) TORSIONAL STIFFNESS CAN BE ADDED TO RESTRAIN THE
C                   WEAK TORSIONAL MODE DUE TO WEAK COUPLING IN FINE MESH
C                 - 4) 4 TYPES OF LOADS BUILT IN
C                   A)GRAVITY ACTION(OPPOSITE TO GLOBAL Z3 DIRECTION)
C                   B)UNIFORM LOAD (IN Z3 DIRECTION)
C                   C) NORMAL PRESSURE(IN LOCAL Z3 DIRECTION)
C                   D) WATER PRESSURE (IN LOCAL Z3 DIRECTION)
C     INPUT  - FORMAT(7F10.0,I5/2F10.0)
C     D(1)-D(4),D(7),D(10),D(9),LBS/D(5),D(6) (ONLY D(4).NE.0)
C                 - D(1)=E
C                 - D(2)=POISSON*S RATIO
C                 - D(3)=UNIFORM THICKNESS (IF=0,THICKNESS IS INPUT FOR
C                   EACH NODE IN USER,S ESTABLISHED NUMNP-SIZED ARRAY
C                   *VECT* (SEE MANUAL SECTION 6)
C                 - D(4)=1 -GRAVITY ACTION, 2 -UNIFORM LOAD
C                   3 -NORMAL PRESSURE, 4 -WATER PRESSURE
C                 - D(7) TORSIONAL DEFORMATION COEFFICIENT , WHERE
C                   TORS.ENERGY=D(7)*MU*TT*INT(RELATIVE TORS ROTATION)**2
C                   RECOMMENDED VALUE = 0.1 TO 1000.
C                 - D(10).NE.0 ONLY INPLANE EFFECT CONSIDERED(DIAPHRAGME)
C                 - D( 9)=MATERIAL DENSITY (FOR LUMPED MASS MATRIX)
C                 - LBS=INTEGRATION SCHEME, IE. LBS=411 MEANS USING
C                   TOTAL 4 .4.1 POINTS FOR BENDING, INPLANE SHEAR
C                   INVARIENCES (LAM*IR(E)**2,2*MU*IR(E*E) ) AND
C                   TRANSVERSE SHEAR RESPECTIVELY (DF. LBS=441)
C                 - D(5)=LOAD INTENSITY CORRESPONDING TO 1)SPECIFIC WEIGH
C                   2) UNIFORM LOAD/PROJECTED AREA IN GLOBAL Z3 DIRECTION
C                   3)NORMAL PRESSURE  4) WATER SPECIFIC WEIGHT
C                 - D(6)=WATER TABLE IN GLOBAL Z3 DIRECTION (CASE 4 ONLY)
C
C*************************************************************************
      DIMENSION D(63,1),P(NST ,2),S(NST ,1),IX(NEL1,1),XYZ(NDM ,1),            ELM   3C
     1    VECT(NSIZV,1),OHM(3,3),EPS(5),XN(5),CHI(3),XM(3)                    ELM   4C
      COMMON/SHAP/VA(3,3),V(3),LX(3),V3(3,4),BU(3,3,4),BR(3,3,4),TR(3,5)      ELM   5C
     1,CR(3,3,4),CT(3,3)                                                       ELM   6C
      DIMENSION RG(4),UG(4)                                                    ELM   7C
      COMMON/LABELS/HEAD(12),O,IPG,XHED(3),UHED(6),XH,FH,UH,NSTR,ELMT(?)      ELM   8C
      COMMON/LOCALS/ DUL(6,20),UL(6,20),UDL(6,20),UDDL(6,20)                   ELM   9C
      COMMON/PRTPLT/ NSIG(9),NPLT(9,2),NT,NSTEP,NUMPLT,NEDATA(20,5),NPR,      ELM  10C
     1    NCOMP                                                                ELM  11C
      COMMON/TIMHIS/TIME,DT,DTP,NH,ISZH,C0,C1,C2,C3,C4,C5,C6,NCT              ELM  12C
      COMMON/ELM04/XJAC,RJAC(3,3),SHP(3,4),BN,B(2,4),R(4),Q(4),T(4)           ELM  13C
      LOGICAL NPR,NPL,ISW6                                                     ELM  14C
      DATA R/-1.0,1.0,1.0,-1.0/,Q/-1.0,-1.0,1.0,1.0/                           ELM  15C
      DATA RG/1.0,1.0,-1.0,-1.0/,QG/-1.0,1.0,1.0,-1.0/                         ELM  16C
C
      IF(ISW.LE.2)GO TO 999                                                    ELM  17C
C
C     COMMON ROUTE
C
      LBS=D(8,MA)                                                              ELM  18C
      RO=D(11,MA)                                                              ELM  19C
      SQ=SQRT(3.0)                                                             ELM  20C
C
C     IF OUT-OF-PLANE EFFECT NEGLECTED,RZERO=0.0
C
      RZERO=1.0                                                                ELM  21C
      IF(D(10,MA).NE.0.0)RZERO=0.0                                             ELM  22C
      DO 301 I=1,NEL                                                           ELM  23C
      IN=IX(I,N)                                                               ELM  24C
      T(I)=D(3,MA)                                                             ELM  25C
      IF(D(3,MA).EQ.0.0)T(I)=VECT(IN,1)                                        ELM  26C
  301 CONTINUE                                                                 ELM  27C
C
C     COMPUTE *NORMAL* AT 4 NODES IN V3
C
      DO 303 I=1,NEL                                                           ELM  28C
      SS=Q(I)                                                                  ELM  29C
      RR=R(I)                                                                  ELM  30C
      CALL LOCALX(RR,SS,TT,NDM,OHM)                                            ELM  31C
      V3(1,I)=OHM(1,3)                                                         ELM  32C
      V3(2,I)=OHM(2,3)                                                         ELM  33C
  303 V3(3,I)=OHM(3,3)                                                         ELM  34C
C
  999 GO TO (1,2,3,4,5,6),ISW                                                  ELM  35C
C
    1 READ(5,1100)(D(I,MA),I=1,4),D(7,MA),D(10,MA),D(11,MA),LBS                ELM  36C
      IF(D(4,MA).NE.0.)READ(5,1100)D(5,MA),D(6,MA)                             ELM  37C
      TEMP=1.2                                                                 ELM  38C
   11 LB=LBS/100                                                               ELM  39C
      LPS=MOD(LBS,100)/10                                                      ELM  40C
      LS=MOD(LBS,10)                                                           ELM  41C
      IF(LB.EQ.0) LB=4                                                         ELM  42C
      IF(LS .EQ.0)LS =1                                                        ELM  43C
      IF(LPS.EQ.0)LPS=LB                                                       ELM  44C
      WRITE(6,2100)(D(I,MA),I=1,2),D(11,MA),TEMP,LB,LPS,LS                     ELM  45C
      IF(D(10,MA).NE.0.0)WRITE(6,2101)                                         ELM  46C
      IF(D(3,MA).EQ.0.0.AND.NVEC.EQ.0) D(3,MA)=1.0                             ELM  47C
      IF(D(3,MA).NE.0.0) WRITE(6,2102) D(3,MA)                                 ELM  48C
      IF(D(4,MA).NE.0.0)WRITE(6,2103)D(4,MA),D(5,MA),D(6,MA)                   ELM  49C
      D(8,MA)=LBS                                                              ELM  50C
      WRITE(6,2104)D(7,MA)                                                     ELM  51C
      C6=0.0                                                                   ELM  52C
      ISW6=.FALSE.                                                             ELM  53C
C
    2 RETURN                                                                   ELM  54C
C
    3 IF(ISW.EQ.3.AND.ISW6)GO TO 389                                           ELM  55C
      LX(1)=LBS/100                                                            ELM  56C
      LX(2)=MOD(LBS,10)                                                        ELM  57C
      LX(3)=MOD(LBS,100)/10                                                    ELM  58C
      IF(LX(1).EQ.0)LX(1)=4                                                    ELM  59C
      IF(LX(2).EQ.0)LX(2)=1                                                    ELM  60C
      IF(LX(3).EQ.0)LX(3)=LX(1)                                                ELM  61C
C
```

```
C
C     ENERGIES SPLIT LOOP - ISP=(1,3)=(VOLUMETRIC,INPLANE SHEAR INVARI.)
C                           ISP= 2   =TRANSVERSE SHEAR ENERGY
      ISPLT=3                                                           ELM 62C
      IF(LX(3).EQ.LX(1))ISPLT=2                                         ELM 63C
      DO 399 ISP=1,ISPLT                                                ELM 64C
      IBS=ISP                                                           ELM 65C
      IF(ISP.EQ.3)IBS=1                                                 ELM 66C
      LL=LX(ISP)                                                        ELM 67C
      LR=1                                                              ELM 68C
      IF(LL.EQ.4)LR=2                                                   ELM 69C
      LS=2                                                              ELM 70C
      IF(LL.EQ.1)LS=1                                                   ELM 71C
      GR=(LR/2)/SQ                                                      ELM 72C
      GS=(LS/2)/SQ                                                      ELM 73C
      WT=(3.0-LR)*(3.0-LS)                                              ELM 74C
      CALL SPACKD(D(1,MA),CT,ISP,ISPLT)                                 ELM 75C
C
C     GAUSS POINTS LOOP
C
      DO 312 L=1,LL                                                     ELM 76C
      RR=RG(L)*GR                                                       ELM 77C
      SS=QG(L)*GS                                                       ELM 78C
      CALL LOCALX(RR,SS,TT,NDM,OHM)                                     ELM 79C
      CALL JACOBI(RR,SS,NDM,OHM)                                        ELM 80C
      DO 304 I=1,NEL                                                    ELM 81C
  304 CALL BMATRIX(RR,SS,TT,NDM,OHM,I,IBS)                              ELM 82C
      WW=XJAC*WT*2.0                                                    ELM 83C
      SJAC=SQRT(RJAC(1,3)**2+RJAC(2,3)**2+RJAC(3,3)**2)*XJAC            ELM 84C
      DO 302 I=1,3                                                      ELM 85C
      DO 302 J=1,I                                                      ELM 86C
      RJAC(I,J)=CT(I,J)*WW                                              ELM 87C
  302 RJAC(J,I)=RJAC(I,J)                                               ELM 88C
      IS=4-IBS                                                          ELM 89C
      IF(ISW.EQ.3.AND.ISP.EQ.1.AND.D(4,MA).NE.0.0)                      ELM 90C
     1CALL LOAD(NDM,NDF,D(4,MA),D(5,MA),D(6,MA),SJAC,TT,OHM,P(1,1))     ELM 91C
      J1=0                                                              ELM 92C
      DO 364 IJ=1,NEL                                                   ELM 93C
      K1=0                                                              ELM 94C
      DO 362 IK=1,IJ                                                    ELM 95C
      CALL TRIMUL(BU(1,1,IJ),RJAC,BU(1,1,IK),S(J1+1,K1+1),1.0,IS,NST)   ELM 96C
      IF(BZERO.NE.0.0)                                                  ELM 97C
     1CALL TRIMUL(BR(1,1,IJ),RJAC,BR(1,1,IK),S(J1+4,K1+4),3.0,IS,NST)   ELM 98C
      IF(IBS.EQ.1)GO TO 362                                             ELM 99C
      CALL TRIMUL(CR(1,1,IJ),RJAC,CR(1,1,IK),S(J1+4,K1+4),1.,2,NST)     ELM100C
      CALL TRIMUL(BU(1,1,IJ),RJAC,CR(1,1,IK),S(J1+1,K1+4),1.0,2,NST)    ELM101C
      CALL TRIMUL(CR(1,1,IJ),RJAC,BU(1,1,IK),S(J1+4,K1+1),1.0,2,NST)    ELM102C
  362 K1=K1+NDF                                                         ELM103C
  364 J1=J1+NDF                                                         ELM104C
  312 CONTINUE                                                          ELM105C
      IF(BZERO.EQ.0.0)GO TO 398                                         ELM106C
      IF(ISP.NE.2)GO TO 399                                             ELM107C
C
C     TORSIONAL ENERGY BY 1-POINT QUADRATURE IF D(7) .GT.0
C
      IF(D(7,MA).NE.0.)CALL TORSION(D(1,MA),S,SJAC,TT,OHM,NST,NEL,NDF)  ELM
  399 CONTINUE                                                          ELM109C
  398 DO 382 J=1,NST                                                    ELM110C
      DO 382 K=1,J                                                      ELM111C
  382 S(K,J)=S(J,K)                                                     ELM112C
      IF(ISW.EQ.6) GO TO 6361                                           ELM113C
  389 ISW6=.FALSE.                                                      ELM114C
      IF(C6.EQ.0.0)RETURN                                               ELM115C
      GO TO 5                                                           ELM116C
 3531 DO 390 I=1,NST                                                    ELM117C
  390 S(I,I)=S(I,I)+C6*P(I,2)                                           ELM118C
      RETURN                                                            ELM119C
C
    4 NHF=NDF/2                                                         ELM120C
      DO 450 J=1,5                                                      ELM121C
      EPS(J)=0.0                                                        ELM122C
  450 CHI(J)=0.0                                                        ELM123C
C
C     COMPUTE STRAIN AND CURVATURE AT CENTER
C
      RR=0.0                                                            ELM124C
      SS=0.0                                                            ELM125C
      CALL LOCALX(RR,SS,TT,NDM,OHM)                                     ELM126C
      CALL JACOBI(RR,SS,NDM,OHM)                                        ELM127C
      TEMP=2.0/TT                                                       ELM128C
      DO 470 I=1,NEL                                                    ELM129C
      CALL BMATRIX(RR,SS,TT,NDM,OHM,I,1)                                ELM130C
      DO 460 K=1,3                                                      ELM131C
      TP=UL(K,I)                                                        ELM132C
      TQ=UL(K+3,I)*TEMP                                                 ELM133C
      DO 460 J=1,3                                                      ELM134C
      EPS(J)=EPS(J)+BU(J,K,I)*TP                                        ELM135C
  460 IF(BZERO.NE.0.)CHI(J)=CHI(J)+BR(J,K,I)*TQ                         ELM136C
      IF(BZERO.EQ.0.0)GO TO 470                                         ELM137C
      CALL BMATRIX(RR,SS,TT,NDM,OHM,I,2)                                ELM138C
      DO 465 K=1,3                                                      ELM139C
      TP=UL(K,I)                                                        ELM140C
      TQ=UL(K+3,I)                                                      ELM141C
      DO 465 J=1,2                                                      ELM142C
  465 EPS(3+J)=EPS(3+J)+BU(J,K,I)*TP       + CR(J,K,I)*TQ               ELM143C
  470 CONTINUE                                                          ELM144C
C
C     COMPUTE STRESS-RESULTANT, COUPLES AT CENTERS
C
      CALL SPACKD(D(1,MA),CT,1,2)                                       ELM145C
      TEMP=TT**3/12.0                                                   ELM146C
      DO 475 J=1,3                                                      ELM147C
      XO(J,5)=0.0                                                       ELM148C
      DO 473 II=1,NEL                                                   ELM149C
  473 XO(J,5)=XO(J,5)+0.25*XO(J,II)                                     ELM150C
      XN(J)=0.0                                                         ELM151C
      XM(J)=0.0                                                         ELM152C
      DO 474 K=1,3                                                      ELM153C
      XN(J)=XN(J)+CT(J,K)*EPS(K)*TT                                     ELM154C
  474 XM(J)=XM(J)+CT(J,K)*CHI(K)*TEMP                                   ELM155C
  475 CONTINUE                                                          ELM156C
      CALL SPACKD(D(1,MA),CT,2,2)                                       ELM157C
      DO 480 J=1,2                                                      ELM158C
      XN(3+J)=0.0                                                       ELM159C
      DO 480 K=1,2                                                      ELM160C
  480 XN(3+J)=XN(3+J)+TT*CT(J,K)*EPS(3+K)                               ELM161C
      IF(NPR)GO TO 495                                                  ELM162C
      MCT=MCT-1                                                         ELM163C
      IF(MCT.GT.0) GO TO 490                                            ELM164C
      WRITE(6,2401) O,HEAD                                              ELM165C
      MCT=50                                                            ELM166C
  490 WRITE(6,2402) N,MA,(XO(J,5),J=1,3),EPS,( OHM(1,J),J=1,3),
     1              XN,( OHM(2,J),J=1,3),CHI,( OHM(3,J),J=1,3),XM       ELM167C
  495 RETURN                                                            ELM168C
C                                                                       ELM169C
    5 DO 512 L=1,4                                                      ELM170C
```

```
      RR=RG(L)/SQ                                                       ELM171C
      SS=QG(L)/SQ                                                       ELM172C
      CALL LOCALX(RR,SS,TT,NDM,OHM)                                     ELM173C
      CALL JACOBI(RR,SS,NDM,OHM)                                        ELM174C
      SJAC=SQRT(RJAC(1,3)**2+RJAC(2,3)**2+RJAC(3,3)**2)*XJAC            ELM175C
      XMASS=RO*SJAC*TT                                                  ELM176C
      RMASS=XMASS*TT*TT/12.0                                            ELM177C
      J1=0                                                              ELM178C
      DO 508 IJ=1,NEL                                                   ELM179C
      DO 306 JJ=1,3                                                     ELM180C
      P(J1+JJ,2)=P(J1+JJ,2)+XMASS*SHP(3,IJ)                             ELM181C
  306 P(J1+JJ+3,2)=P(J1+JJ+3,2)+RMASS*SHP(3,IJ)                         ELM182C
  508 J1=J1+NDF                                                         ELM183C
  512 CONTINUE                                                          ELM184C
      IF(ISW.EQ.3)GO TO 3531                                            ELM185C
      IF(ISW.EQ.6)GO TO 6561                                            ELM186C
      RETURN                                                            ELM187C
C
    6 IF(D(4,MA).EQ.0.0)GO TO 63                                        ELM188C
C
C     FORM ELEMNT-GENERATED LOAD
C
      DO 610 L=1,4                                                      ELM189C
      RR=RG(L)/SQ                                                       ELM190C
      SS=QG(L)/SQ                                                       ELM191C
      CALL LOCALX(RR,SS,TT,NDM,OHM)                                     ELM192C
      CALL JACOBI(RR,SS,NDM,OHM)                                        ELM193C
      SJAC=SQRT(RJAC(1,3)**2+RJAC(2,3)**2+RJAC(3,3)**2)*XJAC            ELM194C
  610 CALL LOAD(NDM,NDF,D(4,MA),D(5,MA),D(6,MA),SJAC,TT,OHM,P(1,1))     ELM195C
   63 DO 630 I=1,NEL                                                    ELM196C
      DO 630 J=1,NDF                                                    ELM197C
  630 IF(UL(J,I).NE.0.0)GO TO 3                                         ELM198C
      GO TO 65                                                          ELM199C
C
C     FORM INTERNAL FORCE  K(U)
C
 6361 K1=0                                                              ELM200C
      DO 633 K=1,NEL                                                    ELM201C
      DO 632 I=1,NDF                                                    ELM202C
      TP=UL(I,K)                                                        ELM203C
      DO 631 J=1,NST                                                    ELM204C
  631 P(J,1)=P(J,1)-S(J,K1+I)*TP                                        ELM205C
  632 CONTINUE                                                          ELM206C
  633 K1=K1+NDF                                                         ELM207C
      ISW6=.TRUE.                                                       ELM208C
   65 IF(C6.EQ.0.0)RETURN                                               ELM209C
      DO 650 I=1,NEL                                                    ELM210C
      DO 650 J=1,NDF                                                    ELM211C
  650 IF(UDDL(J,I).NE.0.0)GO TO 5                                       ELM212C
      RETURN                                                            ELM213C
C
C     FORM INERTIAL FORCE M*UDDL
C
 6561 K1=0                                                              ELM214C
      DO 653 K=1,NEL                                                    ELM215C
      DO 652 I=1,NDF                                                    ELM216C
      TP=UDDL(I,K)                                                      ELM217C
      DO 651 J=1,NST                                                    ELM218C
  651 P(J,1)=P(J,1)-P(J,2)*TP                                           ELM219C
  652 CONTINUE                                                          ELM220C
  653 K1=K1+NDF                                                         ELM221C
      RETURN                                                            ELM222C
C
```

```
C ... FORMATS
C
 1100 FORMAT(7F10.0,2I5)                                                ELM223C
 2100 FORMAT(/5X,7HMODULUS,13X,1H=,G13.4                                ELM224C
     1       /5X,13HPOISSON RATIO ,7X,1H=,G13.4                         ELM225C
     3       /5X,20HMASS DENSITY          1H=,G13.4,17H LUMPED MASS ONLY ELM226C
     2       /5X,17HCONSTRAINT FACTOR,3X,1H=,G13.4                      ELM227C
     4       /5X,20HNO OF GAUSS POINT                                   ELM228C
     5       /5X,20HBENDING              ,1H=,I5                        ELM229C
     5       /5X,20HINPLANE SHEAR        ,1H=,I5                        ELM230C
     5       /5X,20HTRANSVERSE SHEAR     ,1H=,I5      )                 ELM231C
 2101 FORMAT(5X,50HOUT-OF-SURFACE EFFECT NEGLECTED(MEMBRANE ELEMENT)   ) ELM232C
 2102 FORMAT(5X,17HUNIFORM THICKNESS,3X,1H=,G13.4)                      ELM233C
 2103 FORMAT(5X,12HLOADING CODE,8X,1H=,F10.1/5X,14HLOAD INTENSITY,6X,   ELM234C
     1 1H=,G13.4/27X,77H(1-SPECIFIC WEIGHT,2-UNIF.LOAD/PROJ.AREA, 3-NORM ELM235C
     1AL PRESSURE,4-WATER PRESSURE)/5X,20HWATER TABLE AT Z3   ,1H=,G13.4 ELM236C
     2 ,13H(ONLY CODE=4)        )                                       ELM237C
 2104 FORMAT(5X,35HCOEFFICIENT FOR TORSIONAL ENERGY =  ,G13.4)          ELM238C
 2401 FORMAT(A1,12A6,//5X,22HBILNEAR SHELL ELEMENT,/9H ELM MATL,3X,     ELM239C
     1     7H1-COORD,4X,7H2-COORD,4X,7H3-COORD,4X,11H1-DIRECTION,4X,    ELM240C
     2         11H2-DIRECTION,7X,8H12-SHEAR,7X,8H13-SHEAR,7X,8H23-SHEAR//) ELM241C
 2402 FORMAT(2I4,3F11.3,/6X,18H DIRECTION COSINES,6X,                   ELM242C
     1         11HEPS-STRAINS    ,5E15.5/6X,3F6.2,6X                    ELM243C
     2         11HRESULTANTS     ,5E15.5/6X,3F6.2,6X                    ELM244C
     2         11HCHI-STRAINS    ,3E15.5/6X,3F6.2,6X                    ELM245C
     2         11HCOUPLES         3E15.5)                               ELM246C
      END                                                               ELM247C
```

```
      SUBROUTINE BMATRIX(RR,SS,TT,NDM,OHM,I,ISW)                        ELM248C
C     SUBROUTINE TO COMPUTE BMU, BMR  WHERE
C               EM= BMU*U+Z*BMR*ROTATION
      DIMENSION OHM(3,3)                                                ELM249C
      COMMON/SHAP/VA(3,3),VV(3),LX(3),V3(3,4),BMU(3,3,4),BMR(3,3,4),    ELM250C
     1 XO(3,5),CR(3,3,4),CT(3,3)                                        ELM251C
      COMMON/ELM04/XJAC,RJAC(3,3),SHP(3,4),BN,B(2,4),R(4),S(4),T(4)     ELM252C
      GO TO (1,100),ISW                                                 ELM253C
    1 DO 20 J=1,3                                                       ELM254C
      DO 20 K=1,3                                                       ELM255C
      IF(J.EQ.3) GO TO 10                                               ELM256C
      BMU(J,K,I)=B(J,I)*OHM(K,J)                                        ELM257C
      BMR(J,K,I)=0.0                                                    ELM258C
      DO 5 L=1,3                                                        ELM259C
      IF(L.EQ.K) GO TO 5                                                ELM260C
      SIGN=1.0                                                          ELM261C
      IF(L.GT.K)SIGN=-1.0                                               ELM262C
      LK=6-L-K                                                          ELM263C
      BC=V3(LK,I)                                                       ELM264C
      IF(LK.EQ.2)BC=-BC                                                 ELM265C
      BMR(J,K,I)=BMR(J,K,I)+OHM(L,J)*BC*SIGN*T(I)/2.0                   ELM266C
    5 CONTINUE                                                          ELM267C
      GO TO 20                                                          ELM268C
   10 BMU(3,K,I)=B(2,I)*OHM(K,1)+B(1,I)*OHM(K,2)                        ELM269C
      BMR(3,K,I)=B(1,I)*BMR(2,K,I)+B(2,I)*BMR(1,K,I)                    ELM270C
   20 CONTINUE                                                          ELM271C
      DO 30 J=1,2                                                       ELM272C
      DO 30 K=1,3                                                       ELM273C
   30 BMR(J,K,I)=BMR(J,K,I)*B(J,I)                                      ELM274C
      RETURN                                                            ELM275C
C          ES=BSU*U+Z*BSR*R+CSR*ROTATION
  100 FAC=BN*SHP(3,I)                                                   ELM276C
      DO 60 J=1,3                                                       ELM277C
```

```
      DO 60 K=1,3                                                       ELM278C
      CR(J,K,I)=0.0                                                     ELM279C
      DO 65 L=1,3                                                       ELM280C
      IF(L.EQ.K)GO TO 65                                                ELM281C
      SIGN=1.0                                                          ELM282C
      IF(L.GT.K)SIGN=-1.0                                               ELM283C
      LK=6-L-K                                                          ELM284C
      BC=V3(LK,I)                                                       ELM285C
      IF(LK.EQ.2)BC=-BC                                                 ELM286C
      CR(J,K,I)=CR(J,K,I)+OHM(L,J)*BC*SIGN*T(I)/2.0                     ELM287C
   65 CONTINUE                                                          ELM288C
      IF(J.LT.3)CR(J,K,I)=CR(J,K,I)*FAC                                 ELM289C
   60 BMU(J,K,I)=B(J,I)*OHM(K,3)                                        ELM290C
      DO 70 J=1,2                                                       ELM291C
      DO 70 K=1,3                                                       ELM292C
   70 BMR(J,K,I)=B(J,I)*CR(3,K,I)                                       ELM293C
      RETURN                                                            ELM294C
      END                                                               ELM295C
```

```
      SUBROUTINE CROSSP(A,B,C,IN)                                       ELM296C
C      SUBROUTINE TO FINE CROSS-PRODUCT (A*B)=C, AND NORMALIZED IF IN=0
      DIMENSION C(3),B(3),A(3)                                          ELM297C
      C(1)=A(2)*B(3)-A(3)*B(2)                                          ELM298C
      C(2)=A(3)*B(1)-A(1)*B(3)                                          ELM299C
      C(3)=A(1)*B(2)-A(2)*B(1)                                          ELM300C
      IF(IN.NE.0)RETURN                                                 ELM301C
      CNORM=SQRT(C(1)*C(1)+C(2)*C(2)+C(3)*C(3))                         ELM302C
      DO 1 I=1,3                                                        ELM303C
    1 C(I)=C(I)/CNORM                                                   ELM304C
      RETURN                                                            ELM305C
      END                                                               ELM306C
```

```
      SUBROUTINE JACOBI(RR,SS,NDM,OHM)                                  ELM307C
      COMMON/ELM04/XJAC,RJAC(3,3),SHP(3,4),BN,B(2,4),R(4),S(4),T(4)     ELM308C
      COMMON/SHAP/AJAC(3,3),VA(3),LX(3),V3(3,4),BU(3,3,4),BR(3,3,4),    ELM309C
     1XO(3,5), CR(3,3,4),CT(3,3)                                        ELM310C
      DIMENSION OHM(3,3)                                                ELM311C
C     FIND JACOBIAN MATRIC, DETERMINANT AND INVERSE,AJAC,XJAC,RJAC
      DO 35 K=1,NDM                                                     ELM312C
      AJAC(K,3)=0.0                                                     ELM313C
      DO 35 J=1,2                                                       ELM314C
      AJAC(K,J)=0.0                                                     ELM315C
      DO 30 I=1,4                                                       ELM316C
   30 AJAC(K,J)=AJAC(K,J)+SHP(J,I)*XO(K,I)                              ELM317C
   35 CONTINUE                                                          ELM318C
      DO 20 I=1,4                                                       ELM319C
      TP=SHP(3,I)*T(I)/2.0                                              ELM320C
      DO 25 K=1,3                                                       ELM321C
   25 AJAC(K,3)=AJAC(K,3)+V3(K,I)*TP                                    ELM322C
   20 CONTINUE                                                          ELM323C
      XJAC=0.0                                                          ELM324C
      DO 40 J=1,NDM                                                     ELM325C
      J2=J+1                                                            ELM326C
      J3=J+2                                                            ELM327C
      IF(J3.GT.3) J3=J3-3                                               ELM328C
      IF(J2.GT.3)J2=J2-3                                                ELM329C
   40 XJAC=XJAC+AJAC(1,J)*(AJAC(2,J2)*AJAC(3,J3)-AJAC(2,J3)*AJAC(3,J2)) ELM330C
      CALL CROSSP(AJAC(1,2),AJAC(1,3),RJAC(1,1),1)                      ELM331C
      CALL CROSSP(AJAC(1,3),AJAC(1,1),RJAC(1,2),1)                      ELM332C
      CALL CROSSP(AJAC(1,1),AJAC(1,2),RJAC(1,3),1)                      ELM333C
      DO 50 J=1,NDM                                                     ELM334C
      DO 50 K=1,NDM                                                     ELM335C
   50 RJAC(J,K)=RJAC(J,K)/XJAC                                          ELM336C
C
C     TO DETERMINE B=L(N) AND BN=B(ZETA)
C     B(1,I)=OHM(1).(N(I)),X
C     B(2,I)=OHM(2).(N(I)),X
C     BN=OHM(3).(ZETA),X
C
      DO 14 J=1,2                                                       ELM337C
      DO 14 I=1,4                                                       ELM338C
      B(J,I)=0.0                                                        ELM339C
      DO 12 K=1,3                                                       ELM340C
      VA(K)=0.0                                                         ELM341C
      DO 10  L=1,2                                                      ELM342C
   10 VA(K)   =VA(K)+RJAC(K,L)*SHP(L,I)                                 ELM343C
   12 B(J,I)=B(J,I)+VA(K)*OHM(K,J)                                      ELM344C
   14 CONTINUE                                                          ELM345C
      BN=0.0                                                            ELM346C
      DO 15 J=1,3                                                       ELM347C
   15 BN=BN+OHM(J,3)*RJAC(J,3)                                          ELM348C
      RETURN                                                            ELM349C
      END                                                               ELM350C
```

```
      SUBROUTINE LOAD(NDM,NDF,TYPE,F,XR,WW,TT,OHM,P)                    ELM351C
C     SUBROUTINE TO COMPUTE GENERALIZED LOAD CORRESPONDING TO EACH LOAD
C     CASES
      COMMON/SHAP/VA(3,3),V(3),LX(3),V3(3,4),BU(3,3,4),BR(3,3,4),XO(3,5) ELM352C
     1,CR(3,3,4),CT(3,3)                                                ELM353C
      COMMON/ELM04/XJAC,RJAC(3,3),SHP(3,4),BN,B(2,4),R(4),S(4),T(4)     ELM354C
      DIMENSION OHM(3,3) ,P(1)                                          ELM355C
      NJ=1                                                              ELM356C
      JJ=0                                                              ELM357C
      IF(TYPE.GE.3.0)NJ=3                                               ELM358C
      H=1.0                                                             ELM359C
      IF(TYPE.LE.3.0)GO TO 20                                           ELM360C
      H=0.0                                                             ELM361C
      DO 25 I=1,4                                                       ELM362C
   25 H=H+SHP(3,I)*XO(3,I)                                              ELM363C
      H=(XR-H)                                                          ELM364C
      IF(H.LT.0.0)H=0.0                                                 ELM365C
   20 DO 32 I=1,4                                                       ELM366C
      DO 30 K=1,NJ                                                      ELM367C
      J=4-K                                                             ELM368C
      A=OHM(J,3)*H                                                      ELM369C
      IF(TYPE.EQ.2.0)A=OHM(3,3)                                         ELM370C
      IF(TYPE.EQ. 1.0)A=-TT                                             ELM371C
   30 P(JJ+J)=P(JJ+J)+SHP(3,I)*F*WW*A                                   ELM372C
   32 JJ=JJ+NDF                                                         ELM373C
      RETURN                                                            ELM374C
      END                                                               ELM375C
```

```
      SUBROUTINE LOCALX(RR,SS,TT,NDM,OHM)                               ELM376C
      COMMON/SHAP/VA(3,3),V(3),LX(3),V3(3,4),BU(3,3,4),BR(3,3,4),XO(3,5) ELM377C
     1,CR(3,3,4),CT(3,3)                                                ELM378C
      COMMON/ELM04/XJAC,RJAC(3,3),SHP(3,4),BN,B(2,4),R(4),S(4),T(4)     ELM379C
      DIMENSION OHM(3,3)                                                ELM380C
      TT=0.0                                                            ELM381C
```

```
      DO 1  I=1,4                                                       ELM382C
      SHP(1,I)=0.25*R(I)*(1.0+S(I)*SS)                                  ELM383C
      SHP(2,I)=0.25*S(I)*(1.0+R(I)*RR)                                  ELM384C
      SHP(3,I)=0.25*(1.0+S(I)*SS)*(1.0+R(I)*RR)                         ELM385C
    1 TT=TT+SHP(3,I)*T(I)                                               ELM386C
C     FIND DIRECTION COSINE OF LOCAL AXES BETWEEN ANY POINT(RR,SS) AND
C     GLOBAL  AXIS
      DO 10 J=1,NDM                                                     ELM387C
      OHM(J,1)=0.0                                                      ELM388C
      OHM(J,2)=0.0                                                      ELM389C
      DO 10 I=1,4                                                       ELM390C
      OHM(J,1)=OHM(J,1)+SHP(1,I)*XO(J,I)                                ELM391C
   10 OHM(J,2)=OHM(J,2)+SHP(2,I)*XO(J,I)                                ELM392C
      CALL CROSSP(OHM(1,1),OHM(1,2),OHM(1,3),0)                         ELM393C
      DO 12 J=1,NDM                                                     ELM394C
      OHM(J,1)=0.0                                                      ELM395C
      DO 12 I=1,4                                                       ELM396C
   12 OHM(J,1)=OHM(J,1)+R(I)*.25*XO(J,I)                                ELM397C
      CALL CROSSP(OHM(1,3),OHM(1,1),OHM(1,2),0)                         ELM398C
      CALL CROSSP(OHM(1,2),OHM(1,3),OHM(1,1),0)                         ELM399C
      RETURN                                                            ELM400C
      END                                                               ELM401C
```

```
      SUBROUTINE SPACKD(D,CT,K,L)                                       ELM402C
      DIMENSION D(1),CT(3,3)                                            ELM403C
      SLAM=D(1)*D(2)/(1.0-D(2)*D(2))                                    ELM404C
      TMU=D(1)/(1.0+D(2))                                               ELM405C
      DO 5 I=1,3                                                        ELM406C
      DO 5 J=1,3                                                        ELM407C
    5 CT(I,J)=0.0                                                       ELM408C
      IF(K.EQ.2)GO TO 10                                                ELM409C
      IF(L.EQ.2)GO TO 20                                                ELM410C
      GO TO (30,10,40),K                                                ELM411C
   30 CT(1,1)=2.0*SLAM*TMU/(SLAM+2.0*TMU)                               ELM412C
      CT(2,2)=CT(1,1)                                                   ELM413C
      CT(1,2)=CT(1,1)                                                   ELM414C
      CT(2,1)=CT(1,1)                                                   ELM415C
      RETURN                                                            ELM416C
   40 CT(1,1)=TMU                                                       ELM417C
      CT(2,2)=CT(1,1)                                                   ELM418C
      CT(3,3)=CT(1,1)/2.4                                               ELM419C
      RETURN                                                            ELM420C
   20 CT(1,1)=SLAM+TMU                                                  ELM421C
      CT(1,2)=SLAM                                                      ELM422C
      CT(2,1)=SLAM                                                      ELM423C
      CT(2,2)=CT(1,1)                                                   ELM424C
      CT(3,3)=TMU/2.0                                                   ELM425C
      RETURN                                                            ELM426C
   10 CT(1,1)=TMU/2.4                                                   ELM427C
      CT(2,2)=CT(1,1)                                                   ELM428C
      RETURN                                                            ELM429C
      END                                                               ELM430C
```

```
      SUBROUTINE TORSION(D,S,WS,TT,OHM,NST,NEL,NDF)
      DIMENSION D(1),S(NST,1),OHM(3,3)                                  ELM432C
      COMMON/ELM04/XJAC,RJAC(3,3),SHP(3,4),BN,B(2,4),R(4),Q(4),T(4)     ELM433C
      COMMON/SHAP/VA(3,4),LX(3),V3(3,4),BU(3,3,4),BR(3,3,4),XO(3,5)     ELM434C
     1,CR(3,3,4),CT(3,3)                                                ELM435C
      GG=CT(1,1)*1.2                                                    ELM436C
      GW=WS*GG*TT*D(7)*4.0                                              ELM436C
      DO 10  IJ=1,NEL                                                   ELM438C
      DO 10 K=1,3                                                       ELM439C
   10 VA(K,IJ)=0.5*(OHM(K,1)*B(2,IJ)-OHM(K,2)*B(1,IJ))                  ELM440C
      J1=0                                                              ELM442C
      DO 1 IJ=1,NEL                                                     ELM443C
      K1=0                                                              ELM444C
      DO 20 IK=1,IJ                                                     ELM445C
      GW2=GW*SHP(3,IJ)*SHP(3,IK)                                        ELM446C
      DO 40 J=1,3                                                       ELM447C
      DO 40 K=1,3                                                       ELM448C
      S(J1+J,K1+K)=S(J1+J,K1+K)+VA(J,IJ)*VA(K,IK)*GW                    ELM449C
      S(J1+3+J,K1+3+K)=S(J1+3+J,K1+3+K)+OHM(J,3)*OHM(K,3)*GW2           ELM450C
      S(J1+J   ,K1+3+K)=S(J1+J   ,K1+3+K)+VA(J,IJ)*SHP(3,IK)*OHM(K,3)*GW ELM451C
   40 S(J1+J+3,K1+K   )=S(J1+J+3,K1+K   )+SHP(3,IJ)*OHM(J,3)*VA(K,IK)*GW ELM452C
   20 K1=K1+NDF                                                         ELM453C
    1 J1=J1+NDF                                                         ELM454C
      RETURN                                                            ELM455C
      END                                                               ELM456C
```

```
      SUBROUTINE TRIMUL(A,B,C,ABC,FAC,IB,NST)                           ELM457C
      DIMENSION A(3,3),B(3,3),C(3,3),ABC(NST,3),S(3)                    ELM458C
      DO 250 L=1,3                                                      ELM459C
      DO 150 J=1,IB                                                     ELM460C
      TEMP=0.0                                                          ELM461C
      DO 100 K=1,IB                                                     ELM462C
  100 TEMP=TEMP+B(J,K)*C(K,L)                                           ELM463C
  150 S(J)=TEMP                                                         ELM464C
      DO 250 I=1,3                                                      ELM465C
      TEMP=0.0                                                          ELM466C
      DO 200 J=1,IB                                                     ELM467C
  200 TEMP=TEMP+A(J,I)*S(J)                                             ELM468C
  250 ABC(I,L)=ABC(I,L)+TEMP/FAC                                        ELM469C
      RETURN                                                            ELM470C
      END                                                               ELM471C
```
