This is to certify that the

thesis entitled

A NEW THERMAL THEORY AND ITS USE IN THERMO-MECHANICAL ANALYSIS OF LAMINATED COMPOSITES AND SANDWICH PANELS

presented by

Antonio Pantano

has been accepted towards fulfillment
of the requirements for

M. S. degree in MECHANICS

Ronald C. Averill
Major professor

Date 11/30/99

PLACE IN RETURN BOX to remove this checkout from your record.
TO AVOID FINES return on or before date due.
MAY BE RECALLED with earlier due date if requested.

DATE DUE	DATE DUE	DATE DUE

11/00 c:/CIRC/DateDue.p65-p.14

A NEW THERMAL THEORY AND ITS USE IN THERMO-MECHANICAL ANALYSIS OF LAMINATED COMPOSITES AND SANDWICH PANELS

By

Antonio Pantano

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Department of Materials Science and Mechanics

1999

ABSTRACT

A NEW THERMAL THEORY AND ITS USE IN THERMO-MECHANICAL ANALYSIS OF LAMINATED COMPOSITES AND SANDWICH PANELS

By

Antonio Pantano

A new thermal lamination theory and its associated finite element model are presented for analysis of heat transfer in laminated composite structures. The form of the present theory closely resembles that of recent zig-zag sublaminate structural laminate theories. The through-thickness distribution of temperature is assumed to vary linearly within each ply, and continuity of transverse flux at ply interfaces is enforced analytically. This theory lends itself well to development of efficient finite element models. Non-linear capabilities are implemented in the model to solve heat conduction problems in the presence of material non-linearity.

Combining the presented thermal theory with a zig-zag structural theory a 3D finite element model for thermal stress analysis of composite laminates has been realized. The novel features of the formulation allow accurate results in predicting the distribution of temperatures, displacements and stresses in laminated plates wherein the plies have dissimilar thermal and/or structural properties. Both linear and nonlinear numerical results are presented.

To my Mother and Father

ACKNOWLEDGMENTS

I will always be grateful to my beloved parents for years of unconditional support. I hope they will be proud of me, now and in the future.

I would like to reserve a special gratitude to my sweet Giulia. Without her love this result would have been harder to achieve.

I definitely recognize the immense fortune I had to meet a person like my research advisor Prof. Ronald Averill. His uncommon humanity equals only his exceptional academic knowledge.

I also greatly appreciate Prof. Robert WM. Soutas-Little and Prof. Tomas J. Pence for making efforts to serve my thesis committee.

Antonio Pantano

November 1999

East Lansing, Michigan

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1 INTRODUCTION

1.1 Preliminary Information

The technology of composite materials is quickly developing to satisfy the demand for materials with advanced properties. A typical laminated composite sandwich structure allows the possibility to achieve new multi-functional such as high bending stiffness, low weight, variable heat transfer capacities, sound insulation and extended operational life. The sandwich is one of the composite structures that better expresses these qualities. It consists of one or more high strength, stiff face layers separated by a thick low-density flexible core. Whereas the facings provide the primary inplane load capacity, the core carries the majority of the transverse load.

The increasing use of these materials requires a more accurate analysis of stresses induced by thermal as well as mechanical loads. In particular, thermal stresses are achieving a growing consideration in structural design since they come into play in numerous situations, including the manufacturing process. Several components of the aerospace industry, such as engine parts, heat shields, nozzles and nose cones are subjected to repeated and prolonged exposure to severe heat environments. The rising production of electronic devices, such as circuit boards, requires knowledge of thermal stress behavior of composite materials to improve reliability. Many other examples from the automotive and marine industry could be reported.

The accuracy in predicting thermal stresses in composite materials mainly depends upon the correctness of two basic steps: the evaluation of temperature distributions (heat transfer model) and the computation of the resulting stresses (plate theories for structural analysis). To follow industry needs for innovation, precise calculations should be maintained also in case of laminates composed of layers with highly dissimilar material properties.

1.2 Literature Review

The development of heat transfer models for laminated structures has followed closely that of laminate theories for structural analysis. The assumed through-thickness distribution of temperature in thermal theories resembles the through-thickness distribution of displacements in structural laminate theories. It follows that if the structural model has a linear through-thickness distribution of displacements, then the temperature distribution is computed by a thermal model based on a linear through-thickness variation.

1.2.1 Equivalent Single Layer Approach

The most common approach is the equivalent single layer approach, wherein the material properties of all the layers are "smeared", and the laminate is modeled as an equivalent single anisotropic layer. The most popular of these theories is the first order shear deformation theory (FSDT) [1, 2]. This theory yields good predictions of overall laminate behavior and inplane stresses provided the plate is thin and the material properties of adjacent layers do not differ significantly. However, FSDT does not account for warpage of the cross section, which may be significant in laminated composites. In order to reduce the inaccuracies of the FSDT, higher order shear deformation theories (HSDT) were proposed [3-5]. These theories permit nonlinear variations of displacements, strains and stresses through the thickness, and thus warpage of the cross section. However, even though they often can predict well the gross behavior of thin and some moderately thick laminates, all equivalent single layer theories have a common shortcoming. They are unable to account for the sometimes-severe discontinuities in transverse shear strains that occur at interfaces between two adjacent layers with drastically different stiffness properties. In these cases, the local deformations and stresses, and even the overall laminate response, are not well predicted.

The heat transfer models derived from the equivalent single layer approach share the limits of the displacement theory. They assume a polynomial (usually linear) variation of temperature through the thickness of the laminate and replace the discrete-layer thermal properties with an equivalent set of homogeneous anisotropic thermal properties using a Thermal Lamination Theory (TLT) (see, for example [6-11]). For greater accuracy, higher order thermal lamination theories were proposed [4,6,12]. However, when (i) the span-to-thickness ratio of the laminate is small, (ii) the thickness-direction components of the thermal conductivity of adjacent plies differs significantly, and (iii) the time gradients of temperature are large, both the first and high-order TLT produce approximate results.

1.2

de

po

ac

of

an

1.2

dis

are

co

ex

str

ten

in

inc

1.2

de

an

app

1.2.2 FSDT with Postprocessing Techniques

In recent years, since finite element models based on the first order shear deformation theory are not computationally expensive, their use in combination with postprocessing techniques has received increased attention. These approaches allow more accurate evaluations of transverse shear stresses in thermally loaded laminates by the use of: 1) three-dimensional equilibrium equations [13,14], 2) predictor-corrector approaches and 3) simplifying assumptions.

1.2.3 Discrete-Layer Theories

In an effort to overcome the deficiencies of the equivalent single layer approach, discrete-layer (or layerwise) theories have been proposed (e.g., [15-18]). These theories are based on a unique displacement field for each layer and enforce interlaminar continuity of displacements and sometimes of transverse stresses, as well. They predict excellent global and local distributions of in-plane and out-of-plane displacements and stresses. The related thermal model assumes a piecewise, or layerwise, variation of the temperature, generating accurate temperature distributions [19]. The major shortcoming in these theories is their large computational expense, for as the number of layers increases, the number of degrees of freedom increases proportionally.

1.2.4 Linear Flux Theory

This approach employs the assumption that the fluxes vary linearly as the dependent variables over the element [20]. Finite element equations derived for thermal and structural analysis yield finite element in integral form that are different from those appearing in conventional formulations. The advantage of this method consists in a reduc

first c

1.2.5

was c

lamin

thick

numb

stress

very

resea

impr

displ

creat

bend

defle

gene

orde

trans

cond

elem

reduced computational time, but the accuracy of the result is comparable to that of the first order shear deformation theories.

1.2.5 Zig-Zag Laminate Theory

A new class of laminate theory, called here the first order zig-zag theory (FZZT), was developed by DiSciuva in the mid 1980's [21,22]. The in-plane displacements in a laminate are assumed to be piecewise (layerwise) linear and continuous through the thickness, yet the total number of degrees of freedom is only five (not dependent on the number of layers). This is accomplished by analytically satisfying the transverse shear stress continuity conditions at each interface in the laminate. This theory was shown to be very accurate for many cases, especially symmetric laminates.

On the basis of the concept introduced in [21,22], DiSciuva as well as other researchers made significant improvements to the FZZT [23-30]. The primary improvement was achieved by superimposing a piecewise linear variation of in-plane displacements on a continuous cubic function of the transverse coordinate [24-29], creating a displacement field that can better account for the warping that occurs during bending of asymmetric laminates.

However, these theories have the unfortunate limitation that the transverse deflection degree of freedom w_0 is required to be C^1 continuous. Averill proposed a generalized form of the first-order zig-zag theory [23] and a generalized form of the high-order zig-zag theory [24] for beams to alleviate the requirement of C^1 continuity on the transverse deflection. A new variable was introduced along with an associated constraint condition, which was enforced by employing an interdependent (anisoparametric) element interpolation scheme and the penalty method.

Yip and Averill developed a model for laminated beams [25] and plates [26] that combined the benefits of the discrete-layerwise and high-order zig-zag theories. This model allows the laminate to be subdivided into a number of sublaminates, with each sublaminate containing several, even many, physical layers. A zig-zag kinematic assumption is used within a sublaminate. Each finite element represents one sublaminate, and, if cast in the form of a four-noded quadrilateral (for beams) or an eight-noded brick (for plates), refinement of a model by through-thickness discretization can be achieved without the use of any special constraints. When only one sublaminate is used through the entire thickness of the laminate, nodal degrees of freedom are three translations and two rotations. Nevertheless, when multiple elements (sublaminates) are needed to model a laminate, additional shear stress degrees of freedom are present, making the element unsuitable for implementation into many of the current commercial finite element codes.

Cho and Averill [31], presented a refined plate theory based on sublaminate linear zig-zag kinematics and a new 3-D finite element based on that theory. The new element contains eight nodes of which each has only five engineering degrees of freedom -- three translations and two rotations. The element has shown to be accurate, simple to use, and compatible with the requirements of commercial finite element codes.

1.3 Objective of the Present Study

The first task of this research project is to introduce a new thermal lamination theory for evaluating the temperature distribution in composite plates and sandwich panels. This theory applies to heat transfer problems the zig-zag sublaminate concept described above. Accordingly, a layerwise through-thickness distribution of temperature is assumed, and the interior DOFs are eliminated by analytically satisfying the condition of transverse flux continuity at each ply interface. The finite element model is developed in a sublaminate version that allows the laminate to be subdivided into a number of sublaminates, with each sublaminate containing a number of physical layers.

The second objective of the work is to combine the presented thermal theory with the structural theory developed in [31], in order to realize a finite element formulation for accurate thermal stress analysis of composite laminates. With respect to others theories, the presented formulation is expected to show superior results in predicting the distribution of temperatures, displacements and stresses in thin or thick laminated plates wherein the plies have dissimilar thermal and/or structural properties.

1.4 Organization of the Thesis

In Chapter Two, the basic concepts of the new zig-zag sublaminate thermal theory are presented and the possible forms of the finite element models based on the theory are investigated. A four-noded two-dimensional sublaminate finite element model is developed and implemented in a Fortran code. Several numerical results are reported to demonstrate the effectiveness of the theory.

In Chapter Three, to test the validity of the theory in the presence of material non-linearity, the thermal conductivity coefficients of the individual layers are assumed to be a quadratic function of the temperature. This has required non-linear capabilities to be implemented in the model and in a new Fortran code adopting the incremental Newton-Raphson technique. Numerical outcomes are used to show the high degree of adherence of the model to the physical phenomenon.

This

deve

impl

Ther

coda

prop

In Chapter Four, a description of the zig-zag structural plate theory is provided. This task is accomplished neglecting some details related to the mathematical development and focusing the attention on the necessary changes required by the implementation of the thermal stress capabilities.

In Chapter Five, the method of the equivalent nodal force vector is explained. Then, the solutions of numerous thermal stress analyses, obtained by use of a Fortran code developed from the model, are compared with the exact ones to show that the proposed procedure is efficient and accurate.

Final conclusions are given in Chapter Six.

CHAPTER 2 THERMAL LAMINATION THEORY

2.1 Formulation of the Theory

It is assumed that the laminate is composed of N_T perfectly bonded layers. The thickness and material thermal properties may vary arbitrarily from layer to layer. The laminate can be modeled as M sublaminates, with each sublaminate containing N_m layers, where m is the sublaminate number. Mathematically, this is represented as:

$$N_T = \sum_{m=1}^{M} N_m \tag{1}$$

In order to facilitate the development of the theory, all expressions in the following derivation pertain to the m-th sublaminate. The sublaminate number designation m is omitted for brevity. The reference surface of the m-th sublaminate is assumed to be the bottom surface of the sublaminate, as shown in Figure 1.

The balance equation for thermal conduction within a single orthotropic homogeneous layer can be written as:

$$\rho c \frac{\partial T}{\partial t} = \nabla \cdot \overline{q} + Q \tag{2}$$

where:

$$q_i = -k_i \frac{\partial T}{\partial x_i} \tag{3}$$

is the flux in the i-th direction (i=1,2,3), ρ is the density, c is the specific heat, k_i is the thermal conductivity in the i-th direction, and Q is the internal heat generated per unit volume.

When the material principle direction is oriented at an angle θ to the coordinate axis x_1, the thermal conductivity coefficients k_1 and k_2 are calculated as follows. If k_L is the thermal conductivity of the lamina in the direction of the fibers and k_T is the transversal one, then:

$$\begin{aligned} k_1 &= k_L \cdot \cos^2\theta + k_T \cdot \sin^2\theta \\ k_2 &= k_L \cdot \sin^2\theta + k_T \cdot \cos^2\theta \end{aligned} \tag{4}$$

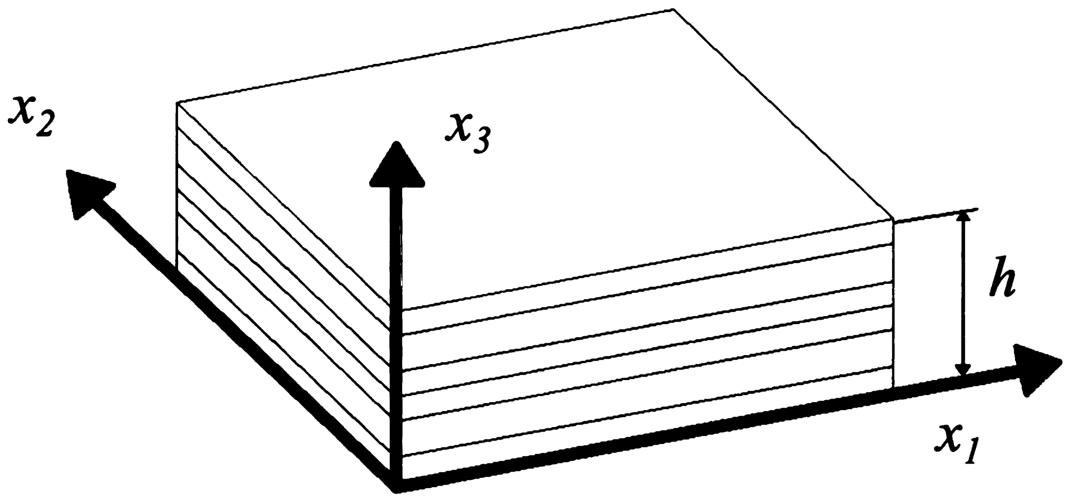


Figure 1 ***Geometry and coordinate system used in the thermal sublaminate theory.***

The set of essential boundary conditions is of the form:

$$T = \tilde{T} \quad \text{on} \quad S_1 \tag{5}$$

and the set of natural boundary conditions is of the form:

$$k_1 \frac{\partial T}{\partial x_1}\hat{n}_1 + k_2 \frac{\partial T}{\partial x_2}\hat{n}_2 + k_3 \frac{\partial T}{\partial x_3}\hat{n}_3 - \tilde{q} + h(T - T_\infty) = 0 \quad \text{on} \quad S_2 \tag{6}$$

where h is the coefficient of convection and T_∞ is the temperature of the external fluid.

The conditions of temperature and normal flux continuity at interfaces between adjacent plies are:

$$\text{at} \quad x_3 = x_{3k} \quad \begin{cases} T^{(k)}\Big|_{x_3 = x_{3k}} = T^{(k+1)}\Big|_{x_3 = x_{3k}} \\ k_3^{(k)} \dfrac{\partial T^{(k)}}{\partial x_3}\Bigg|_{x_3 = x_{3k}} = k_3^{(k+1)} \dfrac{\partial T^{(k+1)}}{\partial x_3}\Bigg|_{x_3 = x_{3k}} \end{cases} \tag{7}$$

for k = 1 to N-1, where N is the number of plies in the sublaminate. $k_3^{(k)}$ is the thermal conductivity in the x_3-direction of the $(k)^{th}$ ply.

The simplest zig-zag temperature distribution that can be assumed is a piecewise linear one, where for the k-th layer:

$$T^{(k)}(x_1, x_2, x_3, t) = T_0(x_1, x_2, t) + x_3 T_1(x_1, x_2, t) + \sum_{i=1}^{k-1} (x_3 - x_{3i}) \hat{T}_i(x_1, x_2, t) \tag{8}$$

In Eq. (8), T_0 is the temperature and T_1 is the thickness gradient of the temperature at the bottom of the sublaminate. The first set of interface conditions in (7) is identically satisfied by the assumed temperature distribution in (8).

Following the example of the zig-zag structural theories, the DOFs $\hat{T}_i$ can be eliminated by enforcing continuity of the normal flux at ply interfaces, the second condition in (7).

This can be performed analytically, resulting in the following expression:

$$\hat{T}_i = a_i T_1 \tag{9}$$

where:

$$a_i = \left(\frac{k_3^{(i)}}{k_3^{(i+1)}} - 1 \right) \cdot \left(1 + \sum_{j=1}^{i-1} a_j \right) \tag{10}$$

Now, the theory contains only the DOFs T_0, T_1 for each sublaminate. If the entire thickness of a structure is modeled using a single sublaminate, then the present theory resembles other commonly used thermal lamination theories, except that here a piecewise linear (or zig-zag) distribution of temperature is assumed. Note that if the transverse component of thermal conductivity is the same from ply to ply, then the parameters a_k are all zero, according to (10). In this case, the present theory reduces to one in which the assumed through-thickness variation of temperature is linear through the entire sublaminate.

When multiple sublaminates are used, it is necessary to enforce continuity of temperature across the sublaminate interfaces. This can be facilitated by replacing T_0, T_1 with the new DOFs T_b, T_t, the temperature at the bottom and top surfaces, respectively, of the sublaminate:

$$T_b = T^{(1)}\Big|_{x_3=0}, \qquad T_t = T^{(N)}\Big|_{x_3=h} \tag{11}$$

where h is the thickness of the sublaminate.

The temperature distribution within a sublaminate is now written as:

$$T^{(k)} = T_b\left(1 - \Phi_k\right) - T_t \Phi_k \tag{12}$$

where:

$$\Phi_k(x_3) = \frac{x_3 + \sum_{i=1}^{k-1}[x_3 - x_{3i}]a_i}{h + \sum_{i=1}^{N-1}[h - x_{3i}]a_i} \tag{13}$$

is a zig-zag through-thickness shape function that satisfies all interface conditions within the sublaminate. Note that continuity of flux across sublaminate interfaces is not enforced.

The heat transfer model described above is the simplest possible form of zig-zag thermal lamination theory, and is apparently the first of its kind. It is not difficult to envision higher-order forms of the model similar to the cubic zig-zag models used in structural analysis [24,25,28].

2.1.1 Demonstration of Zig-Zag Effects

In this section a numerical example is presented in order to demonstrate the types of temperature distributions that exist in laminates. Consider a plate having a unit thickness h in the x_3 direction and subjected to a uniform distribution of temperature $T = 1$ at the top surface, and $T = 0$ at the bottom surface. As a consequence of these boundary conditions the value of the inplane dimension L has no effect on the results. Further, the temperature distribution described in Eq. (12) represents the exact analytical solution for this case. A three-ply laminate is examined with the top and bottom plies consisting of the same material and the middle ply composed of a different one. The material thermal conductivities in the x_3 direction k_3 are varied to demonstrate their effect on laminate response. This variation is performed in such a way that the overall laminate thermal conductivity, denoted $\bar{K}_3$, remains constant. In our case:

wher

respe

ratio

the m

abov

thick

therm

varia

temp

repre

conti

norm

first

lamin

$$\bar{K}_3 = \int_0^h k_3\,dz = \sum_{i=1}^{3} k_3^i h_i \tag{14}$$

where k_3^i and h_i are the thermal conductivity in the x_3 direction and the thickness, respectively, of the i-th ply. The plies are of equal thickness, so h_i = 1/3 for i = 1,2,3. The ratio of thermal conductivity properties in the material of the first and third layer, k_3^1, and the material of the second layer, k_3^2, is defined as:

$$\beta = \frac{k_3^1}{k_3^2} \tag{15}$$

With β and $\bar{K}_3$ specified, the values of k_3^1 and k_3^2 are easily determined from the above equations. In the present case, $\bar{K}_3$ is taken to be 3.5, and β is varied. The through-thickness variation of the temperature is shown in Figure 2 for different values of the ply thermal conductivity ratio β. It can be that seen when $\beta = 1$, the through-thickness variation of the temperature is linear. However, when β is not equal to one, the temperature field exhibits a distinct piece-wise continuous shape that cannot be represented by even a high-order smooth polynomial. In these cases, a piece-wise continuous function is required to capture the essential aspects of the temperature field.

In Table 1, for the four values of β considered in Figure 2, the temperatures, normalized by the result that occurs when $\beta = 1$, are reported at the top surface of the first and second layer. It is interesting to note that the temperature distribution in the laminate depends very strongly upon the value of β.

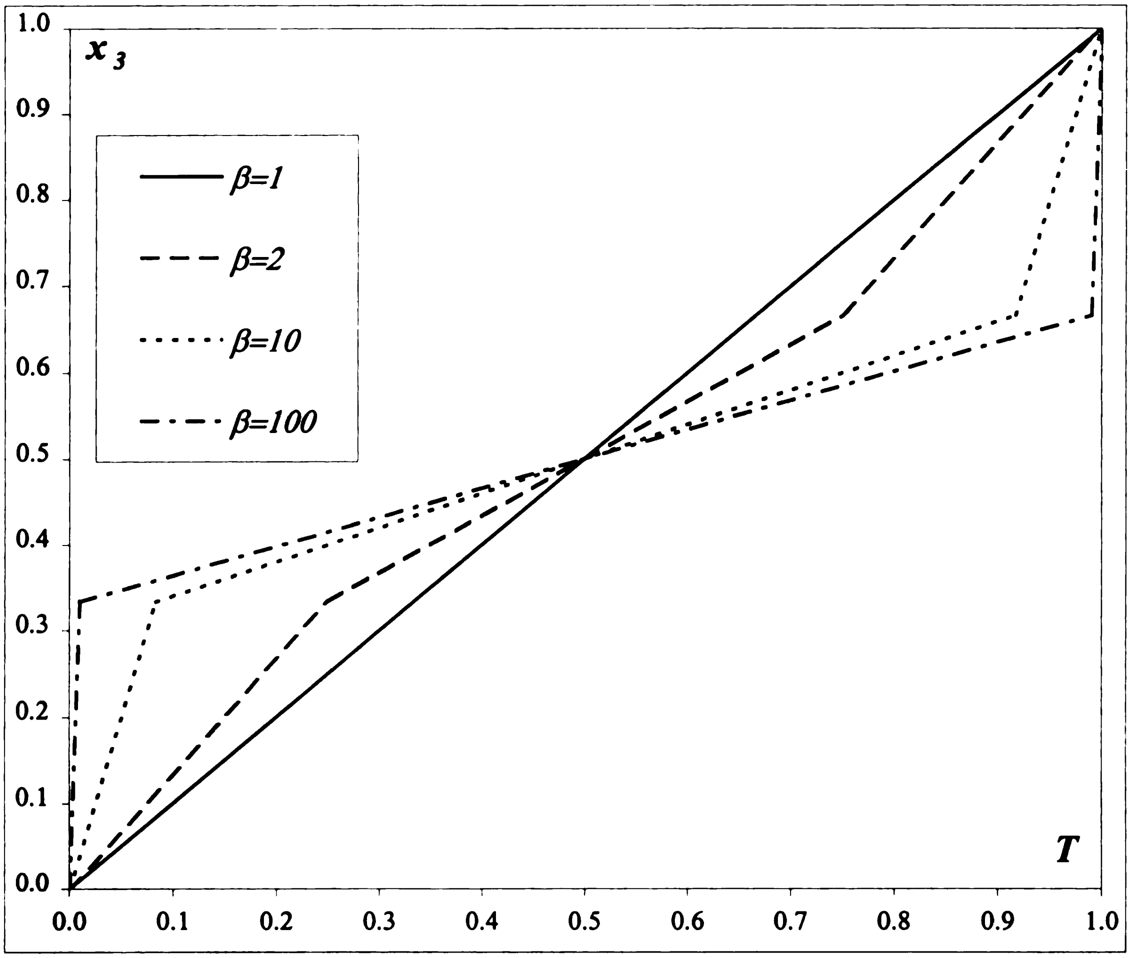


Figure 2 ***Through-thickness variation of the temperature of a three-ply laminate subjected to $T=1$ at the top surface and $T = 0$ at the bottom.***

Table 1. ***Variation of temperature versus ply thermal conductivity ratio.***

β	$\frac{T(\beta)}{T(\beta=1)}$ at x_3 =1/3	$\frac{T(\beta)}{T(\beta=1)}$ at x_3 =2/3
1	1.0000000	1.0000000
2	0.7500000	1.1250000
10	0.2500000	1.3750000
100	0.0294118	1.4852941

2.2 Finite Element Model

2.2.1 Possible Forms

A finite element model based on the above theory could take many forms. In the case where the entire laminate thickness is modeled using a single sublaminate, the simplest elements would take the form of a three-noded triangle or four-noded quadrilateral, with two DOFs per node, T_0, T_1 or T_b, T_t, as shown in Figure 3.

Alternatively, a three-dimensional finite element model of the forms shown in Figure 4a can be developed with one DOF per node. This approach has the additional advantage that multiple elements can be used through the thickness of a laminate, if needed, to further increase the accuracy of the approximation (see Figure 4b). The model can then be considered a multi-sublaminate model with one degree of freedom per node, see Figure 5.

For the purpose of demonstrating and assessing the new thermal lamination theory, a four-noded two-dimensional sublaminate finite element model has been developed, as shown in Figure 6.

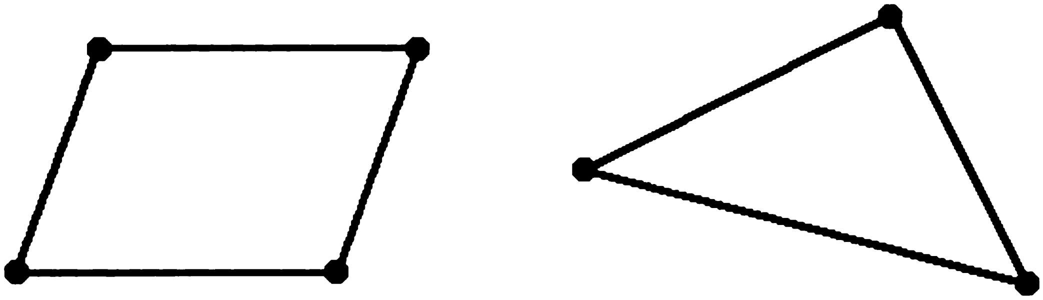
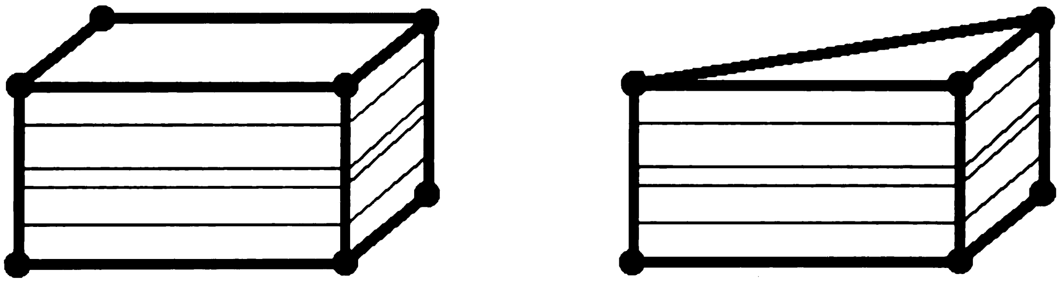
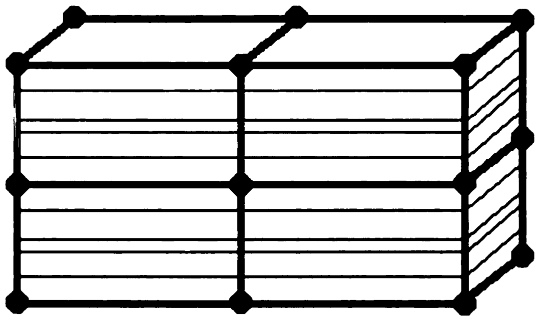


Figure 3 ***Quadrilateral and triangular finite elements based on the zig-zag thermal lamination theory. Each node contains two computational DOFs.***



(a)



(b)

Figure 4 ***(a) Eight-noded brick and six-noded wedge three-dimensional finite elements based on the zig-zag thermal lamination theory. Each node contains one computational DOF - the temperature at that node. (b) An example of discretizing the thickness of a laminate using multiple elements (sublaminates).***

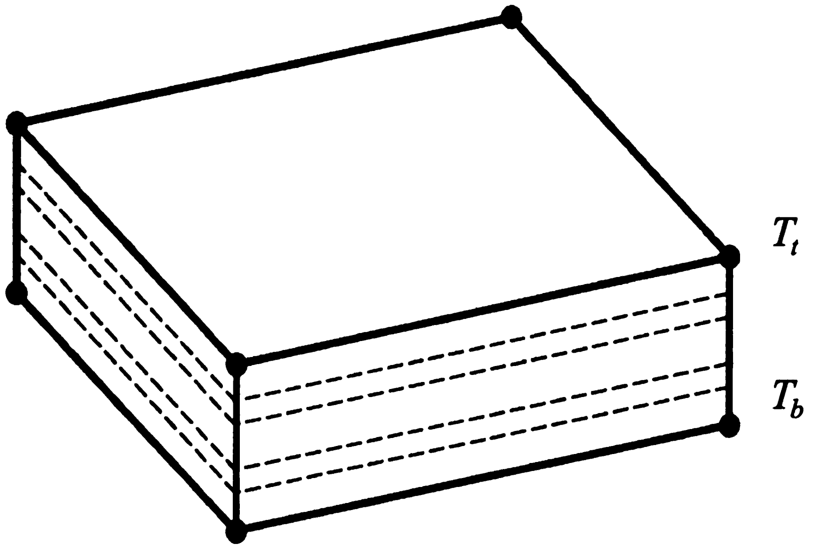


Figure 5 ***Element topology and nodal degrees of freedom for the three-dimensional form of the thermal model.***

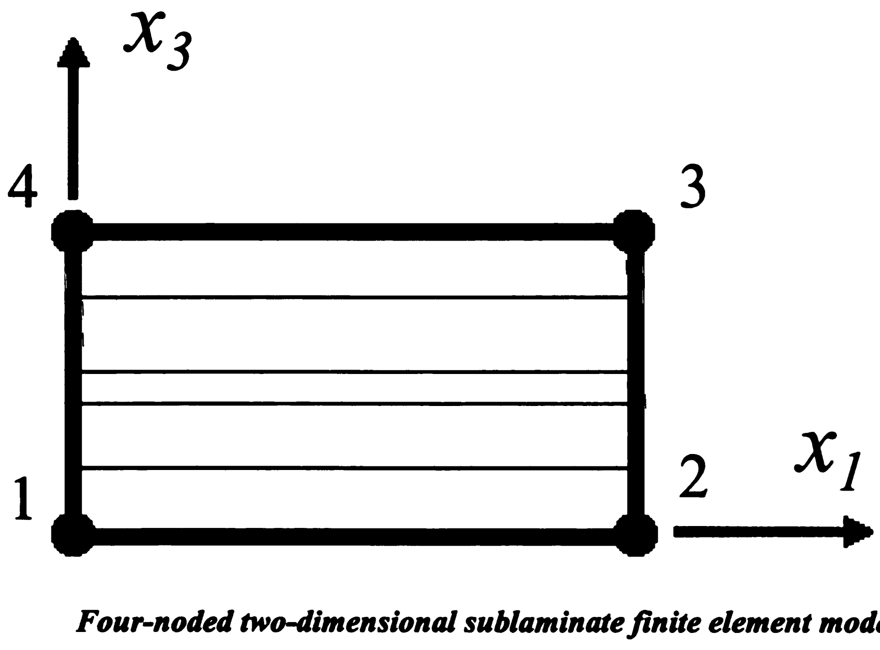


Figure 6 ***Four-noded two-dimensional sublaminate finite element model***

2.2.2 Four-Noded Layered Rectangular Thermal Element

If the temperature does not vary in the x_2-direction, the balance equation for thermal conduction takes the form:

$$-\frac{\partial}{\partial x_1}\left(k_1 \frac{\partial T}{\partial x_1}\right)-\frac{\partial}{\partial x_3}\left(k_3 \frac{\partial T}{\partial x_3}\right)=\rho\, c \frac{\partial T}{\partial t}-Q \tag{16}$$

Ignoring transient effects and internal heat generation, the following equation governs our problem:

$$-\frac{\partial}{\partial x_1}\left(k_1 \frac{\partial T}{\partial x_1}\right)-\frac{\partial}{\partial x_3}\left(k_3 \frac{\partial T}{\partial x_3}\right)=0 \tag{17}$$

The corresponding weak statement is:

$$\int_{\Omega^e}\left\{k_1\left(\frac{\partial \delta T}{\partial x_1}\frac{\partial T}{\partial x_1}\right)+k_3\left(\frac{\partial \delta T}{\partial x_3}\frac{\partial T}{\partial x_3}\right)\right\}dx_1 dx_3=\int_{\Gamma^e}\left[q_n\, \delta T\right]ds \tag{18}$$

where:

$$q_n=k_1\frac{\partial T}{\partial x_1}n_1+k_3\frac{\partial T}{\partial x_3}n_3 \tag{19}$$

is the flux normal to boundary Γ^e.

The following approximation scheme is employed within the four-noded rectangular element:

$$T^{(k)}=\sum_{j=1}^{4}T_j\, N_j^k\left(x_1,x_3\right) \tag{20}$$

where:

$$
\begin{aligned}
N_1^k &= \left[1-\Phi_k\left(x_3\right)\right]\cdot\Psi_1\left(x_1\right)\\
N_2^k &= \left[1-\Phi_k\left(x_3\right)\right]\cdot\Psi_2\left(x_1\right)\\
N_3^k &= \Phi_k\left(x_3\right)\cdot\Psi_2\left(x_1\right)\\
N_4^k &= \Phi_k\left(x_3\right)\cdot\Psi_1\left(x_1\right)
\end{aligned}
\tag{21}
$$

Ψ_1 and Ψ_2 are the Lagrangian linear interpolation functions adopted to approximate T long x_1:

$$
\begin{aligned}
\Psi_1(\xi) &= \frac{1}{2}\ (1-\xi)\\
\Psi_2(\xi) &= \frac{1}{2}\ (1+\xi)
\end{aligned}
\tag{22}
$$

And Φ_k is defined in equation (13) to be:

$$
\Phi_k\left(x_3\right)=\frac{x_3+\sum_{i=1}^{k-1}\left[x_3-x_{3i}\right]a_i}{h+\sum_{i=1}^{N-1}\left[h-x_{3i}\right]a_i}
\tag{23}
$$

The resulting finite element model is:

$$
\sum_{j=1}^{4} K_{ij}^e T_j^e = Q_i^e
\tag{24}
$$

where:

$$
K_{ij}^e=\sum_{k=1}^{N}\left\{\int_{A_k}\left[k_1^k\left(\frac{\partial N_i^k}{\partial x_1}\frac{\partial N_j^k}{\partial x_1}\right)+k_3^k\left(\frac{\partial N_i^k}{\partial x_3}\frac{\partial N_j^k}{\partial x_3}\right)\right]dx_1 dx_3\right\}
\tag{25}
$$

and

$$
Q_i^e=\int_{\Gamma^e}\left[q_n N_i^k\right]ds
\tag{26}
$$

The integration along the x_1 direction is performed using Gauss-Legendre quadrature. The integration along x_3 direction utilizes Newton-Cotes quadrature using 3 base points per layer.

The final expression for calculating the coefficients K_{ij}^{e} is:

$$K_{ij}^{k}=\frac{h^{k}}{6}\cdot JK_{1}^{k}\cdot\left[\Phi_{k}^{i}\left[x_{3}^{Bottom}(k)\right]\cdot\Phi_{k}^{j}\left[x_{3}^{Bottom}(k)\right]\right]\cdot\left[2\frac{\partial\Psi_{i}}{\partial\xi}\frac{\partial\Psi_{j}}{\partial\xi}\right]+$$
$$+\frac{h^{k}}{6}\cdot JK_{1}^{k}\cdot\left[4\Phi_{k}^{i}\left[z_{m}\right]\cdot\Phi_{k}^{j}\left[z_{m}\right]+\Phi_{k}^{i}\left[x_{3}^{Top}(k)\right]\cdot\Phi_{k}^{j}\left[x_{3}^{Top}(k)\right]\right]\cdot\left[2\frac{\partial\Psi_{i}}{\partial\xi}\frac{\partial\Psi_{j}}{\partial\xi}\right]+ \quad (27)$$
$$+h^{k}JK_{1}^{k}\cdot\left[\Psi_{i}\left(\xi_{p=1}\right)\cdot\Psi_{j}\left(\xi_{p=1}\right)+\Psi_{i}\left(\xi_{p=2}\right)\cdot\Psi_{j}\left(\xi_{p=2}\right)\right]\cdot\left[\frac{\partial\Phi_{k}^{i}}{\partial x_{3}}\frac{\partial\Phi_{k}^{i}}{\partial x_{3}}\right]$$

where:

$$\Phi_{k}^{i}=\begin{cases}(1-\Phi_{k}) & \text{if} \quad i=1,2\\ \Phi_{k} & \text{if} \quad i=3,4\end{cases}$$

$$\Phi_{k}^{j}=\begin{cases}(1-\Phi_{k}) & \text{if} \quad j=1,2\\ \Phi_{k} & \text{if} \quad j=3,4\end{cases}$$

$$z_{m}(k)=\frac{x_{3}^{Top}(k)+x_{3}^{Bottom}(k)}{2} \quad (28)$$

J = value of the Jacobian for the change in coordinate system from x_1 to ξ

2.3 The Computer Program "Therm2D"

In order to apply the developed formulation for solving thermal conduction problems a computer program has been realized. The computer language chosen is Fortran77, generally recognized from the scientific community as one of the most appropriate for writing finite element code. The code is reported in Appendix A. Several comments have been inserted in the code to facilitate the reader's understanding. In particular, a detailed heading section explains the function of the main variables.

Input. The input data file is very simple to build, since some time wasting operations like writing the Boolean matrix are automatically accomplished by the program. The material properties and the height must be specified for each layer composing the laminate.

Output. The output file in its first part summarizes the characteristics of the analysis: element data, mesh data, Boolean matrix, nodal boundary conditions, number and property of each layers element by element. It provides a complete output of the analysis avoiding the necessity of printing also the input data file. Moreover, it makes possible to accurately verify of the correctness of the input file. Subsequently, the thermal solution at each node in the mesh is written. The last section of the file provides the temperatures element by element at each layer interface along the two vertical sides. Since the temperatures inside a single layer vary linearly, from the given information the solution in every point of the domain can be calculated.

2.4 The Computer Program "Sinload"

In order to verify the effectiveness of the model the solutions coming from the code had to be compared to the exact solution. Then, another Fortran code, called "Synload", has been realized. It is based on the publication [32], where the problem of a laminated plate subjected to a sinusoidal thermal load is solved exactly. The code is reported in Appendix B. The computer language is Fortran77, like for the other codes.

Structure. In [32] are provided the solution equations for the differential equation governing each layer. The program solves the system of equations determined by imposing the boundary and interface conditions. The results are the unknown coefficients of the solution equations. Several comments have been inserted in the code to facilitate the reader's understanding.

Input. The input data file is simpler than in the previous case, since no information regarding the meshing of the domain is required. What needs to be inserted are: the material properties and the height of each layer, the boundary conditions and the horizontal x_1 coordinates where the solution is desired.

Output. The output file initially reports the characteristics of the analysis. Then, it supplies the temperatures at each layer interface for the required x_1 coordinates.

2.5 Numerical Results

In this section the model is used to solve linear heat conduction problems to demonstrate its applicability for thermal analysis of layered composites. For convenience, all values of temperature are normalized by a reference temperature so as to non-dimensionalize the results. Further, all dimensions and coordinates are normalized by the total thickness of the laminate, h.

We have studied three types of laminates: a symmetric 8-layer laminate, an unsymmetric 6-layer laminate and a laminate similar to the Shuttle thermal protection system. In each case results are presented for two different aspect ratios of the composite plate: 4 and 10.

All computational experiments, except the last one, consider a heat conduction problem for which an analytical solution can be computed by use of the program "Sinload". It is the steady-state analysis of a laminated plate subjected at the top and bottom to a sinusoidal distribution of normalized temperature of the type:

$$T_{top}(x_1) \quad = T_t \cdot \sin\left(\frac{\pi \cdot x_1}{L}\right)$$
$$T_{bottom}(x_1) = T_b \cdot \sin\left(\frac{\pi \cdot x_1}{L}\right) \tag{29}$$

On the two lateral sides, at $x_1 = 0$ and $x_1 = L$, a uniform temperature $T = 0$ is imposed. The value of the normalized temperatures T_t and T_b have been taken equal to 1 and 0.1 respectively. For the case of the symmetric 8-layer laminate the model with the applied boundary conditions is shown in the figure 7.

Figure 7 ***BC's for the symmetric 8-layer laminate.***

For the Shuttle thermal system an additional case is considered where three surfaces are insulated: the top and the two sides.

It should be noted that when a uniform distribution of temperature is imposed on the top and on the bottom of the laminated plate, the current model yields the exact solution, with just one element through the thickness. Hence, in the examples presented, only the more complex load schemes are considered.

2.5.1 Symmetric 8 Layer Laminate

Aspect ratio l/h = 10. In this test we have considered a steady-state heat conduction problem in an 8-layer symmetric laminate composed of two materials whose thermal properties are shown in Table 2.

Table 2. ***Material properties used in the numerical analyses.***

Material	k_1 **– Conductivity in the** x_1 **direction,** $\frac{W}{m \cdot K}$	k_3 **– Conductivity in the** x_3 **direction,** $\frac{W}{m \cdot K}$
Material 1	10.0	5.0
Material 2	1.0	0.5

The lamination scheme is (1/2/1/2/2/1/2/1). For the comparison between the exact solution and our finite element model, the temperature is reported along the thickness at $x_1 = 2.5$ and $x_1 = 5$. Eight different uniform meshes are used: 8x8, 8x4, 8x2, 8x1, 4x8, 4x4, 4x2 and 4x1, where the two numbers indicate respectively the subdivisions in x_1 and x_3 directions. For example, since the laminate contains 8 layers of equal thickness, it follows that in a mesh with 4 divisions along the thickness each element includes 2 layers.

Normalized temperature distributions are shown in tabular form in Tables 3 and 4 and in graphic form in Figures 8 and 9.

A close agreement between the exact analytical solution and the new theory is obtained with the 4 by 1 mesh, and convergence toward the exact solution is achieved as the mesh is refined. The linear solution is provided for the sake of comparison.

Table 3. ***Temperature distribution along the thickness at x_1 = 2.5 for the 8 layer symmetric plate with aspect ratio of 10. (Temperatures are normalized by the exact solution.)***

x_3	Exact	8x8	8x4	8x2	8x1	4x8	4x4	4x2	4x1
0.000	*0.070711*	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
0.125	*0.083926*	0.99981	1.00135	1.00566	1.01486	0.99925	1.00086	1.00533	1.01487
0.250	*0.217500*	0.99931	0.99917	1.01751	1.05659	0.99717	0.99710	1.01610	1.05660
0.375	*0.231230*	0.99926	1.00074	1.01630	1.05640	0.99716	0.99875	1.01485	1.05641
0.500	*0.372420*	0.99944	0.99936	0.99863	1.04426	0.99776	0.99772	0.99697	1.04428
0.625	*0.514760*	0.99953	1.00101	1.01847	1.03648	0.99815	0.99975	1.01782	1.03649
0.750	*0.529870*	0.99957	0.99947	1.01818	1.03422	0.99825	0.99823	1.01761	1.03423
0.875	*0.689930*	0.99994	1.00149	1.00280	1.00392	0.99980	1.00142	1.00277	1.00393
1.000	*0.707107*	1.00000	1.00000	0.99999	0.99999	1.00000	1.00000	1.00000	1.00000

Table 4. ***Temperature distribution along the thickness at x_1= 5 for the 8 layer symmetric plate with aspect ratio of 10. (Temperatures are normalized by the exact solution.)***

x_3	Exact	8x8	8x4	8x2	8x1	4x8	4x4	4x2	4x1
0.000	*0.10000*	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
0.125	*0.11869*	0.99983	1.00135	1.00566	1.01487	0.99924	1.00084	1.00532	1.01487
0.250	*0.30759*	0.99932	0.99922	1.01754	1.05660	0.99717	0.99714	1.01610	1.05660
0.375	*0.32701*	0.99927	1.00076	1.01632	1.05641	0.99716	0.99872	1.01484	1.05641
0.500	*0.52668*	0.99943	0.99937	0.99865	1.04428	0.99776	0.99772	0.99697	1.04428
0.625	*0.72798*	0.99953	1.00104	1.01849	1.03649	0.99815	0.99974	1.01782	1.03649
0.750	*0.74934*	0.99957	0.99951	1.01821	1.03424	0.99827	0.99824	1.01762	1.03424
0.875	*0.97571*	0.99995	1.00151	1.00281	1.00393	0.99980	1.00141	1.00277	1.00393
1.000	*1.00000*	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000

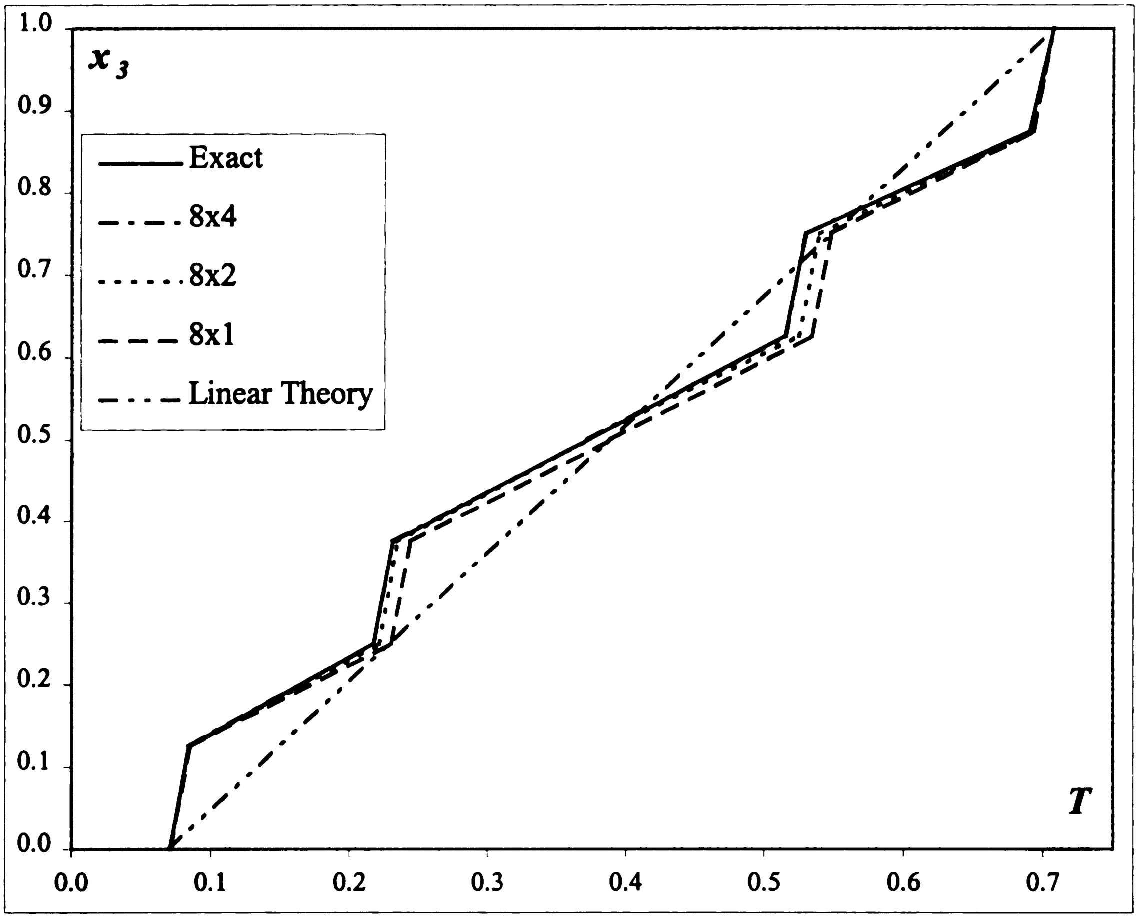


Figure 8 ***Through-thickness distribution of temperature at $x_1 = 2.5$ for an 8-layer symmetric laminate with aspect ratio of 10.***

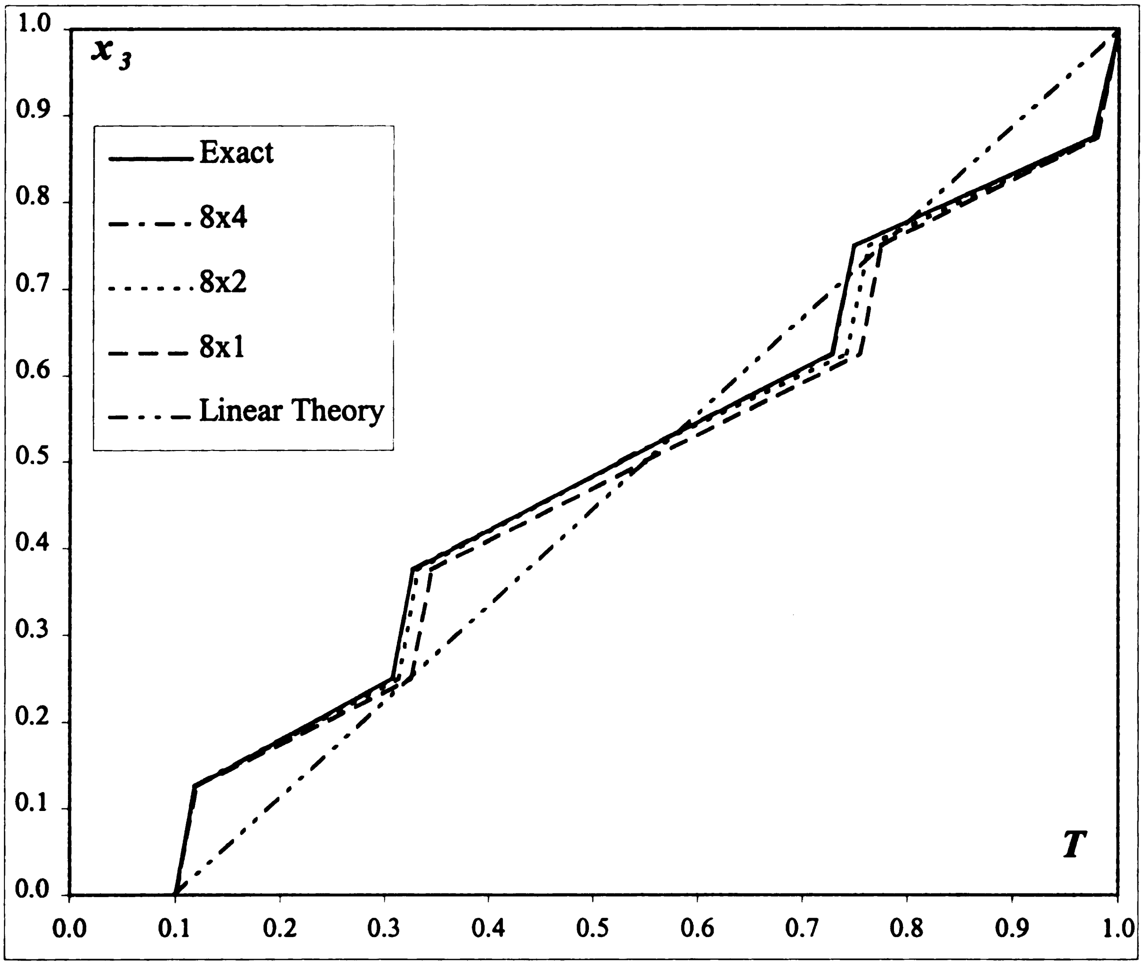


Figure 9 ***Through-thickness distribution of temperature at $x_1 = 5$ for an 8-layer symmetric laminate with aspect ratio of 10.***

Aspect ratio $l/h = 4$. A smaller aspect ratio produces a larger gradient of temperature inside the plate in the x_1 direction. So this case tests the limits of our theoretical approach. Materials, lamination scheme and the imposed sinusoidal distribution of temperature are the same as in the previous analysis; the length-to-thickness aspect ratio is taken to be 4.

Results are shown in Figure 10 only at the center of the plate, since the level of agreement does not change significantly at different values of the x_1 coordinate. In this case the solution obtained with an 8 by 4 mesh follows almost exactly the analytical solution. The solution based on the 8 by 2 keeps a good degree of accuracy, instead the 8 by 1 meshes is able to give only a reasonable approximation.

2.5.2 Unsymmetric 6 Layer Laminate

Here we consider an unsymmetric 6-layer laminate composed of the same two materials as in the previous analysis. The applied sinusoidal distribution of temperature is also the same, but the lamination scheme is (1/2/2/1/1/1).

For aspect ratios of 10 and 4, respectively, results are reported in Figures 11 and 12. An analysis of the presented graph confirms that the present approach works well for unsymmetrical laminates. Some degree of approximation is registered for the 8x1 mesh in the plate having an aspect ratio of 4.

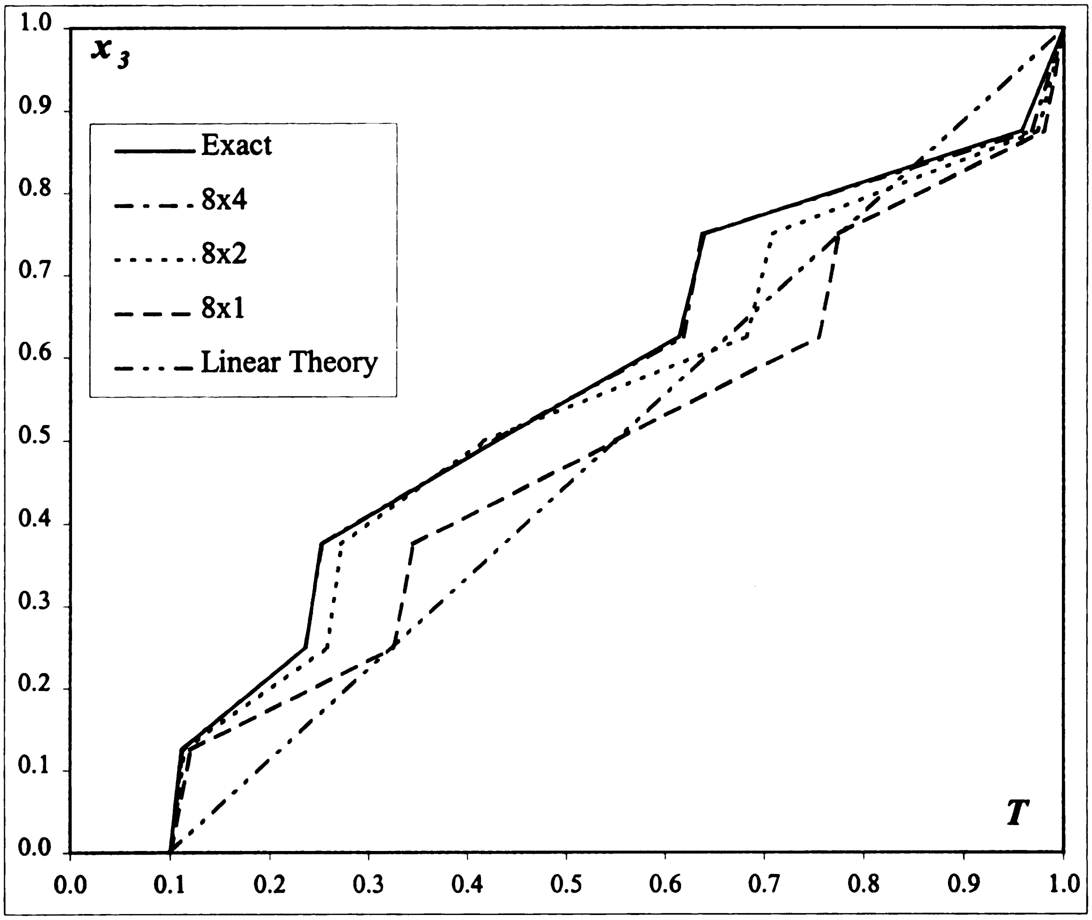


Figure 10 ***Through-thickness distribution of temperature at x_1 = 2 for an 8-layer symmetric laminate with aspect ratio of 4.***

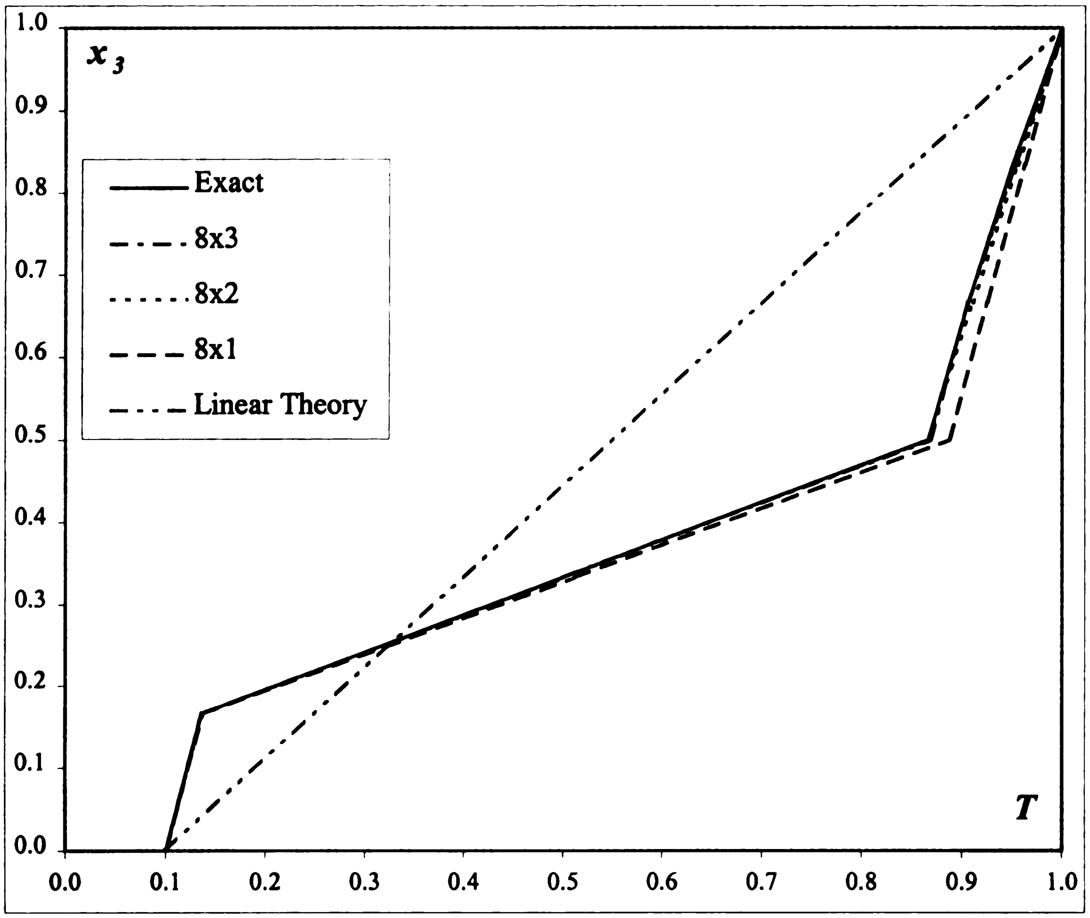


Figure 11 ***Through-thickness distribution of temperature at x_1 = 5 for a 6-layer unsymmetric laminate with aspect ratio of 10.***

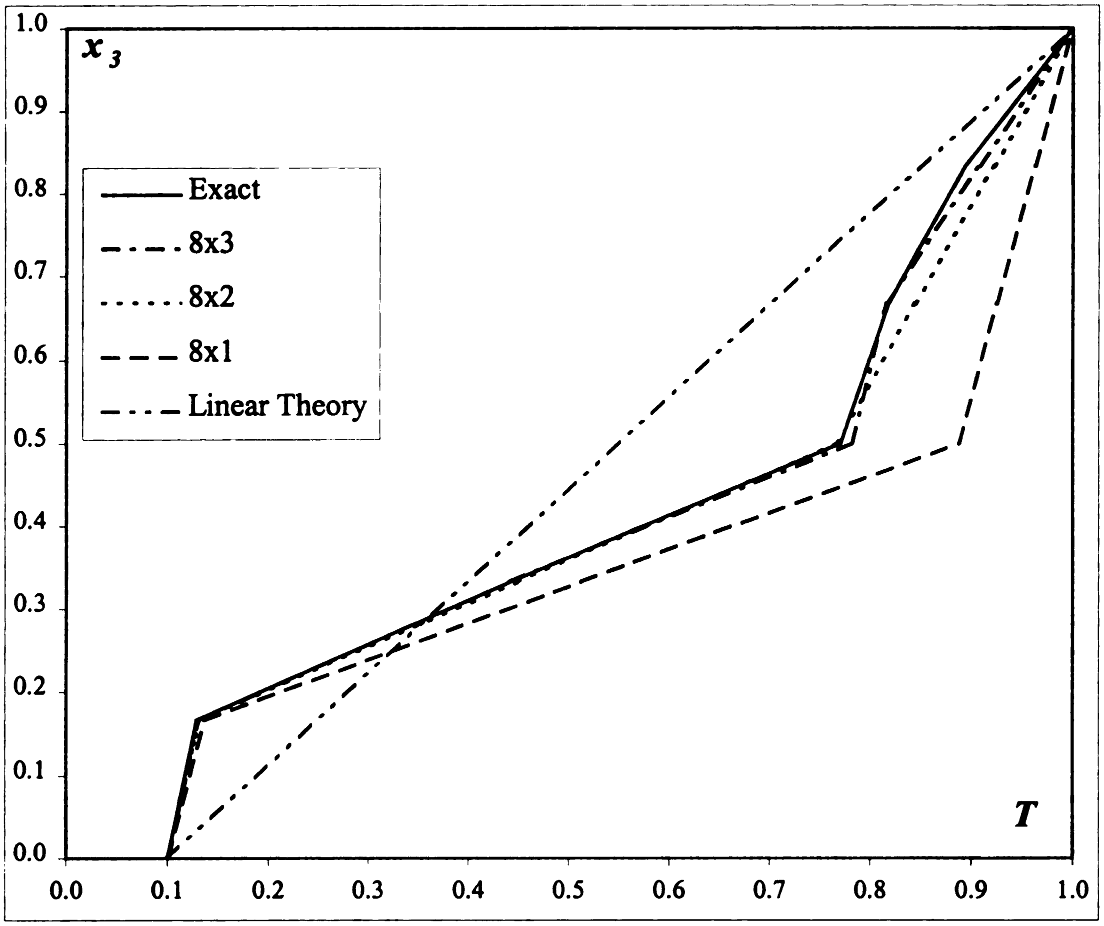


Figure 12 ***Through-thickness distribution of temperature at $x_1 = 2$ for a 6-layer unsymmetric laminate with aspect ratio of 4.***

2.5.3 Shuttle Thermal Protection System

The Space Shuttle thermal protection system is composed of five different materials, whose thermo physical properties are assumed to be constant and are given in Table 5 (adopted from [12]). This case tests the accuracy of the present model when there is a large difference in the values of thermal conductivity in adjacent plies. The thickness of the RSI layer is much greater than all the other layers, so we will focus on the results for the last four layers (RTV / FELT / RTV / ALUMINIUM) in order to get a better view of the quality of the solution. An image of the model, for the aspect ratio $L/h = 4$, is given in Figure 13.

Table 5. ***Shuttle thermal protection system properties adopted from [12].***

Material	**Conductivity,** $\frac{W}{m \cdot K}$	**Density,** $\frac{Kg}{m^3}$	**Specific heat,** $\frac{J}{Kg \cdot K}$
O42 coating	0.05	1665.92	500
RSI	0.043	114.17	500
RTV	0.32	1409.62	1200
Felt	0.0286	96.11	1200
Aluminum	125.0	2851.28	950

The characteristics of this analysis are similar to those of the symmetric and unsymmetric laminates previously studied, but the applied sinusoidal distribution of temperature is inverted. In this way the higher temperature is applied at the bottom to the 042-coating layer, that is the one having the stronger resistance to the severe environment produced by the friction with the atmosphere.

Results in Figures 14 and 15 confirm that for an aspect ratio of 10 the current approach provides adequate accuracy with only one element through the thickness. For an aspect ratio of 4, only a small decrease in accuracy is observed in Figures 16 and 17.

In the final test case, the top and the sides of the plate are insulated and a sinusoidal distribution of temperature is applied on the bottom surface. For this type of boundary conditions, the temperature on the top is unspecified. Because an exact analytical solution is not available for the present problem, a reference solution is obtained using the finite element program Ansys 5.3 [38]. The mesh adopted for the analysis with Ansys is made of 16 elements along x_1 and 8 along x_3. As seen in Figures 18 and 19, even one element through the thickness is able to accurately capture the temperature distribution in this case.

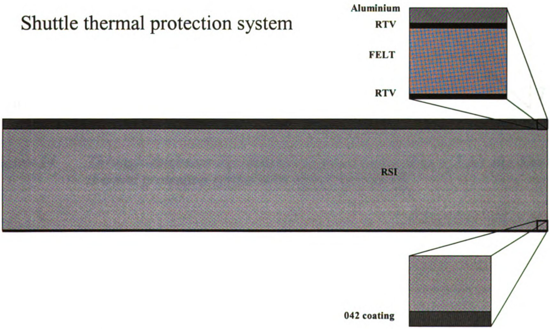


Figure 13 Shuttle thermal protection system.

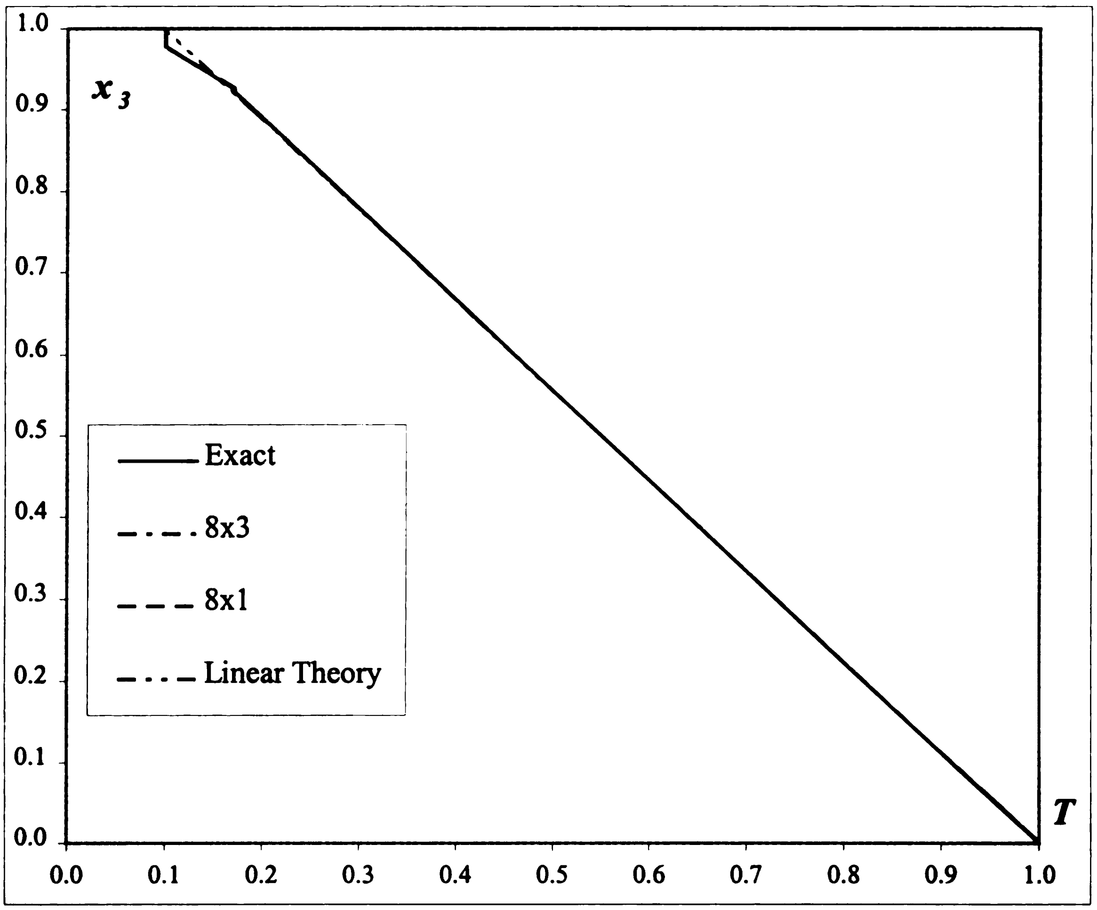


Figure 14 ***Through-thickness distribution of temperature at x_1 = 5 for the Shuttle thermal protection system with aspect ratio of 10.***

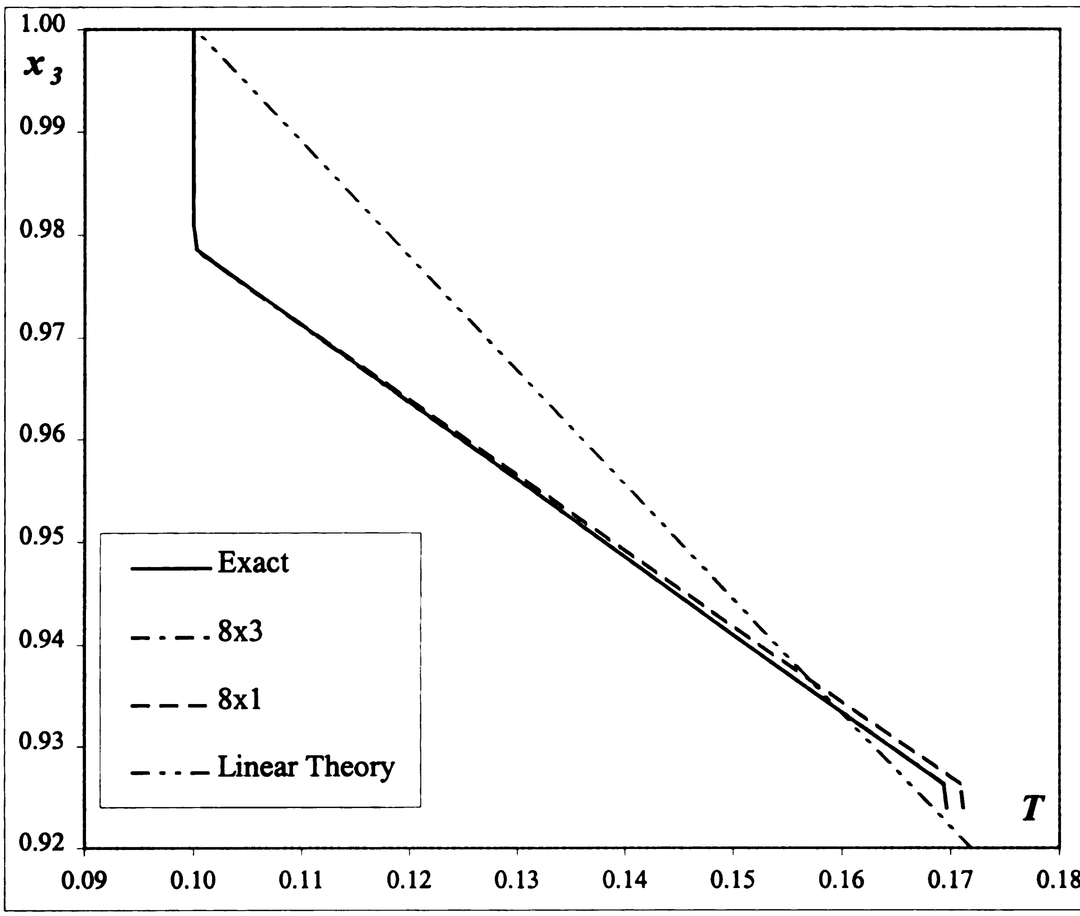


Figure 15 ***Through-thickness distribution of temperature at $x_1 = 5$ for the Shuttle thermal protection system with aspect ratio of 10. Zoom view showing the last 4 layers.***

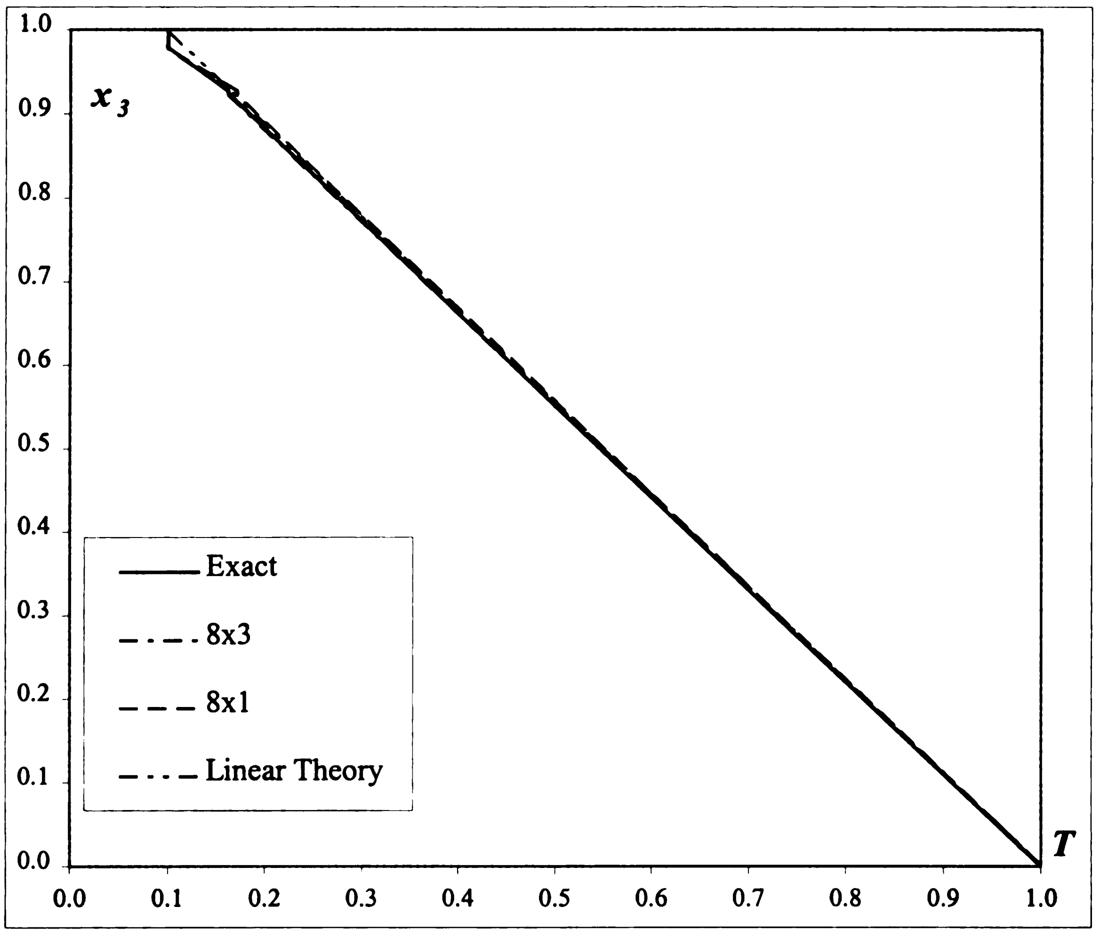


Figure 16 ***Through-thickness distribution of temperature at $x_1 = 2$ for the Shuttle thermal protection system with aspect ratio of 4.***

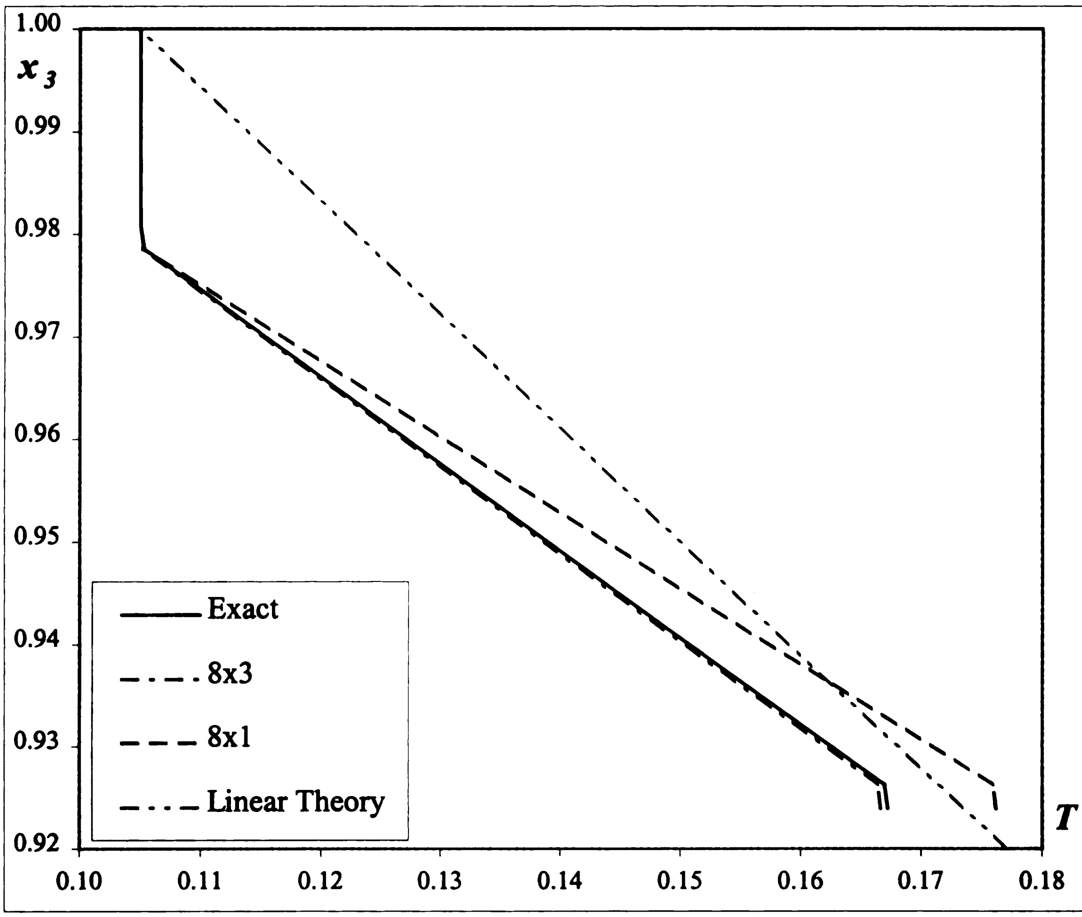


Figure 17 ***Through-thickness distribution of temperature at x_1 = 2 for the Shuttle thermal protection system with aspect ratio of 4. Zoom view showing the last 4 layers.***

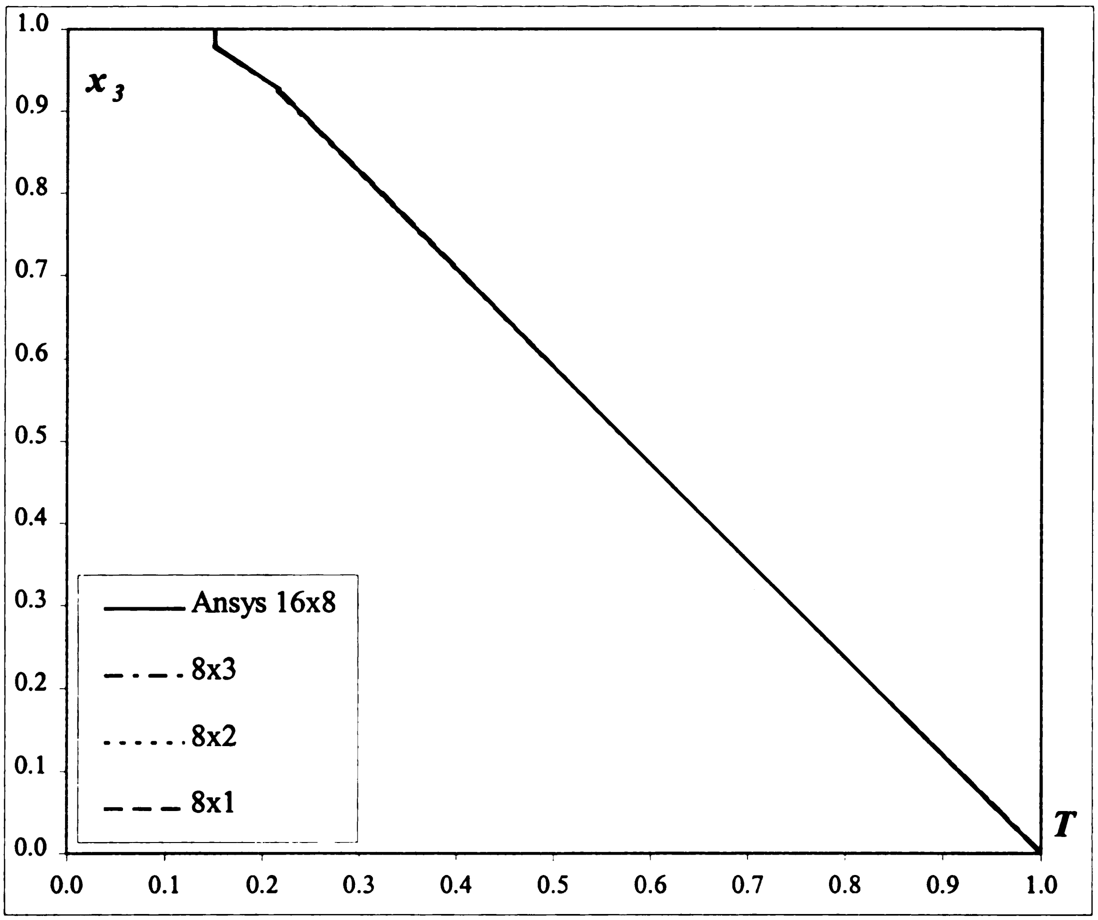


Figure 18 ***Through-thickness distribution of temperature at $x_1 = 5$ for the Shuttle thermal protection system with aspect ratio of 10. Test case with top and sides of the plate insulated.***

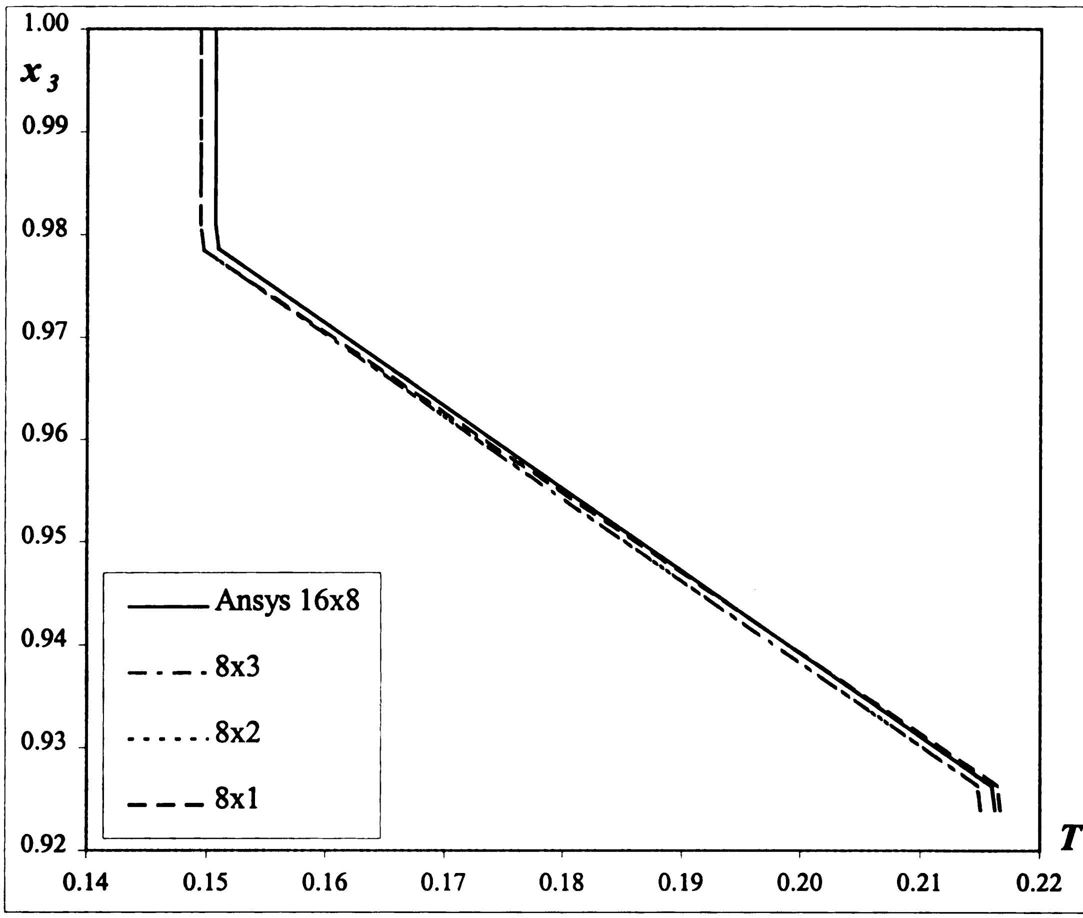


Figure 19 ***Through-thickness distribution of temperature at $x_1 = 5$ for the shuttle thermal protection system with aspect ratio of 10. Test case with top and sides of the plate insulated. Zoom view showing the last 4 layers.***

CHAPTER 3 NONLINEAR THERMAL MODEL

To test the validity of the theory in the presence of material non-linearity, the thermal conductivity coefficients of the individual layers were assumed to be a quadratic function of the temperature. This has required non-linear capabilities to be implemented in the model. Among the solution techniques more adopted for solving non-linear problems, the incremental Newton-Raphson has been chosen.

3.1 *Basic Relations*

The mathematical formulation of the presented thermal theory obliges to evaluate the temperatures in all layers composing the element at each iteration.

The stiffness matrix, Eq. (23) is a function of the thermal conductivity coefficients of the materials composing the laminate, because they appear directly in the expression. But this is not the only dependence, since the N_j^k terms are also function of Φ_k:

$$N_j^k = N_j^k\left(\Phi_k, \Psi_1, \Psi_2\right) \tag{30}$$

And Φ_k is directly related to k_3^k, as it has been previously defined in Eq. (10).

In Figure 20 the four-noded two-dimensional sublaminate finite element model and the points where we need to evaluate the temperature are shown.

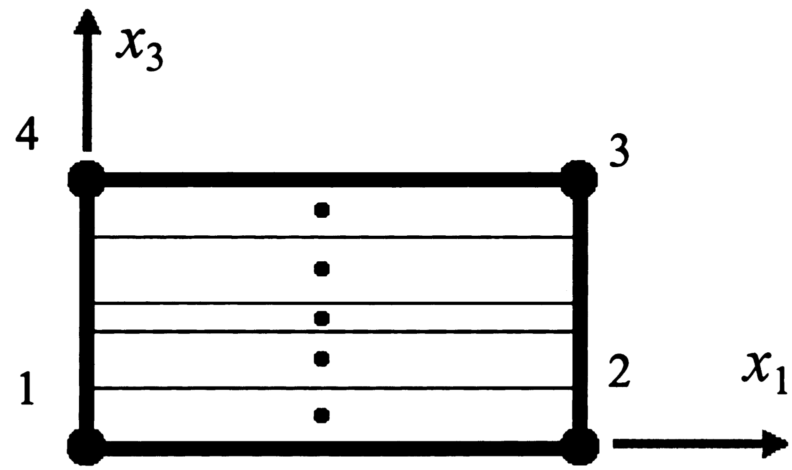


Figure 20 ***Points of the sublaminate element where the temperature needs to be evaluated.***

According to [7], a quadratic dependence of the coefficients of thermal conductivity on the temperature has been assumed. The adopted function is:

$$k = k_0 + \lambda_1 \cdot T + \lambda_2 \cdot T^2 \tag{31}$$

where the material parameters k_0, λ_1 and λ_2 define the behavior of the thermal conductivity coefficient. Since the materials composing a laminate are usually orthotropic, we will have different values of this parameters in the two directions x_1 and x_3.

3.2 The Computer Program "ThermNL"

The program has been realized in Fortran77.

Structure. The non-linear part of the structure is a typical example of a Processor subroutine for applying the Incremental Newton-Raphson method. However, since in the mixed B.C.'s we increment the flux while in presence of essential B.C.'s we scale the temperatures, two processor subroutines have been written for managing in a different way the two cases.

Input. The input file requires more information than the linear one, among these we have: number of load steps, maximum number of iterations, convergence tolerance and all the parameters required to evaluate the thermal conductivity in each material for the two directions.

Output. The output file in its first part summarize the characteristics of the analysis: element data, non-linear analysis data, mesh data, Boolean matrix, nodal boundary conditions, number and property of each layer element by element. Then, the thermal solution at each increment for a chosen node is written. After the last increment the temperature at each node is presented, and for all the layers included in an element is produced the solution at the four corners and at the center.

3.3 Numerical Results

The same symmetric 8-layer laminate studied in the first chapter has been analyzed. It has the following lamination scheme (1/2/1/2/2/1/2/1), is composed by two different materials and has an aspect ratio 1:10. Two types of analysis have been made: one in which only essential boundary conditions are specified, and another where the boundary conditions are mixed. For both problems, two possible sub-cases have been studied: in the first one the value of λ_1 and λ_2 are positive, while in the second they are negative. This has been done since the response of the model will be different for thermal conductivity coefficients that grow with the temperature as opposed to decreasing ones. In the first case, the dependence on the temperature for the two materials, as shown in

Figure 21 in a range going from 0 to 100 °C, has been assumed:

$$
\begin{aligned}
k_{x_1}^{1} &= 10 + 0.001 \cdot T + 0.01 \cdot T^2 \\
k_{x_3}^{1} &= 5 + 0.001 \cdot T + 0.01 \cdot T^2
\end{aligned} \tag{32}
$$

$$
\begin{aligned}
k_{x_1}^{2} &= 1 + 0.0001 \cdot T + 0.001 \cdot T^2 \\
k_{x_3}^{2} &= 0.5 + 0.0001 \cdot T + 0.001 \cdot T^2
\end{aligned} \tag{33}
$$

where the upper number marks the material and the conductivity coefficients are given in units of $\dfrac{W}{m \cdot {}^\circ C}$.

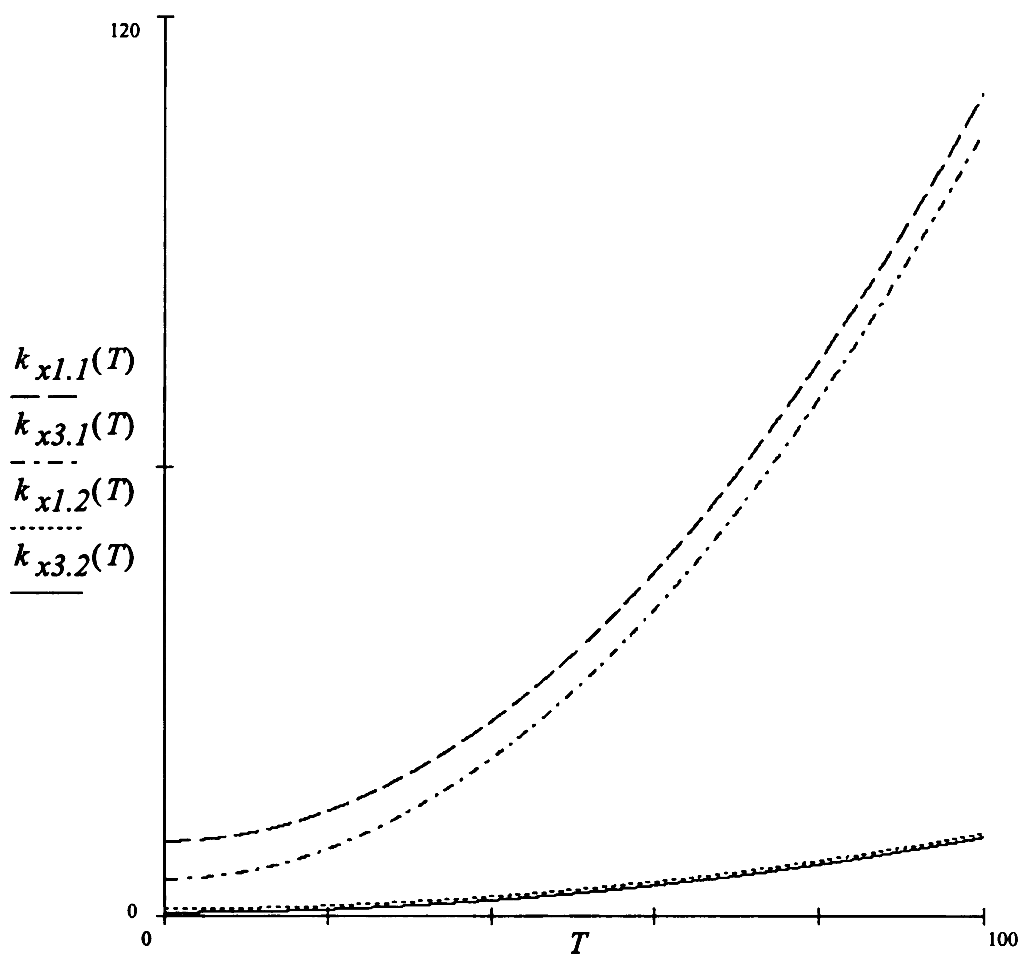


Figure 21 ***Thermal conductivities variation with T (°C) for λ_1 and λ_2 positive***

For decreasing thermal conductivity coefficients the dependence on the temperature for the two materials has been reduced.

$$
\begin{aligned}
k_{x_1}^{1} &= 10 - 0.001 \cdot T - 0.0004 \cdot T^2 \\
k_{x_3}^{1} &= 5 - 0.001 \cdot T - 0.0004 \cdot T^2
\end{aligned}
\tag{34}
$$

$$
\begin{aligned}
k_{x_1}^{2} &= 1 - 0.0001 \cdot T - 0.00004 \cdot T^2 \\
k_{x_3}^{2} &= 0.5 - 0.0001 \cdot T - 0.00004 \cdot T^2
\end{aligned}
\tag{35}
$$

The assumed parameters let us reach very small values of thermal conductivity as the temperature approaches 100 °C, avoiding negative values that are not admissible (see Figure 22).

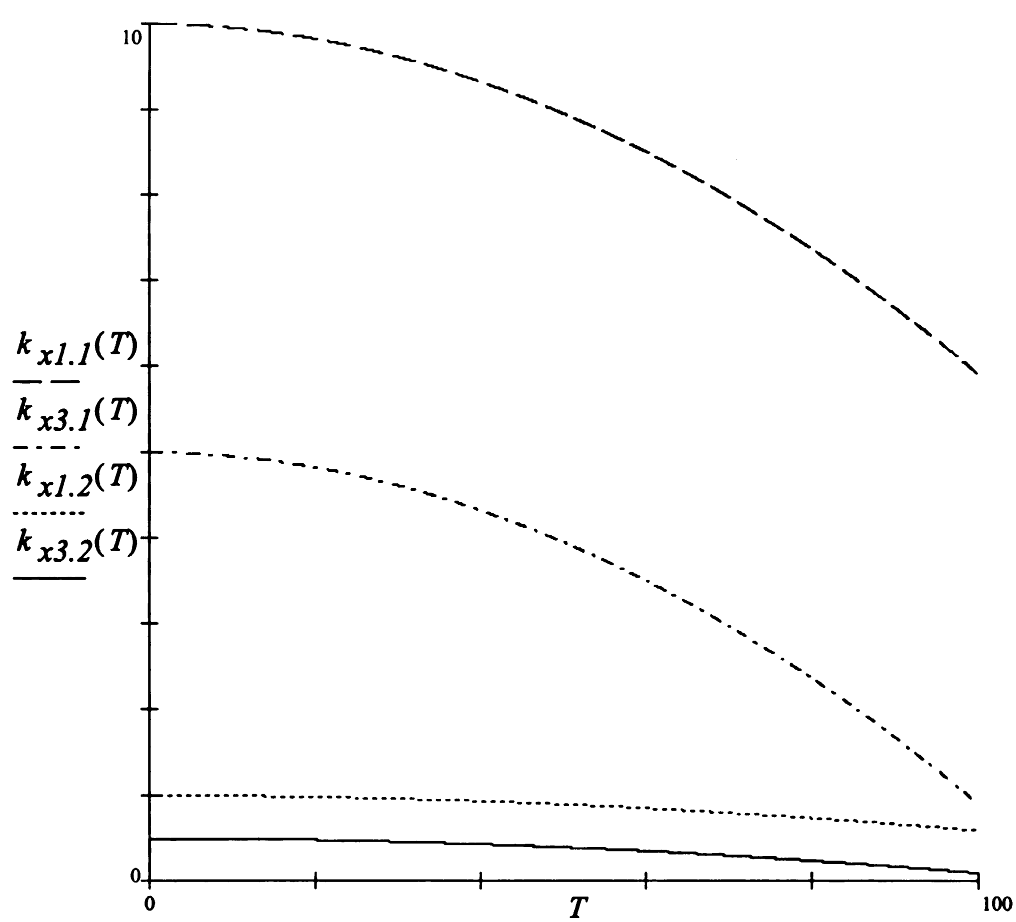


Figure 22 ***Thermal conductivities variation with T (°C) for λ_1 and λ_2 negative.***

The previous choice of the parameters comes out from several analyses that have been run in order to find values producing a high level of non-linearity. Being careful at the same time to avoid problems that can converge only by the use of special techniques like the arc-length method. These tests, however, worked on the values of the parameters λ_1 and λ_2, because the constant term k_0 has not been changed for comparison reasons with the linear results.

Because an exact analytical solution is not available for the present problems, a reference solution is obtained using the finite element program Ansys 5.3 [38] with an 8x8 mesh.

3.3.1 Mixed Boundary Conditions Analysis

As it is shown in the Figure 23, a uniform heat flux has been specified on the top surface of a laminated plate while a uniform temperature is assigned to the bottom surface. Lateral surfaces are insulated.

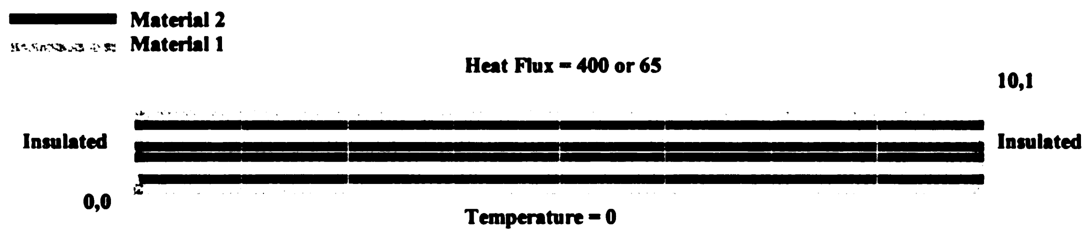


Figure 23 Model for mixed B.C.'s analysis

3.3.1.1 Growing Thermal Conductivity coefficients

<u>*Validation of final results.*</u> We will compare the final solutions presented from Ansys5.3, with those produced by our finite element model. The specified heat flux is $400\ \frac{W}{m^2}$.

By the use of Figures 24 and 25, the temperature is reported along the thickness at $x_1 = 2.5$ and $x_1 = 5$. Four different meshes are used in ThermNL: 8x4, 8x2 and 8x1, where the two numbers indicate respectively the subdivisions in x_1 and x_3 directions. A close agreement between the solutions from Ansys5.3 and the program is obtained with all the meshes; only a small error can be notice at $x_1 = 2.5$ for the 8x1 mesh.

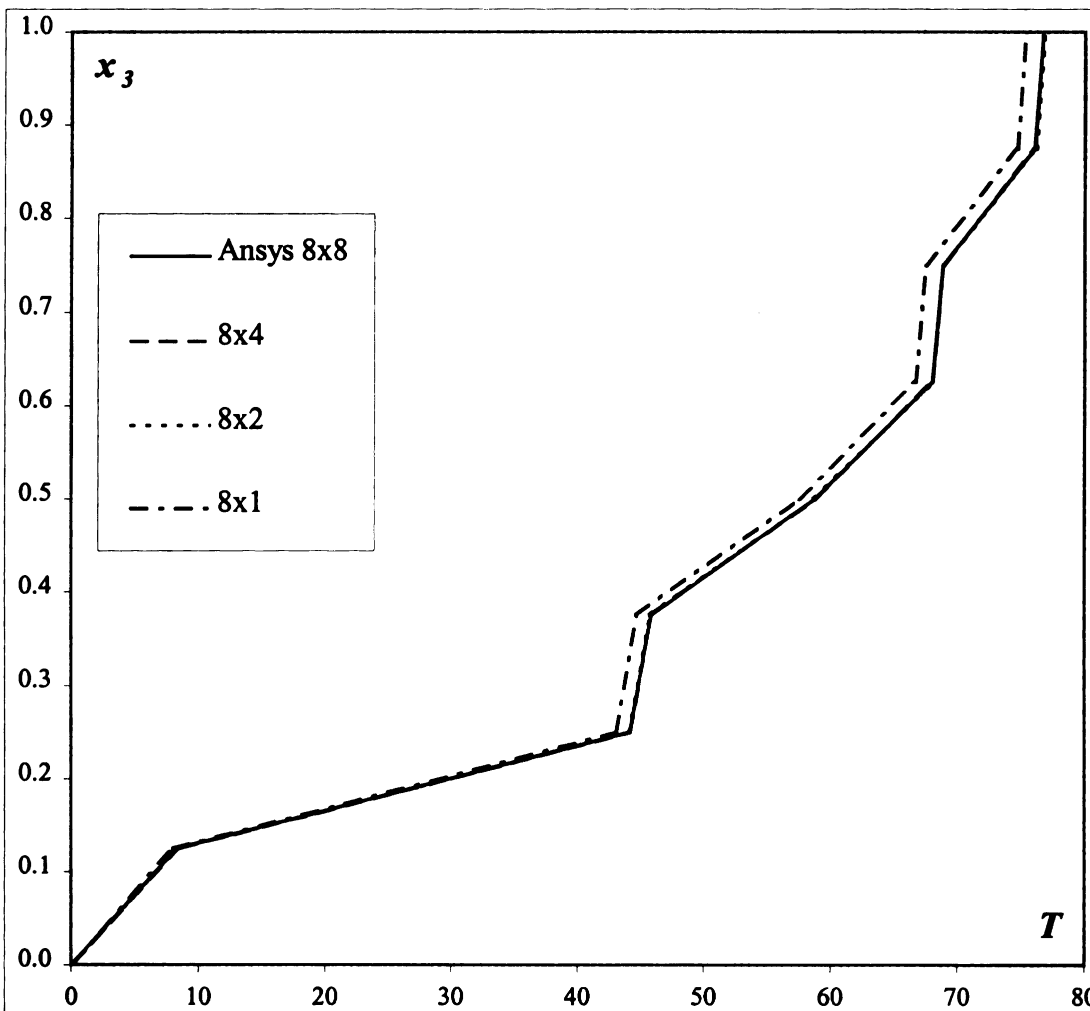


Figure 24 ***Final through-thickness distribution of temperature at x_1 = 2.5 for values of the conductivity coefficients k increasing with T.***

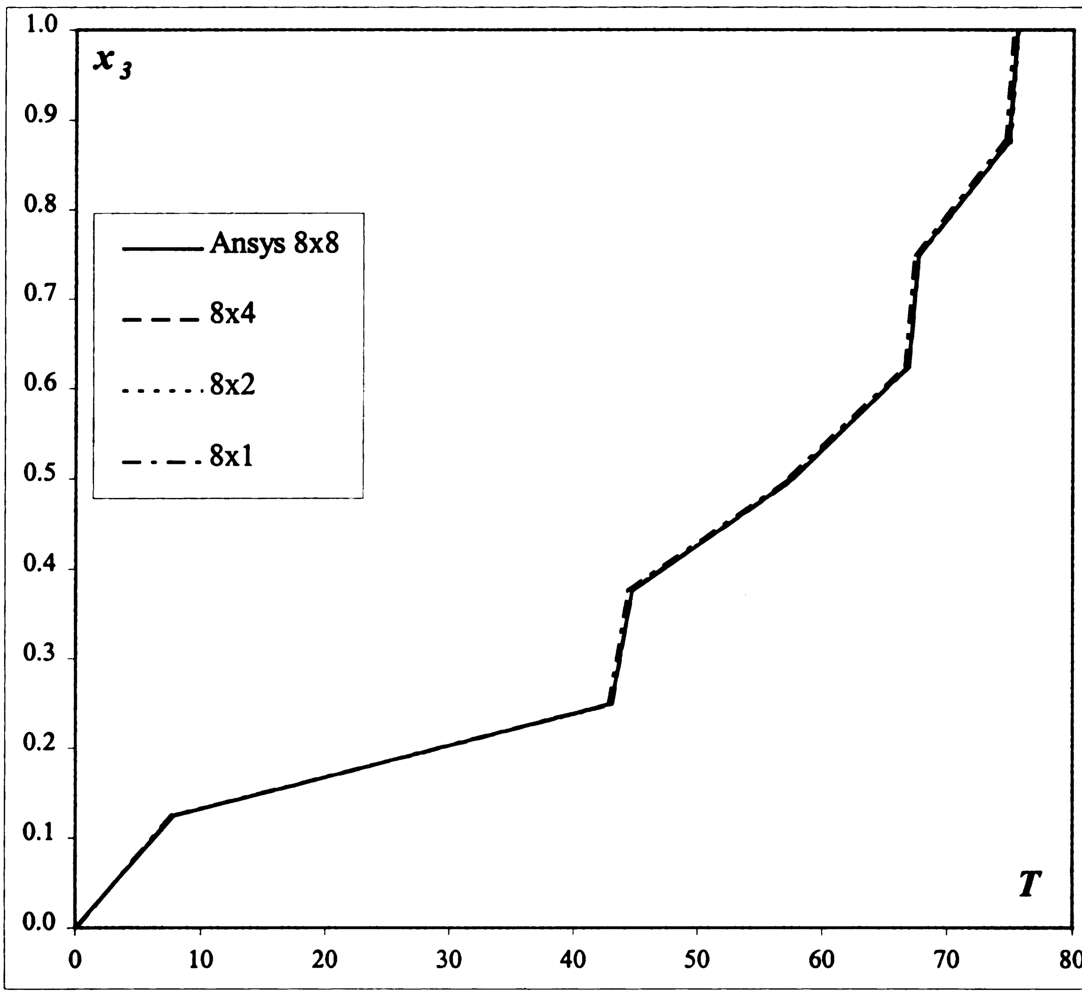


Figure 25 ***Final through-thickness distribution of temperature at $x_1 = 5$ for values of the conductivity coefficients k increasing with T.***

Validation of the solution through the increments. In this validation test, the solutions given by Ansys and ThermNL have been recorded at the end of each increment, or load step. In particular, in Figure 26 are reported the solutions at the central node of the top surface obtained using 100 increments. The linear solution allows us to see how soon the non-linear ones diverge from the temperatures that we obtain if the thermal conductivity coefficients are constant. Since the results from the 8x2 and 8x4 meshes perfectly agree with the Ansys, only those from the 8x2 and 8x1 meshes are reported in order to get a clearer graph.

We could have predicted the observed behavior, knowing that for a specified heat flux, the temperature in all points of the mesh decrease as soon as the values of the thermal conductivity coefficients increase. In our analysis, as the flux keeps growing with the increments the values of the temperature in the mesh increase. As a consequence, the thermal conductivity coefficients grow making the temperature at the end of the following increments smaller than those in the linear solution.

The adherence among the solutions is excellent for all the meshes. The same test has been done at the central point of the mesh. The quality of the solution is unchanged, as Figure 27 demonstrates.

3.3.1.2 Decreasing Thermal Conductivity coefficients

This validation analysis is similar to the previous one, with the only change that now we will apply the negative values of the thermal conductivity coefficients, defined in (34) and (35). In order to maintain the solution in a comparable range of temperature like the other analysis, the specified heat flux is reduced from 400 to 65.

In Figure 28 is reported the solutions at the central node of the top surface. In this case the linear solution goes under the others, because the thermal conductivity coefficients decrease while the temperatures in the mesh increase.

Again, the adherence among the solutions is excellent for all the meshes.

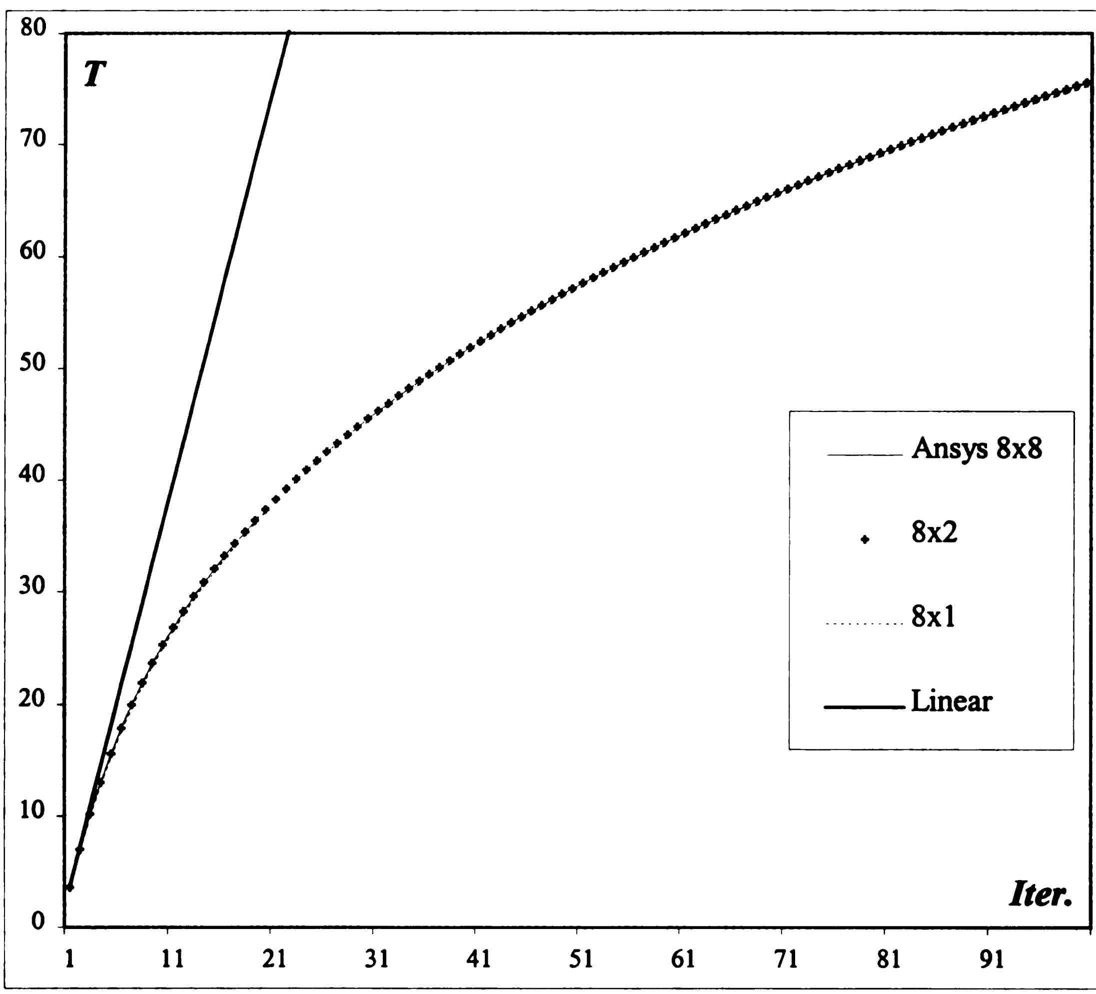


Figure 26 ***Solution at every increments at $x_1 = 5$ and $x_3 = 1$ for values of the conductivity coefficient k increasing with T.***

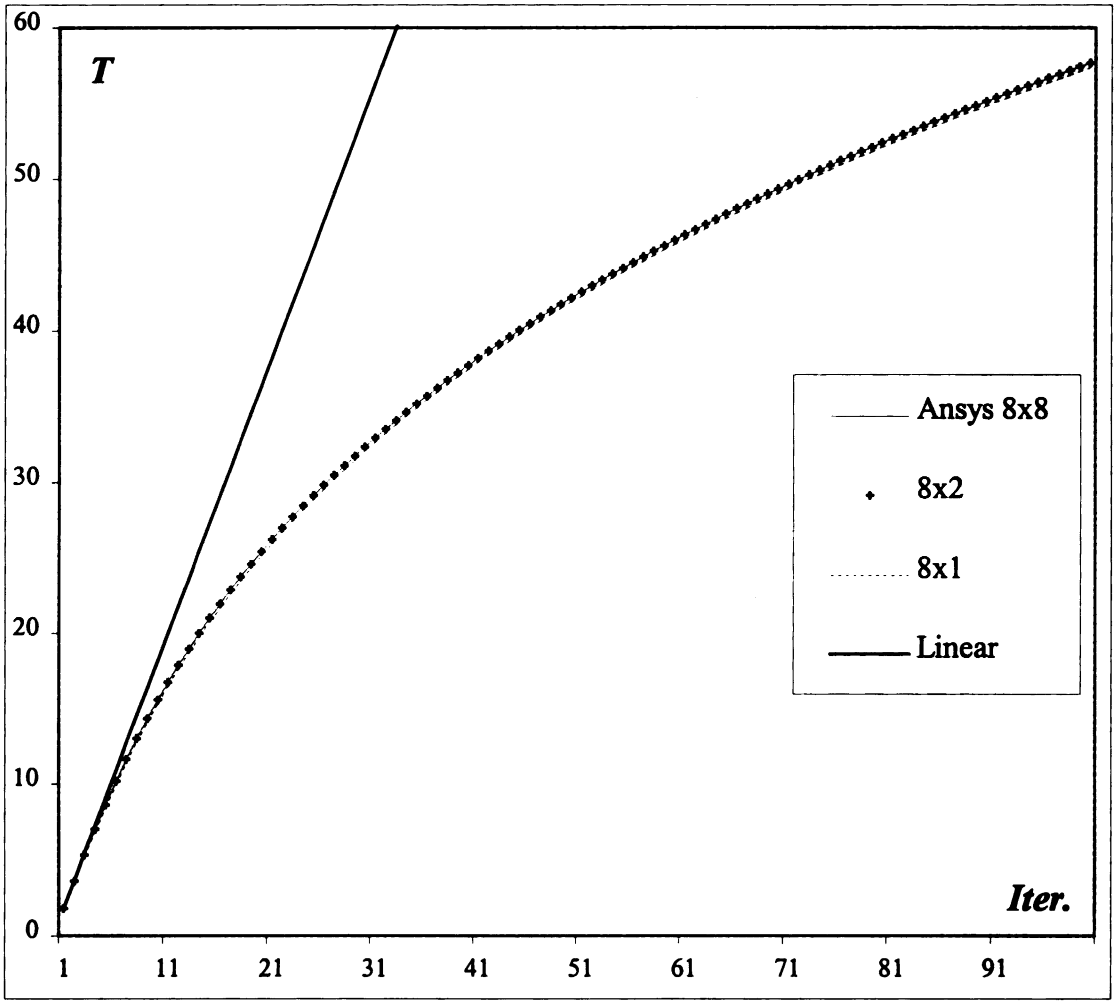


Figure 27 ***Solution at every increments at $x_1 = 5$ and $x_3 = 0.5$ for values of the conductivity coefficient k increasing with T.***

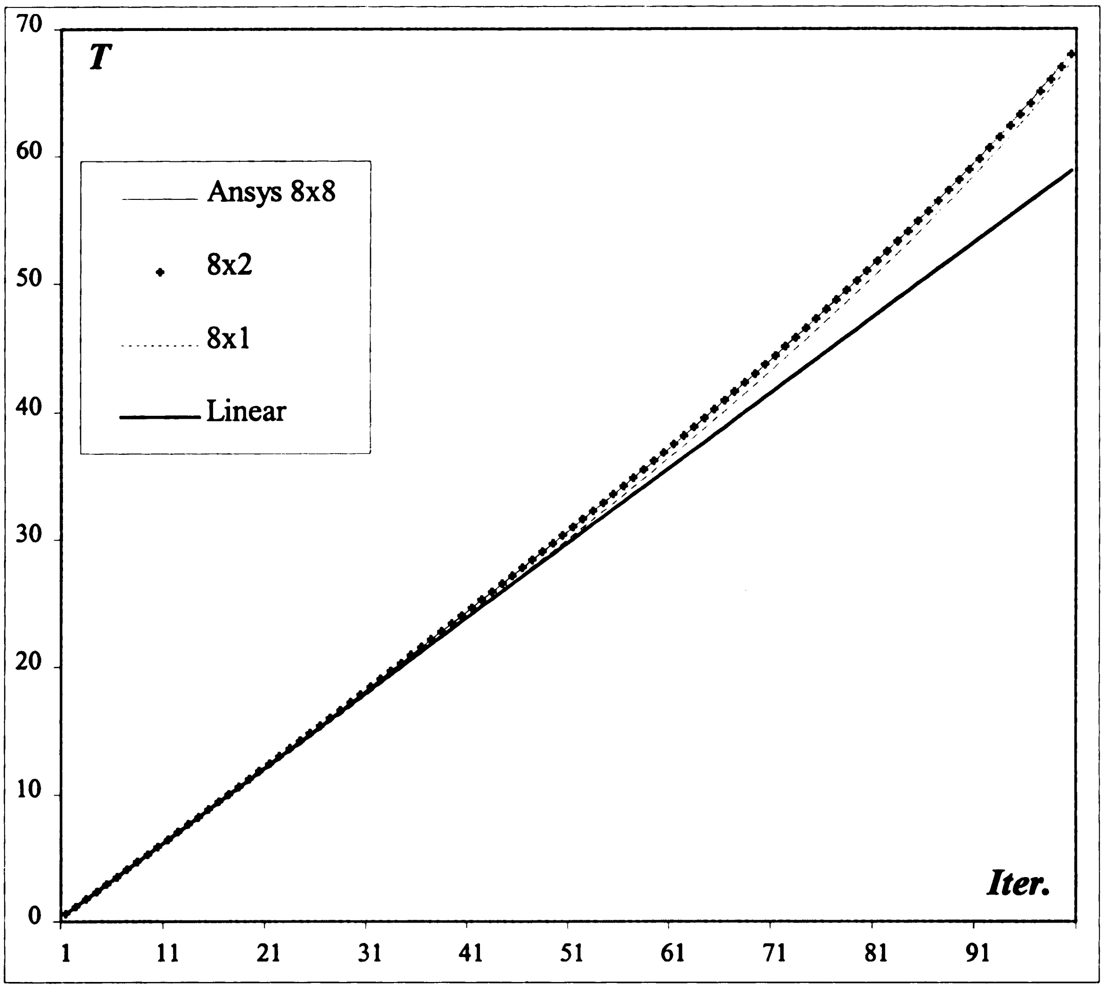


Figure 28 ***Solution at every increments at $x_1 = 5$ and $x_3 = 1$ for values of the conductivity coefficient k decreasing with T.***

3.3.2 Essential Boundary Conditions Analysis

In this problem the same laminated plate has been subjected at the top and bottom to a sinusoidal distribution of temperature of the type used in the first chapter:

$$\begin{aligned} T_{top}(x_1) &= T_t \cdot \sin\left(\frac{\pi \cdot x_1}{L}\right) \\ T_{bottom}(x_1) &= T_b \cdot \sin\left(\frac{\pi \cdot x_1}{L}\right) \end{aligned} \tag{36}$$

On the two lateral sides, at $x_1 = 0$ and $x_1 = L$, a uniform temperature $T = 0$ is imposed. The value of the constants T_t and T_b have been taken equal to 100 and 10 respectively. In this analysis at each load step, the program will increment the values of the temperature. They will start from zero and reach their specified value at the last increment. The model with the applied boundary conditions is shown in the Figure 29.

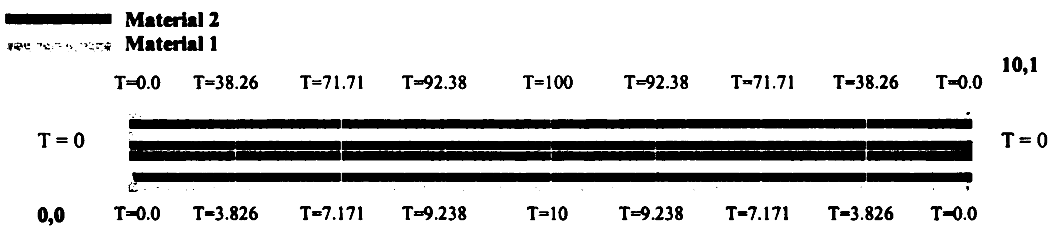


Figure 29 ***Model for essential B.C.'s analysis.***

3.3.2.1 Growing Thermal Conductivity coefficients

As in the previous two tests, we compare the solution at the end of each increment. Since the temperatures are specified on all the surfaces, here we can check the solution only in the internal points. The central node of the mesh has been chosen; its coordinates are $x_1 = 5$ and $x_3 = 0.5$. In Figure 30 are reported the solutions at this node for an analysis of 100 increments. The adherence among the solutions is excellent for the

8:

th

te

3

ar

T

3.

nc

sc

ex

4.

wi

alc

rep

8x8, 8x4 and 8x2 meshes. A small error is registered for the 8x1 mesh. The results from the 8x8 and 8x4 meshes are not reported in order to get a clearer graph. The final temperatures obtained along the thickness at $x_1 = 5$ are shown in Table 6 and in Figure 31. A close agreement between the two solutions is obtained even with the 8 by 1 mesh, and convergence toward the solution from Ansys is obtained as the mesh is refined.

Table 6. ***Comparison among the temperature distributions along the thickness at x_1= 5 at the end of the nonlinear analysis.***

x_3	Ansys 8x8	8x8	8x4	8x2	8x1
0.000	0.000000	0.000000	0.000000	0.000000	0.000000
0.125	0.203390	0.203410	0.201832	0.203779	0.209496
0.250	0.580270	0.580232	0.579958	0.584959	0.601314
0.375	0.600150	0.600109	0.600415	0.604547	0.621314
0.500	0.759770	0.759751	0.759634	0.758592	0.778503
0.625	0.874860	0.874850	0.875548	0.883694	0.892125
0.750	0.885700	0.885698	0.885641	0.894493	0.902067
0.875	0.989720	0.989717	0.990600	0.991265	0.991842
1.000	1.000000	1.000000	1.000000	1.000000	1.000000

3.3.2.2 Decreasing Thermal Conductivity coefficients

This validation analysis resembles the previous one, except that the values of the nonlinear thermal conductivity coefficients are negative. In Figure 32 is reported the solutions in the central node of the mesh. Again the adherence among the solutions is excellent for the 8x4 (not shown in the graph) and 8x2 meshes, but an error of about 4.7% is registered for the 8x1 mesh. There is a reason why the solutions are less accurate with respect to those in which the flux is specified. Here the gradient of the temperature along the x_1 direction is greater and the thermal theory has more difficulty in representing it with few elements.

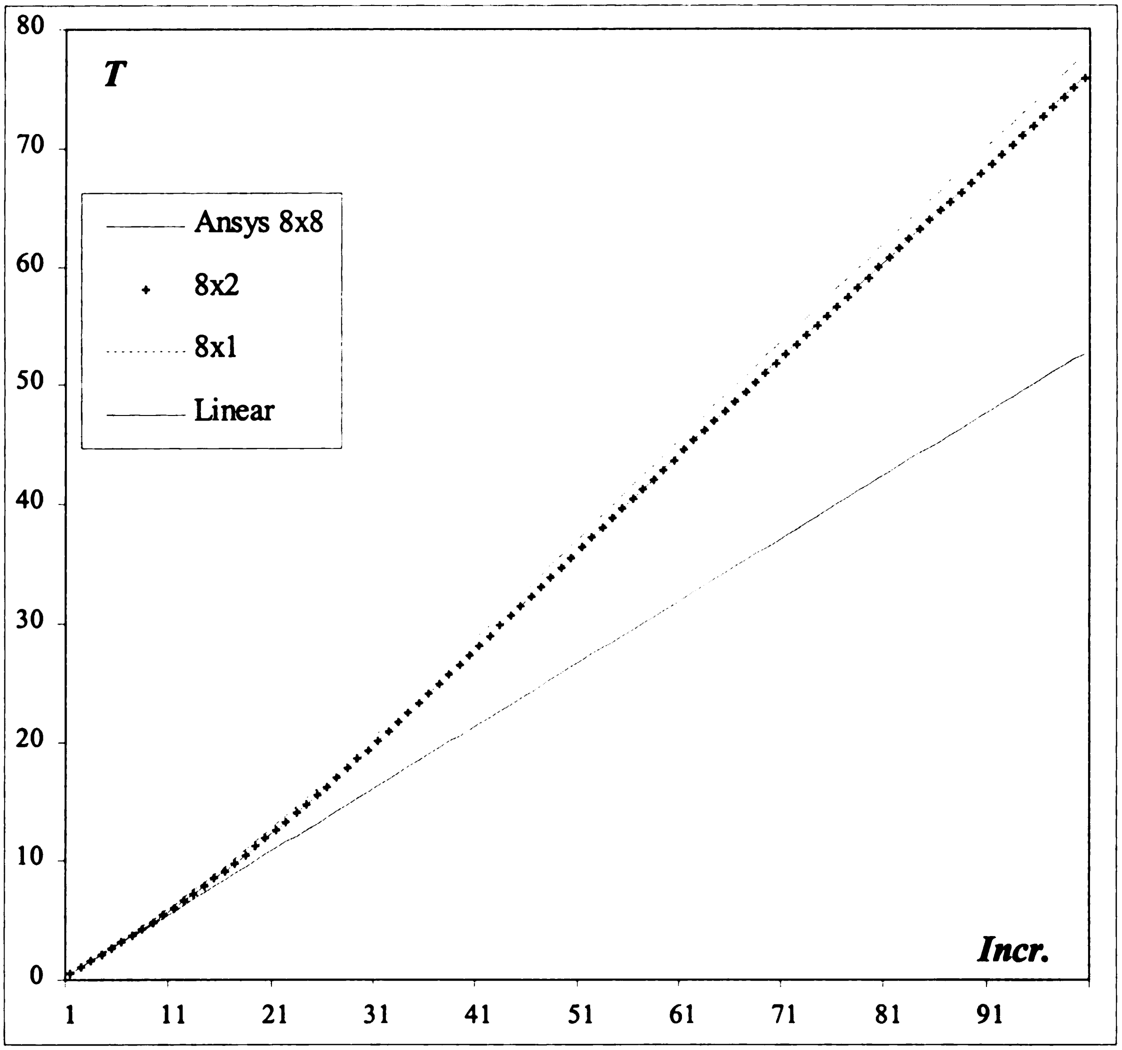


Figure 30 ***Solution at every increments at $x_1 = 5$ and $x_3 = 0.5$ for values of the conductivity coefficient k increasing with T.***

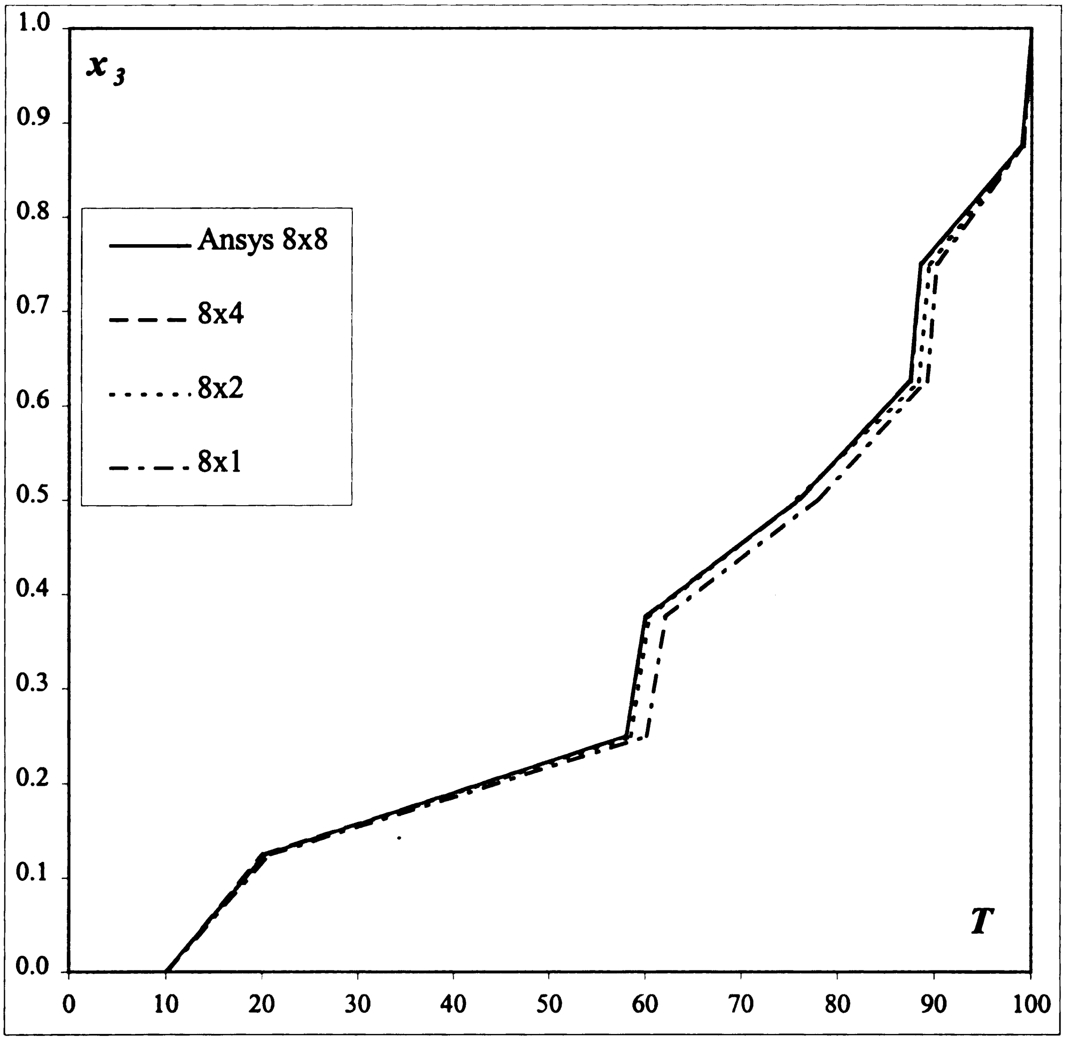


Figure 31 ***Final through-thickness distribution of temperature at x_1 = 5 for values of the conductivity coefficient k increasing with T.***

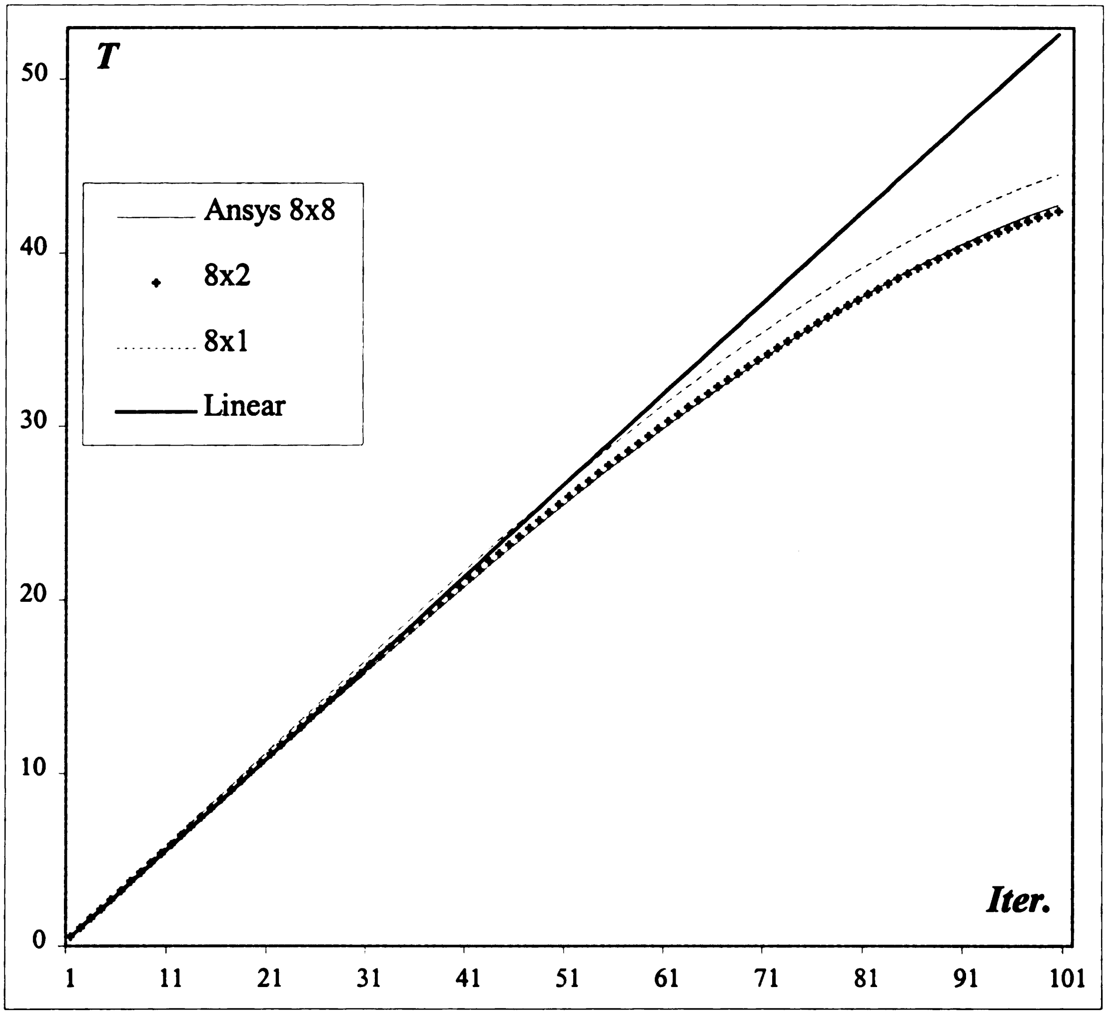


Figure 32 ***Solution at every increments at $x_1 = 5$ and $x_3 = 0.5$ for values of the conductivity coefficient k decreasing with T.***

CHAPTER 4 FIRST ORDER ZIG-ZAG PLATE THEORY

4.1 Formulation of the Theory

In this section a brief description of the first order zig-zag plate theory will be provided, neglecting details related to the mathematical development and focusing the attention on the implementation of the thermal stress capabilities. For a complete explanation of the theory and the numerical validation studies, the reader is referred to [31].

As in the thermal theory the laminate will be modeled as M sublaminates, with each of them consisting of N_m perfectly bonded layers. Again the global X_3 axis is taken perpendicular to the inplane X_1, X_2 coordinate axes and has its origin at the bottom of the laminate, while x_3 is a local sublaminate coordinate in the direction of X_3 with its origin at the bottom of the sublaminate (see Fig. 1). In the following derivation all expressions are related to the m-th sublaminate, but all indices indicating the sublaminate number are omitted for brevity.

The constitutive relations for a three dimensional stress state in the k-th layer of the m-th sublaminate in the presence of a thermal load can be written as:

$$\begin{Bmatrix} \sigma_{11}^{(k)} \\ \sigma_{22}^{(k)} \\ \sigma_{33}^{(k)} \\ \sigma_{23}^{(k)} \\ \sigma_{13}^{(k)} \\ \sigma_{12}^{(k)} \end{Bmatrix} = \begin{bmatrix} C_{11}^{(k)} & C_{12}^{(k)} & C_{13}^{(k)} & 0 & 0 & C_{16}^{(k)} \\ C_{12}^{(k)} & C_{22}^{(k)} & C_{23}^{(k)} & 0 & 0 & C_{26}^{(k)} \\ C_{13}^{(k)} & C_{23}^{(k)} & C_{33}^{(k)} & 0 & 0 & C_{36}^{(k)} \\ 0 & 0 & 0 & C_{44}^{(k)} & C_{45}^{(k)} & 0 \\ 0 & 0 & 0 & C_{45}^{(k)} & C_{55}^{(k)} & 0 \\ C_{16}^{(k)} & C_{26}^{(k)} & C_{36}^{(k)} & 0 & 0 & C_{66}^{(k)} \end{bmatrix} \begin{Bmatrix} \varepsilon_{11}^{(k)} - \alpha_1^{(k)} \cdot \Delta T^{(k)} \\ \varepsilon_{22}^{(k)} - \alpha_2^{(k)} \cdot \Delta T^{(k)} \\ \varepsilon_{33}^{(k)} - \alpha_3^{(k)} \cdot \Delta T^{(k)} \\ \gamma_{23}^{(k)} \\ \gamma_{13}^{(k)} \\ \gamma_{12}^{(k)} - \alpha_{12}^{(k)} \cdot \Delta T^{(k)} \end{Bmatrix} \tag{37}$$

where: $$\Delta T^{(k)} = T_{current}^{(k)} - T_{initial}^{(k)}$$

For most practical laminated composites, the 13 coefficients of the material stiffness matrix are obtained from 9 independent material constants and a transformation law (see [39]).

In the case of the thermal expansion coefficients the change between the material and the laminate coordinate system follows the usual transformation rule:

$$\begin{aligned} \alpha_1 &= \alpha_L \cos^2\theta + \alpha_T \sin^2\theta \\ \alpha_2 &= \alpha_L \sin^2\theta + \alpha_T \cos^2\theta \\ \alpha_{12} &= 2(\alpha_L - \alpha_T)\sin\theta\cos\theta \\ \alpha_3 &= \alpha_T \end{aligned} \tag{38}$$

where α_L and α_T are, respectively, the thermal expansion coefficient in the directions parallel and perpendicular to the fibers, and θ is the angle between fiber direction and laminate x_1 axis.

The sublaminate displacement fields are initially assumed in the following form:

$$\begin{aligned} u_1^{(k)}(x_1,x_2,x_3,t) &= u_b + x_3 \cdot \Psi_1 + \sum_{i=1}^{k-1}(x_3 - x_{3i}) \cdot \xi_i \\ u_2^{(k)}(x_1,x_2,x_3,t) &= v_b + x_3 \cdot \Psi_2 + \sum_{i=1}^{k-1}(x_3 - x_{3i}) \cdot \eta_i \\ u_3^{(k)}(x_1,x_2,x_3,t) &= w_b \cdot \left(1 - \frac{x_3}{h}\right) + w_t \cdot \left(\frac{x_3}{h}\right) \end{aligned} \tag{39}$$

where h is the thickness of the sublaminate, u_b and v_b are the axial displacements in x_1 and x_2 directions, respectively, at $x_3 = 0$, Ψ_1 and Ψ_2 are rotations of the normal at $x_3 = 0$, and w_b and w_t are the transverse deflections of the bottom and top surfaces, respectively, of the m-th sublaminate. Thus, it is assumed that $u_1^{(k)}$ and $u_2^{(k)}$ vary in a piecewise linear fashion through the thickness of a sublaminate and $u_3^{(k)}$ varies linearly through the thickness. The parameters ξ_i and η_i in Eq. (39) are eliminated by enforcing the continuity of transverse shear stress at each interface [21,22]. The shear stress continuity conditions at the k-th interface can be expressed as:

$$\sigma_{23}^{(k)} = \sigma_{23}^{(k+1)}, \qquad \sigma_{13}^{(k)} = \sigma_{13}^{(k+1)} \tag{40}$$

Assuming that infinitesimal strain theory holds, the form of the assumed displacement fields in Eq. (39) is such that the inplane displacements $u_1^{(k)}$ and $u_2^{(k)}$ contribute only constant terms to the transverse shear stresses, $\sigma_{13}^{(k)}$ and $\sigma_{23}^{(k)}$. For consistency, then, the transverse displacement $u_3^{(k)}$ should also contribute only constant terms to these stress components. In order to achieve this consistency, terms in the transverse shear strain expressions that involve the transverse displacement variables are evaluated at the midplane of the sublaminate, thereby taking a thickness averaged value of these terms. Substituting Eqs. (37, 39) into Eq. (40) and solving for ξ_i and η_i, it can be seen that ξ_i and η_i depend on the ratios of shear properties between adjacent layers and the shear deformation in each sublaminate. If either of these quantities is small, then discrete-layer effects will also be small.

The rotational variables Ψ_1 and Ψ_2 in the displacement fields are now eliminated by introducing the variables u_t and v_t, the inplane translations at the top surface of the sublaminate [25,26], in order to expedite the development of versatile finite element models. Thus, rather than describing the inplane displacement field by a translation and rotation at one point, it can more conveniently be described here by the translation at two points.

The displacement fields must be further manipulated in order to develop a C^0 finite element. Because the derivatives of transverse deflections, $\frac{\partial w_b}{\partial x_1}, \frac{\partial w_t}{\partial x_1}, \frac{\partial w_b}{\partial x_2}$ and $\frac{\partial w_t}{\partial x_2}$ appear in the displacement field, their second derivatives will be present in the strain energy functional. C^1 continuity of w_b and w_t, is thus required. It is desirable to alleviate such a requirement by introducing new rotational degrees of freedom as follows:

$$\frac{\partial w_b}{\partial x_1} = -\theta_{2b}, \ \frac{\partial w_t}{\partial x_1} = -\theta_{2t}, \ \frac{\partial w_b}{\partial x_2} = \theta_{1b}, \ \frac{\partial w_b}{\partial x_2} = \theta_{1t} \tag{41}$$

These relationships will be constrained in the strain energy via a combination of the interdependent interpolation and the penalty method. The displacement fields can now be written using indicial notation as follows:

$$u_1^{(k)} = \bar{u}_{\alpha\beta}\Phi_{\alpha\beta}^{(k)}, \ u_2^{(k)} = \bar{u}_{\alpha\beta}\Psi_{\alpha\beta}^{(k)}, \ u_3^{(k)} = w_\beta \Omega_\beta \qquad \alpha = 1\ldots4; \ \beta = 1,2 \qquad 42)$$

where the index β is used to denote the top and bottom of the sublaminate with 1: bottom and 2: top, and $\Phi_{\alpha\beta}^{(k)}, \Psi_{\alpha\beta}^{(k)}$ and Ω_β are shape functions in the thickness direction [31]. Unless noted otherwise, summation on repeated indices is implied. The variables $\bar{u}_{\alpha\beta}$ are:

$$\bar{u}_{1\beta} = u_\beta, \ \bar{u}_{2\beta} = v_\beta, \ \bar{u}_{3\beta} = \theta_{1\beta}, \ \bar{u}_{4\beta} = \theta_{2\beta} \tag{43}$$

The transverse normal strain arising from the assumed displacement field (42) is constant through the thickness of a sublaminate. This assumption may generate substantial errors in composites that have a soft core or layer, especially when thermal loads are applied. Consider a laminate composed of 3 layers, one of which is very compliant. When it is under loading the uniform transverse normal strain distribution through the thickness is predicted to be uniform, but the actual distribution is not. Thus, it is desirable to improve the distribution of the transverse normal strain through the thickness. The improvement can be achieved by instead assuming a constant transverse normal stress, $\bar{\sigma}_{33}$ through the thickness of a sublaminate. $\bar{\sigma}_{33}$ can be determined using Reissner's mixed variational principle [40]. From Eq. (37), the transverse normal strain is obtained as:

$$\begin{aligned}\bar{\varepsilon}_{33}^{(k)} &= \frac{1}{C_{33}^{(k)}}\left(\bar{\sigma}_{33}^{(k)} - C_{13}^{(k)}\varepsilon_{11}^{(k)} - C_{23}^{(k)}\varepsilon_{22}^{(k)} - C_{36}^{(k)}\gamma_{12}^{(k)}\right) + \\ &\frac{1}{C_{33}^{(k)}}\left(C_{13}^{(k)}\alpha_1 + C_{23}^{(k)}\alpha_2 + C_{36}^{(k)}\alpha_{12} + C_{33}^{(k)}\alpha_3\right)\Delta T^{(k)}\end{aligned} \tag{44}$$

Where, the part of the strain due to the thermal load is denoted by:

$$\bar{\varepsilon}_{33}^{TL(k)} = \frac{1}{C_{33}^{(k)}}\left(C_{13}^{(k)}\alpha_1 + C_{23}^{(k)}\alpha_2 + C_{36}^{(k)}\alpha_{12} + C_{33}^{(k)}\alpha_3\right)\Delta T^{(k)} \tag{45}$$

$\bar{\sigma}_{33}$ is then determined analytically from the following relation:

$$0 = \sum_{k=1}^{N}\int_{x_{3(k-1)}}^{x_{3(k)}}\left(\bar{\varepsilon}_{33}^{(k)} - \frac{w_t - w_b}{h}\right)dx_3 \tag{46}$$

The constant transverse normal stress $\bar{\sigma}_{33}$ is found to be:

$$\bar{\sigma}_{33} = f_1(\bar{u}_{\alpha\beta}, w_\beta, \Phi_{\alpha\beta}^{(k)}, \Psi_{\alpha\beta}^{(k)}, \Omega_\beta) - C_{33}^{(k)}\cdot\bar{\varepsilon}_{33}^{TL(k)} \tag{47}$$

The newly defined transverse normal strain can be written as:

$$\bar{\varepsilon}_{33}^{(k)} = \bar{\varepsilon}_{33}^{M(k)} - \bar{\varepsilon}_{33}^{TL(k)} \tag{48}$$

where:

$$\bar{\varepsilon}_{33}^{M(k)} = f_2(\bar{u}_{\alpha\beta}, w_\beta, \Phi_{\alpha\beta}^{(k)}, \Psi_{\alpha\beta}^{(k)}, \Omega_\beta) \tag{49}$$

The functions f_1 and f_2 are defined in [31].

The equations of motion, the essential and natural boundary conditions, as well as the displacement-based finite element model can be developed from Hamilton's principle. Again, attention will be limited to a single sublaminate. The energy functional is modified here to include the imposition of the constraints in Eq. (41) via the penalty method. Hamilton's principle for the *m*-th sublaminate can be defined as follows:

$$\delta\int_{t_0}^{t_1}\left[U - K - W + P\right]dt = 0 \tag{50}$$

U is the internal strain energy:

$$U = \sum_{k=1}^{N_m}\int_{V_k}\frac{1}{2}\left[\sigma_{11}^{(k)}\varepsilon_{11}^{(k)} + \sigma_{22}^{(k)}\varepsilon_{22}^{(k)} + \sigma_{33}^{(k)}\varepsilon_{33}^{(k)} + \sigma_{12}^{(k)}\gamma_{12}^{(k)} + \sigma_{13}^{(k)}\gamma_{13}^{(k)} + \sigma_{23}^{(k)}\gamma_{23}^{(k)}\right]dV_k \tag{51}$$

where V_k is the volume of the *k*-th layer.

K is the kinetic energy:

$$K = \sum_{k=1}^{N_m}\int_{V_k}\frac{1}{2}\rho^{(k)}\left[\left(\dot{u}_1^{(k)}\right)^2 + \left(\dot{u}_2^{(k)}\right)^2 + \left(\dot{u}_3^{(k)}\right)^2\right]dV_k \tag{52}$$

where $\rho^{(k)}$ is the density of the *k*-th layer and a superposed dot indicates differentiation with respect to time. W is the work of external forces:

$$W = \int_\Omega w_\alpha P_\alpha \, d\Omega + \int_\Gamma t_i u_i \, d\Gamma \tag{53}$$

P_α = concentrated forces; $t_i = \sigma_{ij}\eta_j$; $\alpha = 1,2$; $i = 1,2$

P is the penalty constraint:

$$P = \frac{\gamma}{2} \int_{\Omega} \left\{ \sum_{\beta=1}^{2} \left[\left(u_{3\beta} - \frac{\partial w_{\beta}}{\partial x_2} \right)^2 + \left(u_{4\beta} + \frac{\partial w_{\beta}}{\partial x_1} \right)^2 \right] \right\} d\Omega \tag{54}$$

where γ is the penalty parameter. As γ is assigned successively larger values, the constraints of Eq. (41) are satisfied more exactly in the least squares sense. In the internal strain energy expression, Eq. (51), the transverse normal stress term requires special attention. The constant transverse normal stress leads to an asymmetric stiffness matrix. Thus, instead of the constant transverse normal stress $\bar{\sigma}_{33}^{(k)}$, $\sigma_{33}^{(k)}$ from the constitutive equations should be employed. Moreover, the term $\varepsilon_{33}^{(k)}$ to be substituted in (51) will consist only of the part $\bar{\varepsilon}_{33}^{M(k)}$. The thermal component of the total strain $\bar{\varepsilon}_{33}^{TL(k)}$ will play its role in the nodal force vector. Making the necessary substitutions into Eq. (50), taking the variation, and integrating by parts, the equations of motion are obtained (see [31] for details).

4.2 Finite Element Model

The novel features of the theory suggest developing the finite element model in the form of an 8-noded brick element with five degrees of freedom per node, Figure 33.

This approach has been used with great success [21,22], and has the very important advantage of allowing discretization in both inplane and through-thickness directions without the need for any special multi-point constraints. Elements are assembled in the standard way. Therefore, the element can be used to predict the global response of a laminate using only one (or a small number) of elements through the thickness, and local effects such as interlaminar stresses can be captured by refining the mesh in the thickness direction near the interface(s) of interest. The capability of discretization in the through-thickness direction also enables one to simulate the separation of laminated composites due to delamination.

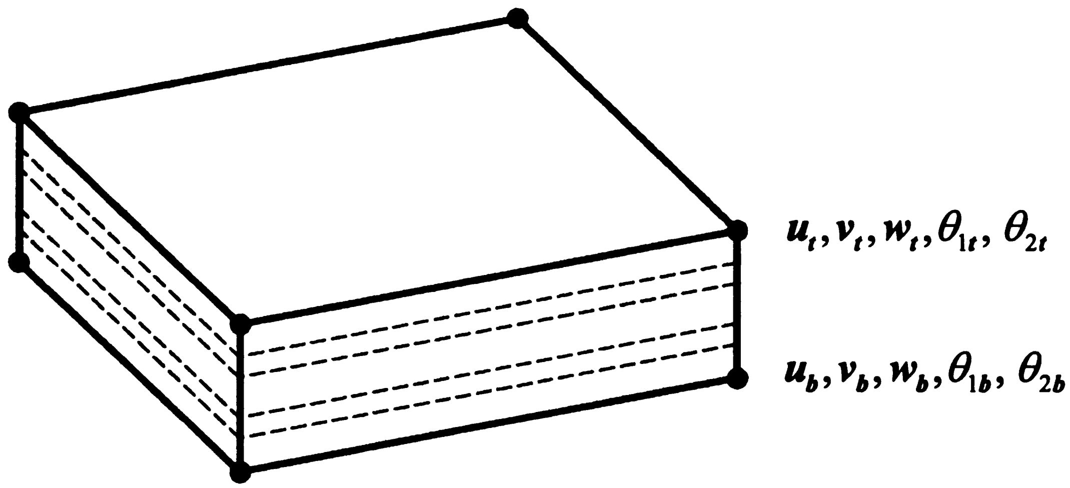


Figure 33 Element topology and nodal degrees of freedom of the structural model.

Inplane displacement and rotational degrees of freedom are approximated by the bilinear Lagrange interpolation functions. For transverse deflection degrees of freedom, an interdependent interpolation concept similar to that developed by Tessler and Hughes [41] is utilized (see [26,31] for details).

The interdependent interpolation scheme alleviates the shear locking problem but does not eliminate it totally. In order to prevent the shear locking phenomenon, the transverse shear strain fields have been made field consistent using the approach presented in [42] (see also [26,31]).

CHAPTER 5 THERMAL STRESS MODEL

5.1 Equivalent Nodal Force Vector Method

The nodal temperature vector is obtained from a thermal analysis adopting the zig-zag thermal lamination theory. Then, the effect of the thermal strain is imposed through an equivalent nodal force vector $\{P\}$. In order to evaluate $\{P\}$ it is necessary to compute at the elemental level $\{P_e\}$ by the following relation:

$$\{P_e\} = \sum_{k=1}^{N} \int_{V_k} \left[B^{(k)}\right]^T \left[D^{(k)}\right] \left\{\varepsilon^{T(k)}\right\} dV_k \tag{55}$$

where $\left[B^{(k)}\right]$ and $\left[D^{(k)}\right]$ are, respectively, the nodal strain-displacement relation matrix and the elasticity matrix governing the layer *k*. $\left\{\varepsilon^{T(k)}\right\}$ is the thermal strain vector evaluated at the layer level by:

$$\begin{Bmatrix} \varepsilon_{11}^{T(k)} \\ \varepsilon_{22}^{T(k)} \\ \varepsilon_{33}^{T(k)} \\ \gamma_{23}^{T(k)} \\ \gamma_{13}^{T(k)} \\ \gamma_{12}^{T(k)} \end{Bmatrix} = \begin{Bmatrix} \alpha_1^{(k)} \cdot \Delta T^{(k)} \\ \alpha_2^{(k)} \cdot \Delta T^{(k)} \\ \alpha_3^{(k)} \cdot \Delta T^{(k)} \\ 0 \\ 0 \\ \alpha_{12}^{(k)} \cdot \Delta T^{(k)} \end{Bmatrix} \tag{56}$$

The integration is accomplished numerically by Gauss quadrature, using one point along x_1 and x_2 directions and two along x_3 in each layer. The zig-zag thermal lamination theory is once again used to find the temperature at the integration points for each layer in terms of the nodal temperatures (see Eqs. (12,13)).

It has been shown in the previous section, that the term $\bar{\varepsilon}_{33}^{TL(k)}$, defined in (45), should be subtracted from the total strain $\bar{\varepsilon}_{33}^{(k)}$ to obtain the corrected transverse strain and to maintain consistency with the constant transverse normal stress $\bar{\sigma}_{33}$. This is accomplished by modifying the term $\varepsilon_{33}^{T(k)}$ in (56):

$$\varepsilon_{33}^{T(k)} = \alpha_3^{(k)} \cdot \Delta T - \bar{\varepsilon}_{33}^{TL(k)} \tag{57}$$

The equivalent nodal force vector $\{P\}$ will be added to the external forces applied to get the total force vector. Then the system is solved and the displacement vector $\{u\}$ is found. The strain vector is calculated at the integration points for each layer by use of the relation:

$$\left\{\varepsilon^{(k)}\right\} = \left[B^{(k)}\right]\left\{u^{(e)}\right\} \tag{58}$$

Finally, the stress vector $\left\{\sigma^{(k)}\right\}$ for a single layer is defined by:

$$\left\{\sigma^{(k)}\right\} = \left[D^{(k)}\right]\left(\left\{\varepsilon^{(k)}\right\} - \left\{\varepsilon^{T(k)}\right\}\right) \tag{59}$$

The possibility of having inconsistent thermal stress evaluation, see [34,35], is avoided by calculating the stress at the center of the element with respect to the x_1 and x_2 dimensions. Moreover, at this point stresses are super-convergent [43].

5.2 Numerical Results

A computer program, realized in Fortran77, has been used to apply the presented thermal stress model to the analysis of laminate plates. The original structural code of Cho [31] was modified for this purpose. It is not reported in an appendix due to its length.

To assess the effectiveness of the presented model in evaluating thermal stresses and displacements, two different simply supported laminated plates were studied: a sandwich and a four-layered antisymmetric cross-ply laminate ([0/90/0/90]).

In both cases the plates are subjected to a temperature rise at the external faces that has a double sinusoidal variation in the x_1 - x_2 plane:

$$\begin{aligned} T_{top}(x_1,x_2) &= T_t \cdot \sin\left(\frac{\pi \cdot x_1}{L_1}\right) \cdot \sin\left(\frac{\pi \cdot x_2}{L_2}\right), \\ T_{bottom}(x_1,x_2) &= T_b \cdot \sin\left(\frac{\pi \cdot x_1}{L_1}\right) \cdot \sin\left(\frac{\pi \cdot x_2}{L_2}\right) \end{aligned} \tag{60}$$

For the sandwich the value of the temperatures T_t and T_b have been taken equal to -1 and 1, respectively, yielding a temperature gradient through the thickness. The antisymmetric laminate is loaded with a uniform temperature through the thickness produced by the conditions $T_t = T_b = 1$.

The plates are square $L_1 = L_2 = a$ and are constrained by the following boundary conditions:

$$\begin{aligned} u_2 = 0,\ u_3 = 0,\ \theta_1 = 0 \quad \text{at} \quad x_1 = 0, L_1 \\ u_1 = 0,\ u_3 = 0,\ \theta_2 = 0 \quad \text{at} \quad x_2 = 0, L_2 \end{aligned} \tag{61}$$

These boundary conditions allow an exact solution to be obtained. In the case of the sandwich panel the exact solution is provided by Cho and Park [36]. The exact solutio

[13,14

lamina

follow

where

coeffic

of the

Where

directi

follow

layers

are co

the ex

5.2.1

first.

extern

solution for the antisymmetric laminate has been calculated by Rolfes, Noor, and Sparr [13,14].

In all graphs the x_3 dimension is normalized by the total thickness of the laminate, h. Moreover, displacements and stresses will be given according to the following non-dimensional forms:

$$\bar{\sigma}_{ij} = \frac{\sigma_{ij}}{\alpha_r \bar{T} E_r}, \quad \bar{u}_i = \frac{u_i}{\alpha_r \bar{T} h} \tag{62}$$

where $\alpha_r = \dfrac{10^{-6}}{K}$ and $E_r = 10^6 GPa$ are the reference thermal expansion and elastic coefficients, and $\bar{T} = |T_b| = |T_t|$.

Taking advantage of the symmetry of load and boundary conditions, one quarter of the plate has been analyzed with three different meshes: 8x8x2, 4x4x4, and 2x2x2. Where the three numbers indicate respectively the subdivisions in x_1, x_2 and x_3 directions. For example, given that the antisymmetric laminate contains 4 layers, it follows that in a mesh with 2 divisions along the thickness each element includes 2 layers.

Since stresses are evaluated at the centroid of each element, the results presented are computed at the center of the element that is nearest the point of the plate for which the exact solution is given.

5.2.1 Sandwich Panel

The steady-state thermo elastic bending of a sandwich square plate is analyzed first. The thickness of the soft core is equal to $0.6h$, while that of each of the two stiff external layers is $0.2h$, where h is the thickness of the plate. The physical properties of the two materials are the following:

Stiff external layers (Unidirectional graphite-epoxy composite)

$$E_L = 200GPa,\ E_T = 8GPa,\ G_{LT} = 5GPa,\ G_{TT} = 2.2GPa,\ \nu_{LT} = 0.25,\ \nu_{TT} = 0.35,$$
$$\alpha_L = -2\times\frac{10^{-6}}{K},\ \alpha_T = 50\times\frac{10^{-6}}{K},\ k_L = 50\frac{W}{K\cdot m},\ k_T = 0.5\frac{W}{K\cdot m} \tag{63}$$

Soft core

$$E_I = 1GPa,\ E_T = 2GPa,\ G_{IT} = 0.8GPa,\ G_{TT} = 3.7GPa,\ \nu_{IT} = 0.25, \nu_{TT} = 0.35,$$
$$\alpha_I = \alpha_T = 30\times\frac{10^{-6}}{K},\ \ k_I = k_L = 50\frac{W}{K\cdot m} \tag{64}$$

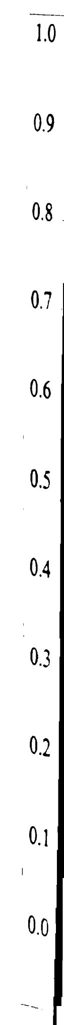
Where subscripts L and T stand for the directions parallel and perpendicular to the fibers for the external layers, whereas I and T mark inplane and transverse directions for the soft core. Results are presented for two different aspect ratios of the sandwich plate: 4 and 20.

5.2.1.1 Demonstration of Thermal Zig-Zag Effects

In subchapter 2.1.1 a numerical example was presented in order to demonstrate the types of temperature distribution that exist in a laminate. It has been proved that the often assumed linear variation of temperature through the thickness of the laminate can produce considerable errors. To further underline this statement, the exact temperature distribution (from [32]) through the thickness at the center of the studied sandwich plate $(\frac{a}{2},\frac{a}{2})$, is compared with the solution produced by the zig-zag thermal theory and with that obtained by a linear variation in Figure 34. A remarkable difference in the quality of the results between the two theories can be observed. The errors in temperature distribution produced by the assumption of a linear variation will have a negative effect on th

graph

assun



Figu

on the accuracy of stresses and displacements. To emphasize this problem, in all the graphs presenting the solutions for the sandwich plate, the results obtained with an assumed linear variation of temperature have been included.

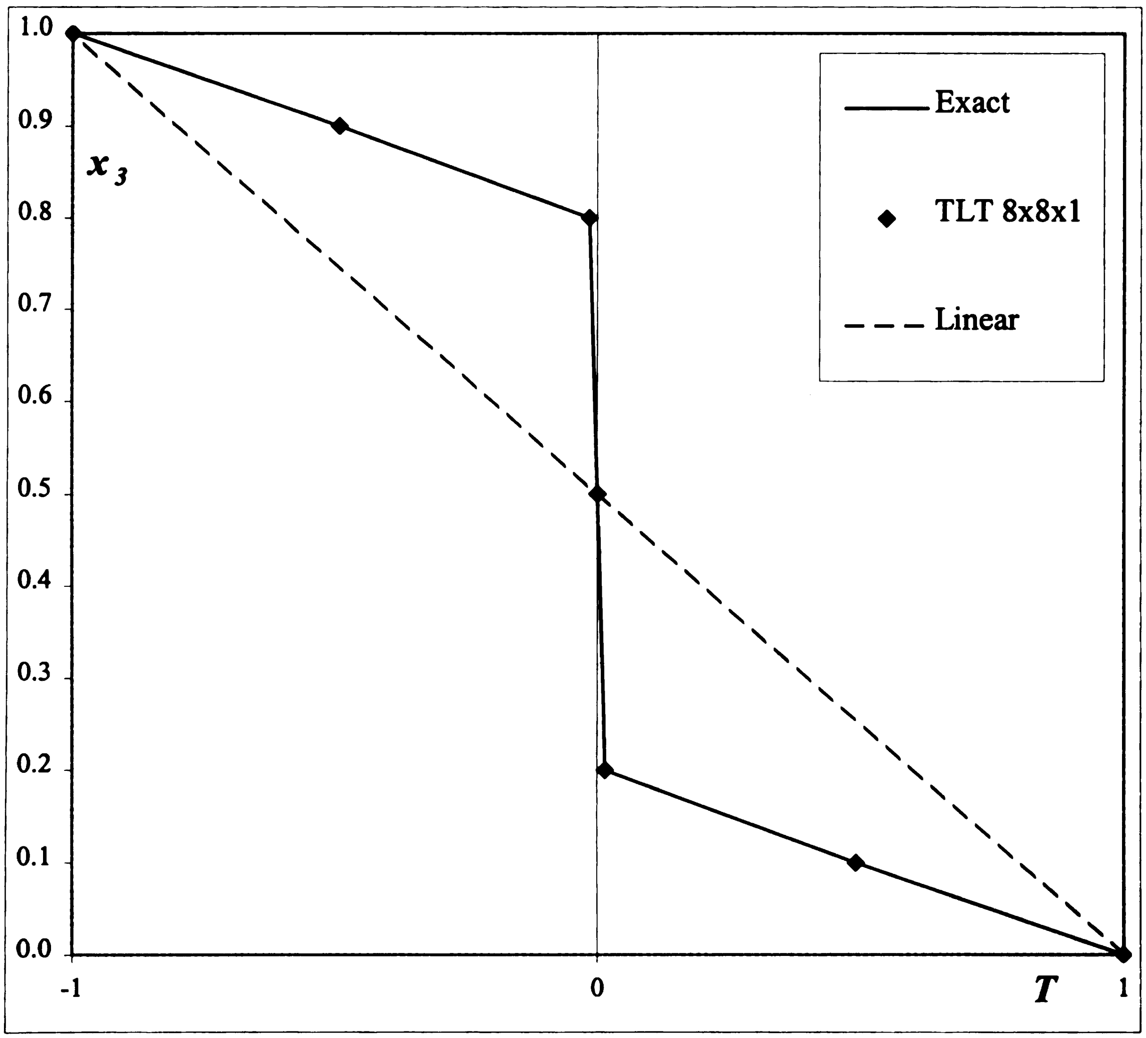


Figure 34 ***Through-thickness distribution of temperature at $x_1 = x_2 = a/2$ in the sandwich plate with aspect ratio 20.***

5.2.1.2 Sandwich with Aspect Ratio a/h=20

Figures 35 shows the predicted through the thickness variation of the inplane displacement u_1 at $x_1 = 0$ and $x_2 = \frac{a}{2}$, whereas Figure 36 illustrates the distribution of u_2 at $x_1 = \frac{a}{2}$ and $x_2 = 0$. The normal stresses σ_{11}, σ_{22} and σ_{33} in the center of the plate ($x_1 = \frac{a}{2}$ and $x_2 = \frac{a}{2}$) are plotted in Figures 37, 38 and 39, respectively. The transverse shear stresses are represented in Figures 40 and 41 (σ_{23} at $x_1 = \frac{a}{2}$ and $x_2 = 0$, σ_{13} at $x_1 = 0$ and $x_2 = \frac{a}{2}$). The inplane shear stress σ_{12} is evaluated at the corner of the plate ($x_1 = 0$ and $x_2 = 0$) and is shown in Figure 42.

A close agreement between the exact analytical solution and the new theory is obtained with all meshes, and convergence toward the exact solution is obtained as the mesh is refined. The only exception is represented by σ_{33}, where the assumed constant through-thickness variation is not sufficient to capture the true variation of σ_{33}. Because σ_{33} is about 5 order of magnitude smaller than σ_{11} and σ_{22} in this problem, the current predictions are considered to be satisfactory.

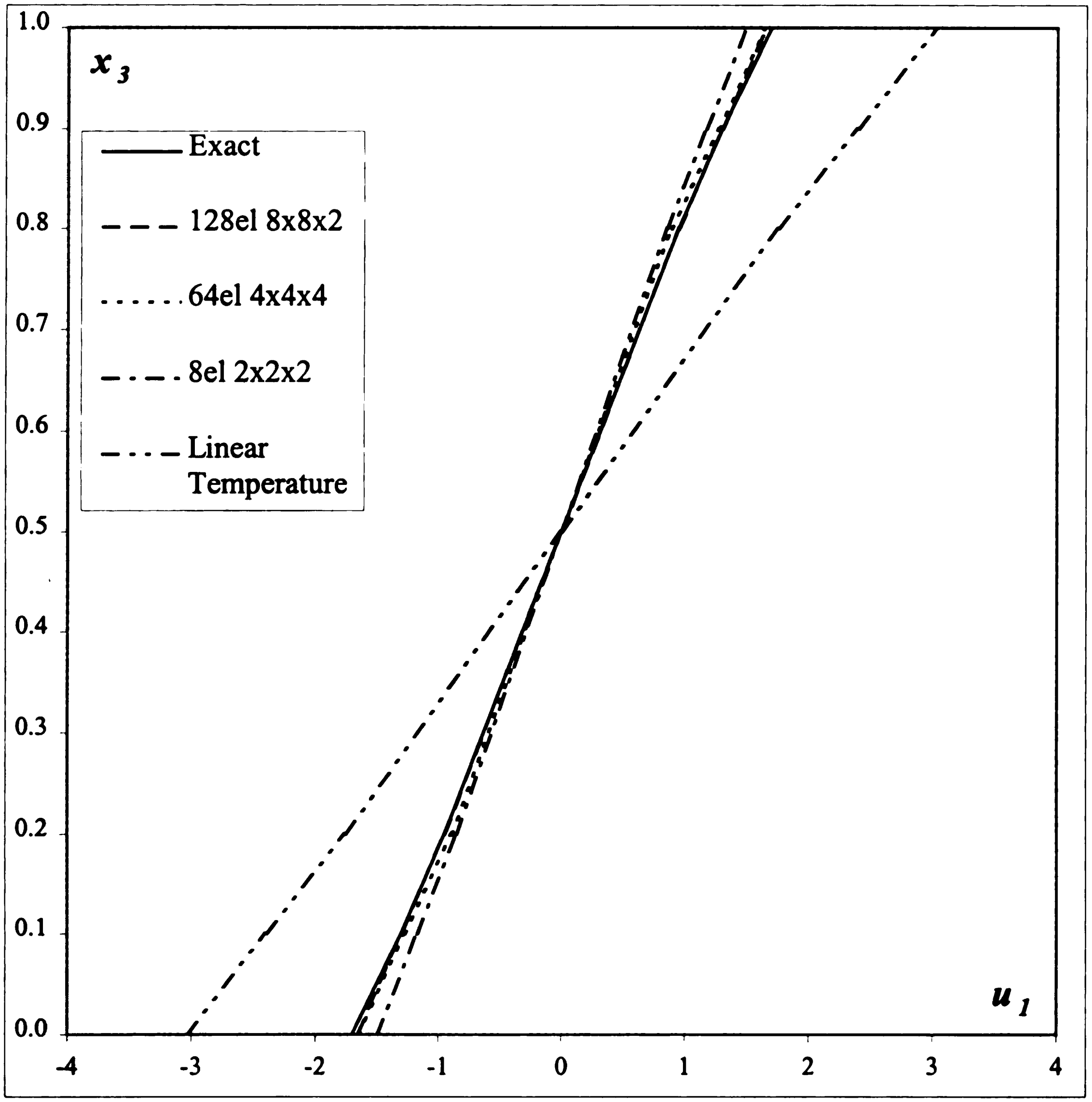


Figure 35 ***Through-thickness variation of the displacement u_1 at $x_1 = 0$ and $x_2 = a/2$ in the sandwich plate with aspect ratio 20.***

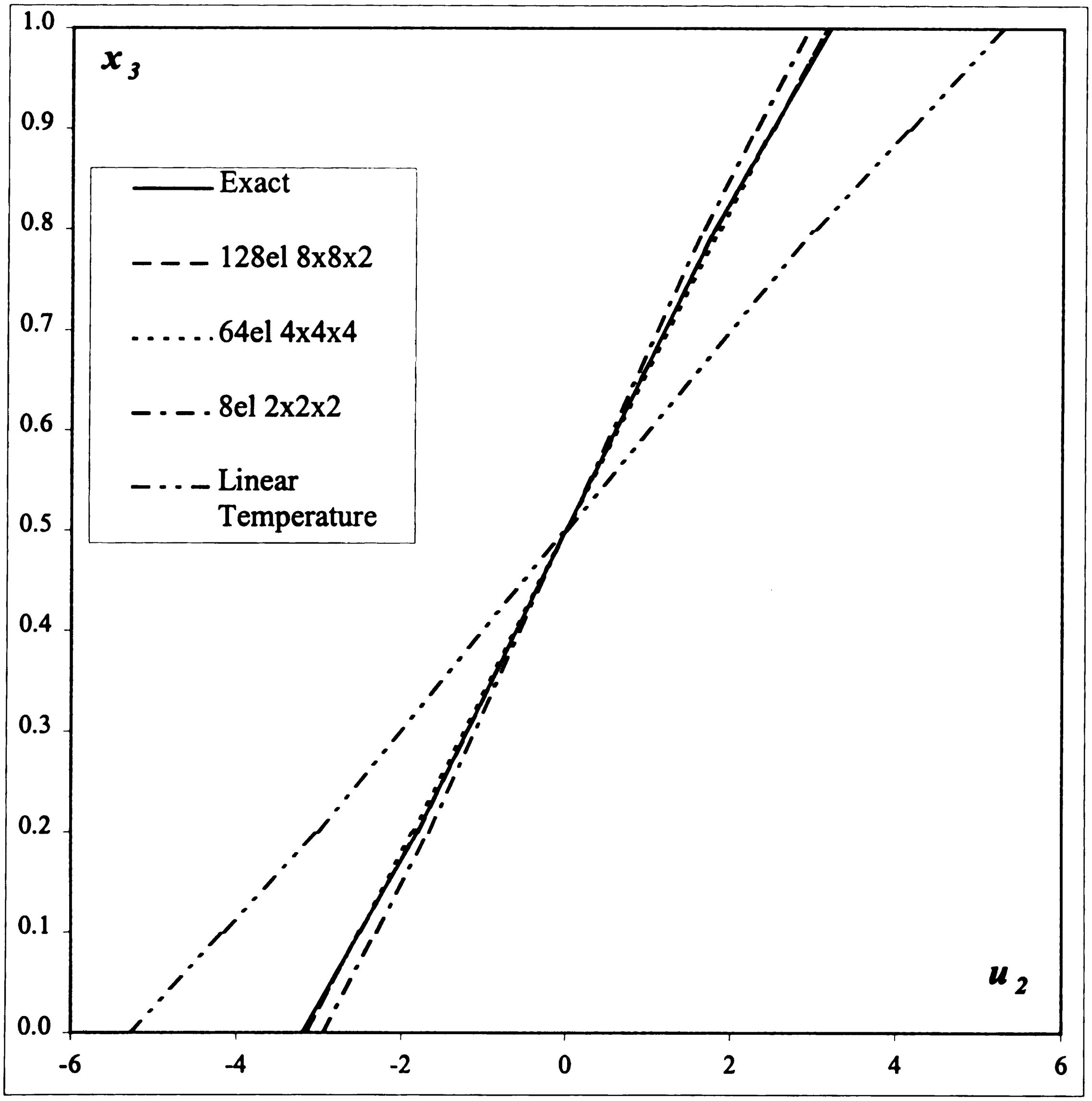


Figure 36 ***Through-thickness variation of the displacement u_2 at $x_1 = a/2$ and $x_2 = 0$ in the sandwich plate with aspect ratio 20.***

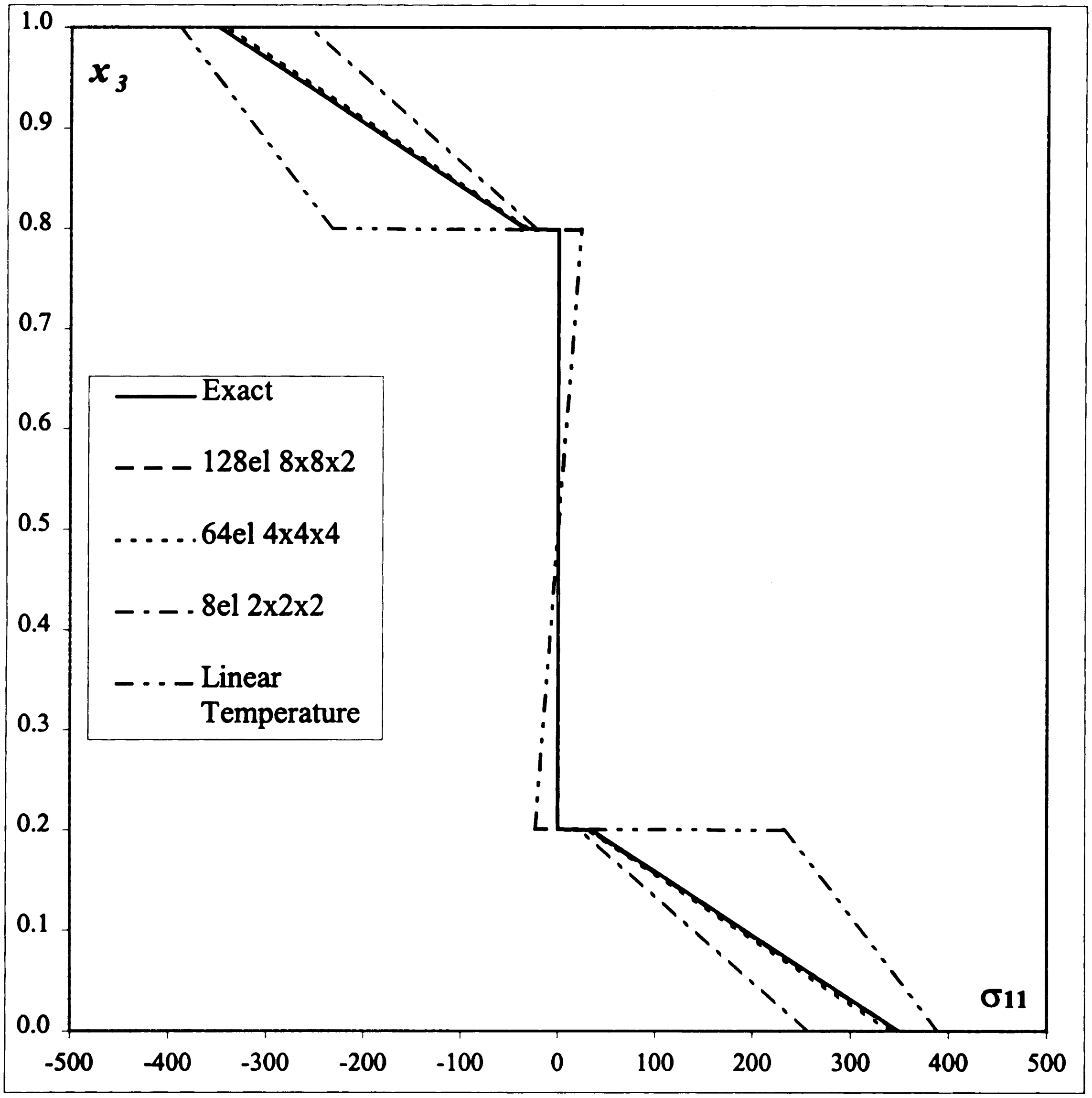


Figure 37 ***Through-thickness variation of the stress σ_{11} at $x_1 = a/2$ and $x_2 = a/2$ in the sandwich plate with aspect ratio 20.***

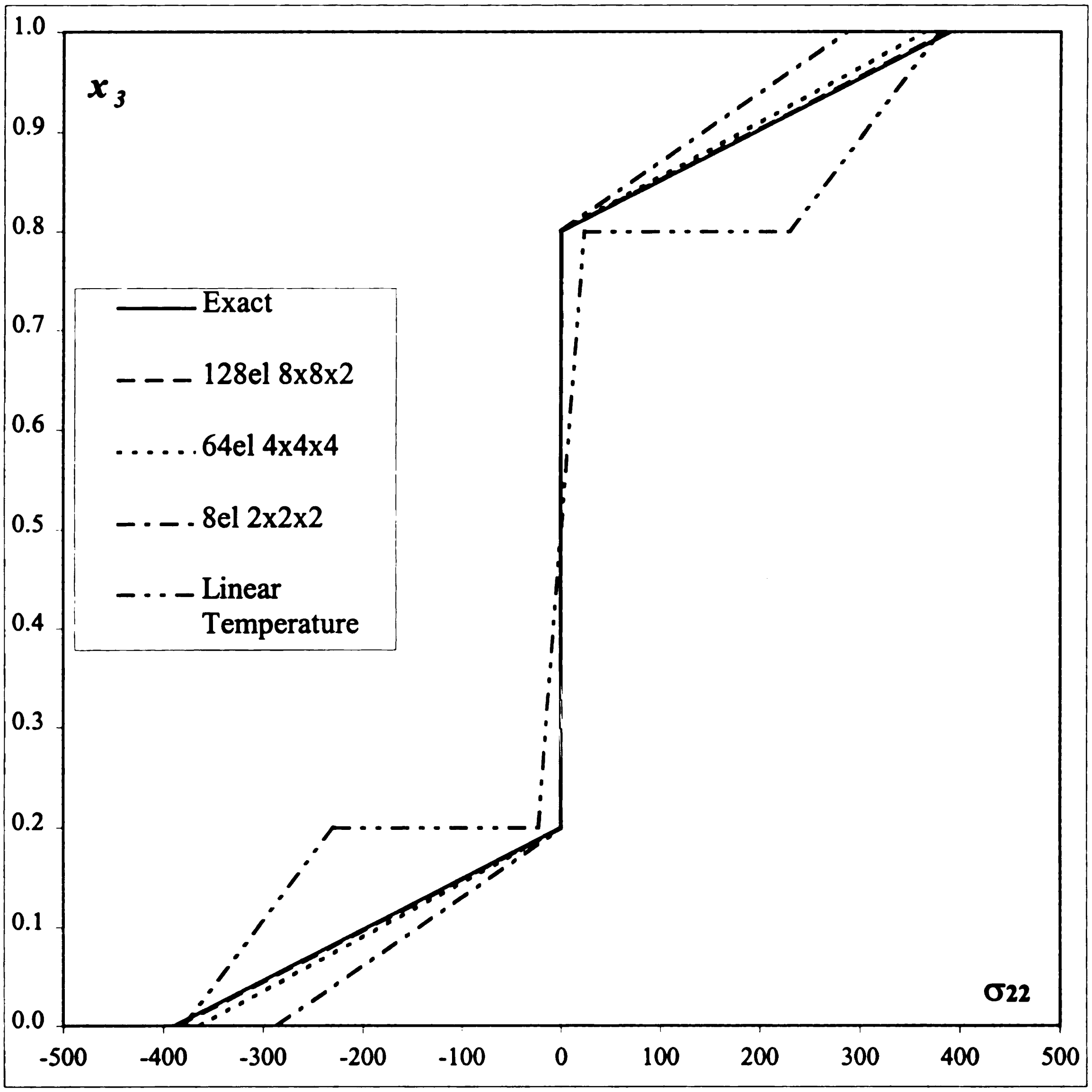


Figure 38 ***Through-thickness variation of the stress σ_{22} at $x_1 = a/2$ and $x_2 = a/2$ in the sandwich plate with aspect ratio 20.***

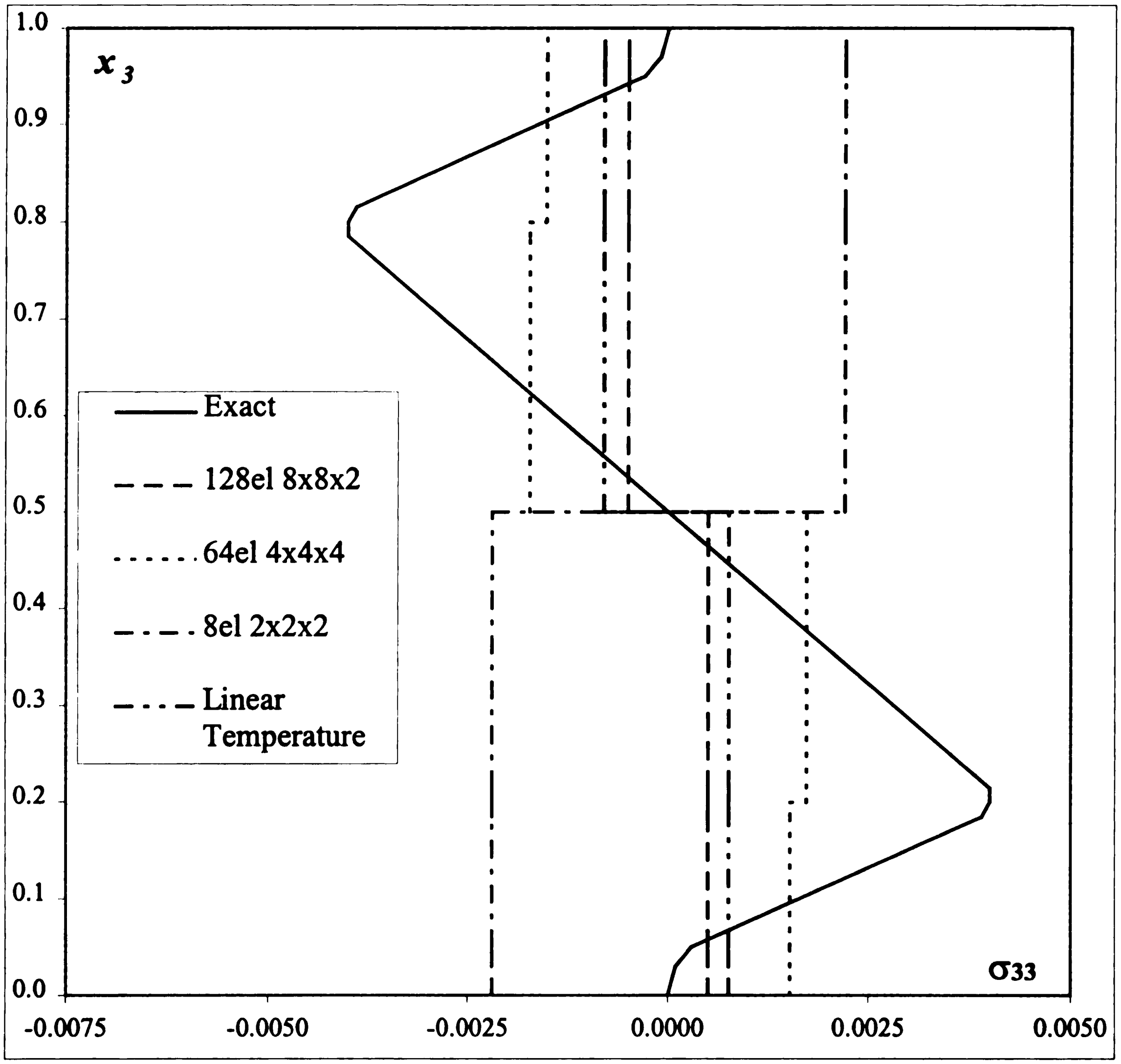


Figure 39 ***Through-thickness variation of the stress σ_{33} at $x_1 = a/2$ and $x_2 = a/2$ in the sandwich plate with aspect ratio 20.***

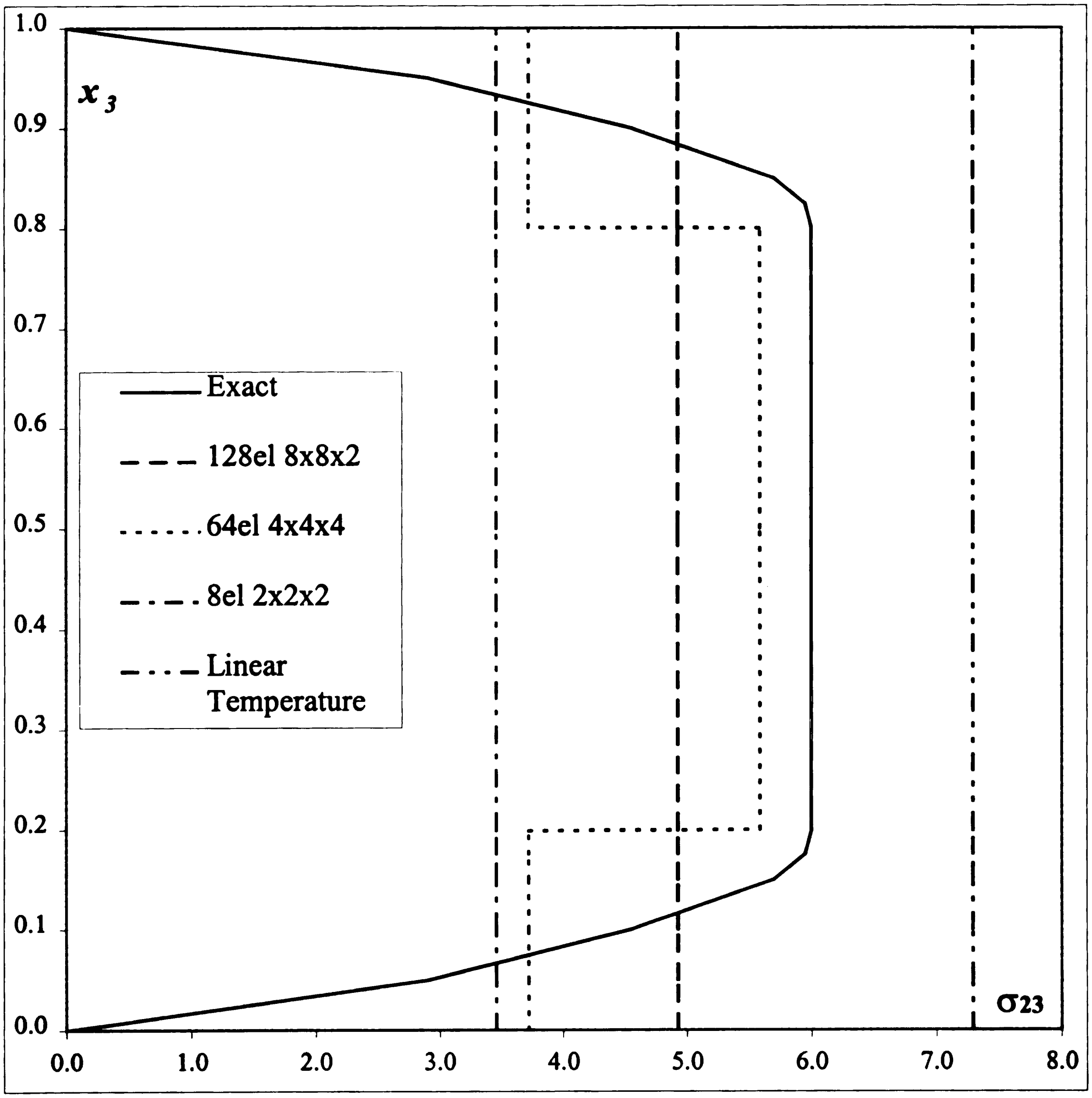


Figure 40 ***Through-thickness variation of the stress σ_{23} at $x_1 = a/2$ and $x_2 = 0$ in the sandwich plate with aspect ratio 20.***

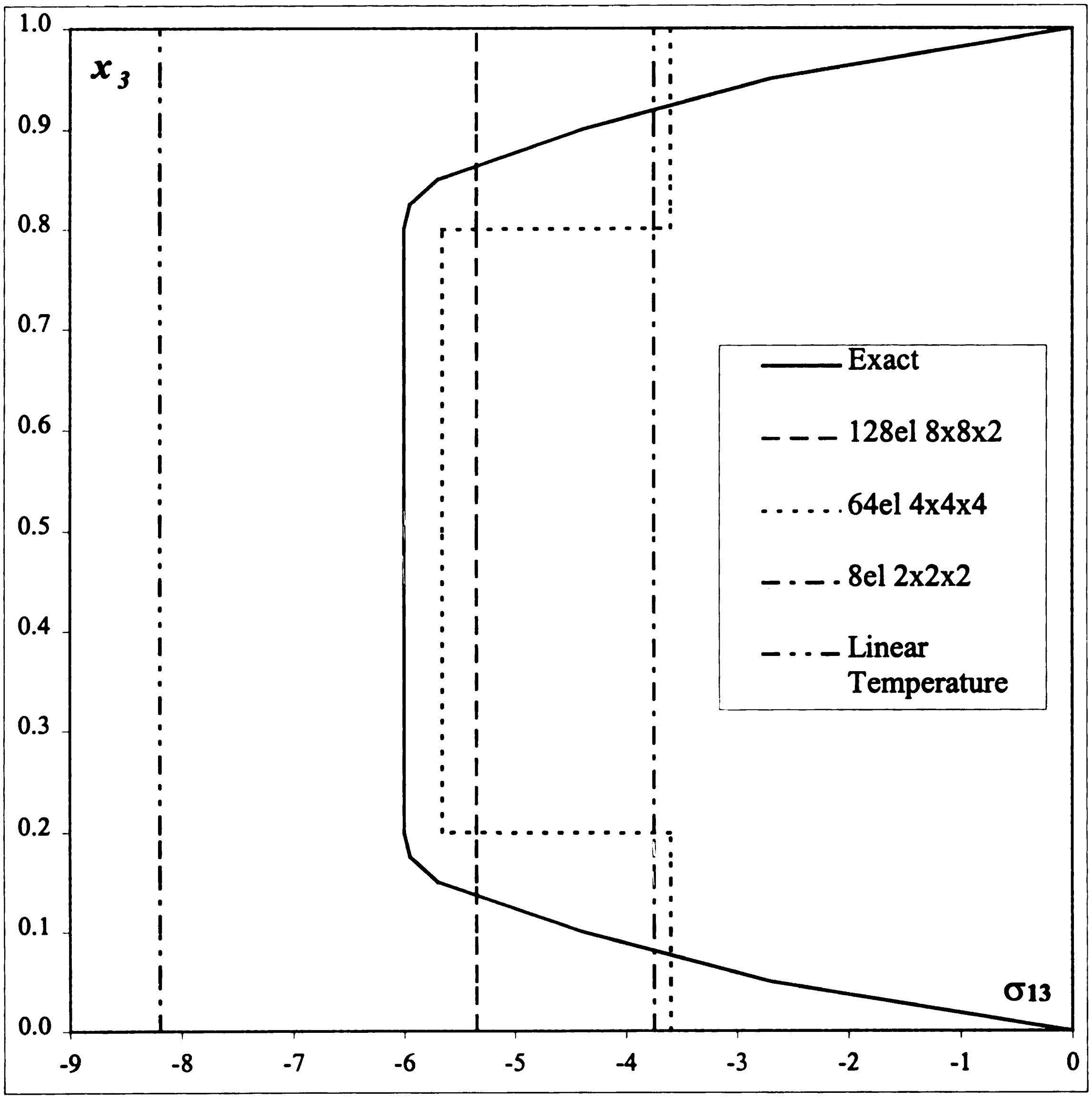


Figure 41 ***Through-thickness variation of the stress σ_{13} at $x_1 = 0$ and $x_2 = a/2$ in the sandwich plate with aspect ratio 20.***

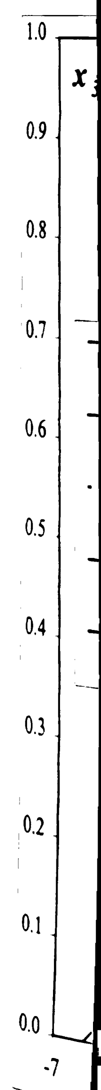


Figure 42

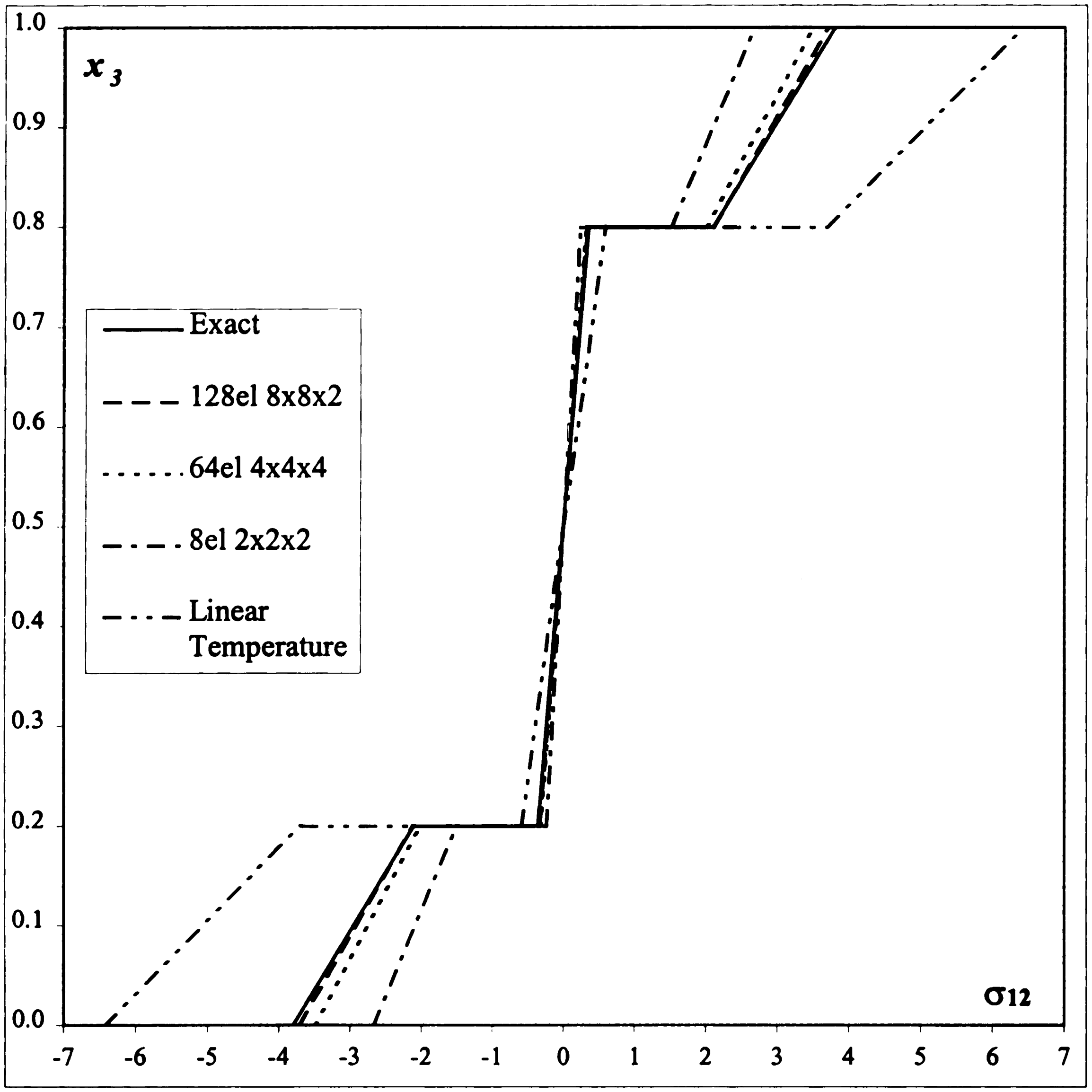


Figure 42 ***Through-thickness variation of the stress σ_{12} at $x_1 = 0$ and $x_2 = 0$ in the sandwich plate with aspect ratio 20.***

5.2.1.3 Sandwich with Aspect Ratio a/h=4

An aspect ratio of 4 was chosen in the present example in order to highlight the ability of the present model to obtain accurate results for very thick laminates.

All displacements and stresses are shown at the same positions in the plate as for the previous case. Figures 43 and 44 show the plot of the displacements u_1 and u_2. Normal stresses σ_{11}, σ_{22} and σ_{33} in the center of the plate are shown in Figures 45-47. The shear stresses σ_{23}, σ_{13} and σ_{12} are represented in Figures 48-50.

In this case the highly non-linear displacements, especially u_1, are not captured exactly by use of the adopted meshes. As a consequence, the accuracy of the transverse shear stresses σ_{23} and σ_{13} show a small decrease. Prediction of normal stresses maintains a high level of precision, with the only exception being σ_{33} for the same reasons expressed above.

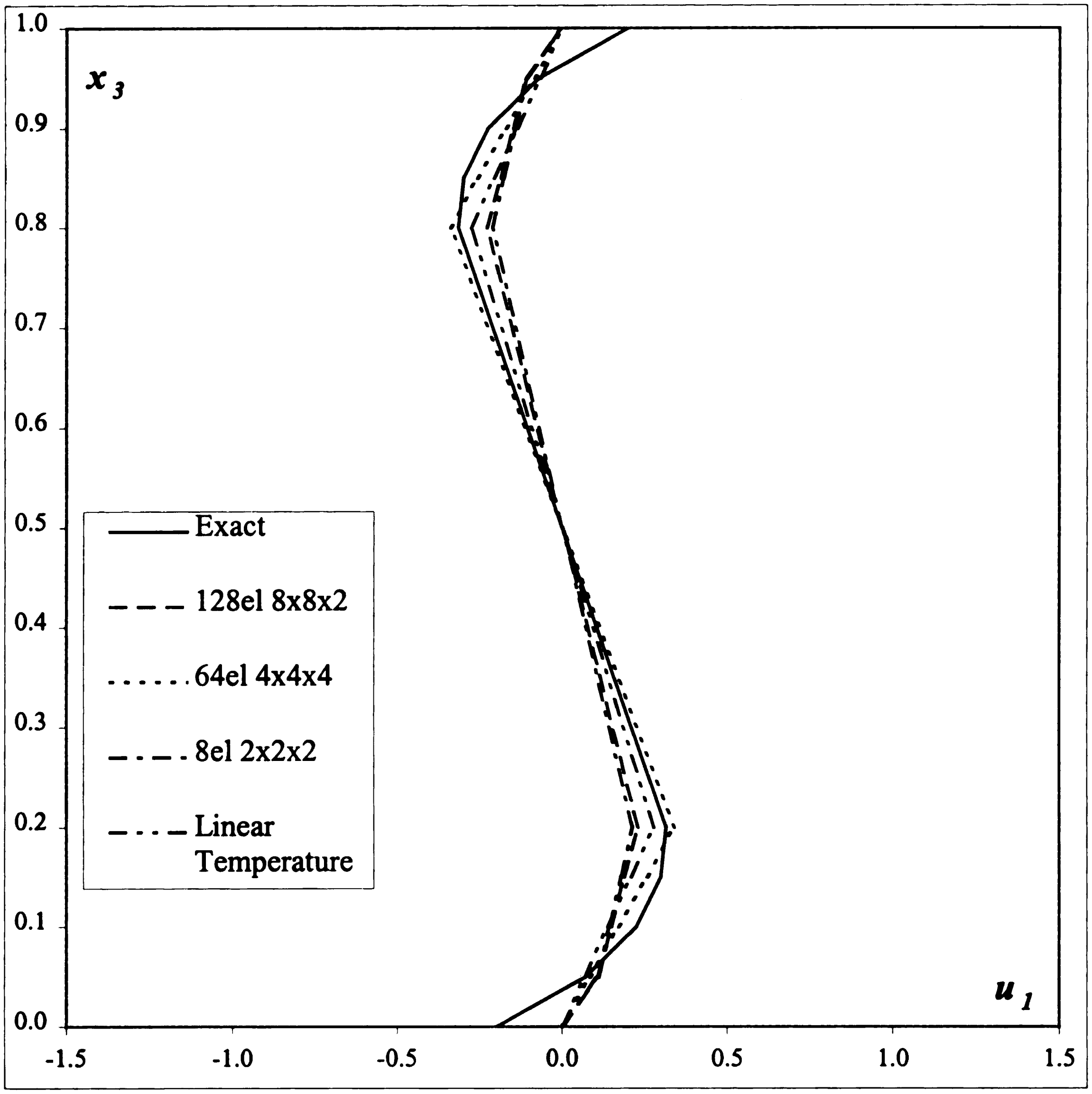


Figure 43 ***Through-thickness variation of the displacement u_1 at $x_1 = 0$ and $x_2 = a/2$ in the sandwich plate with aspect ratio 4.***

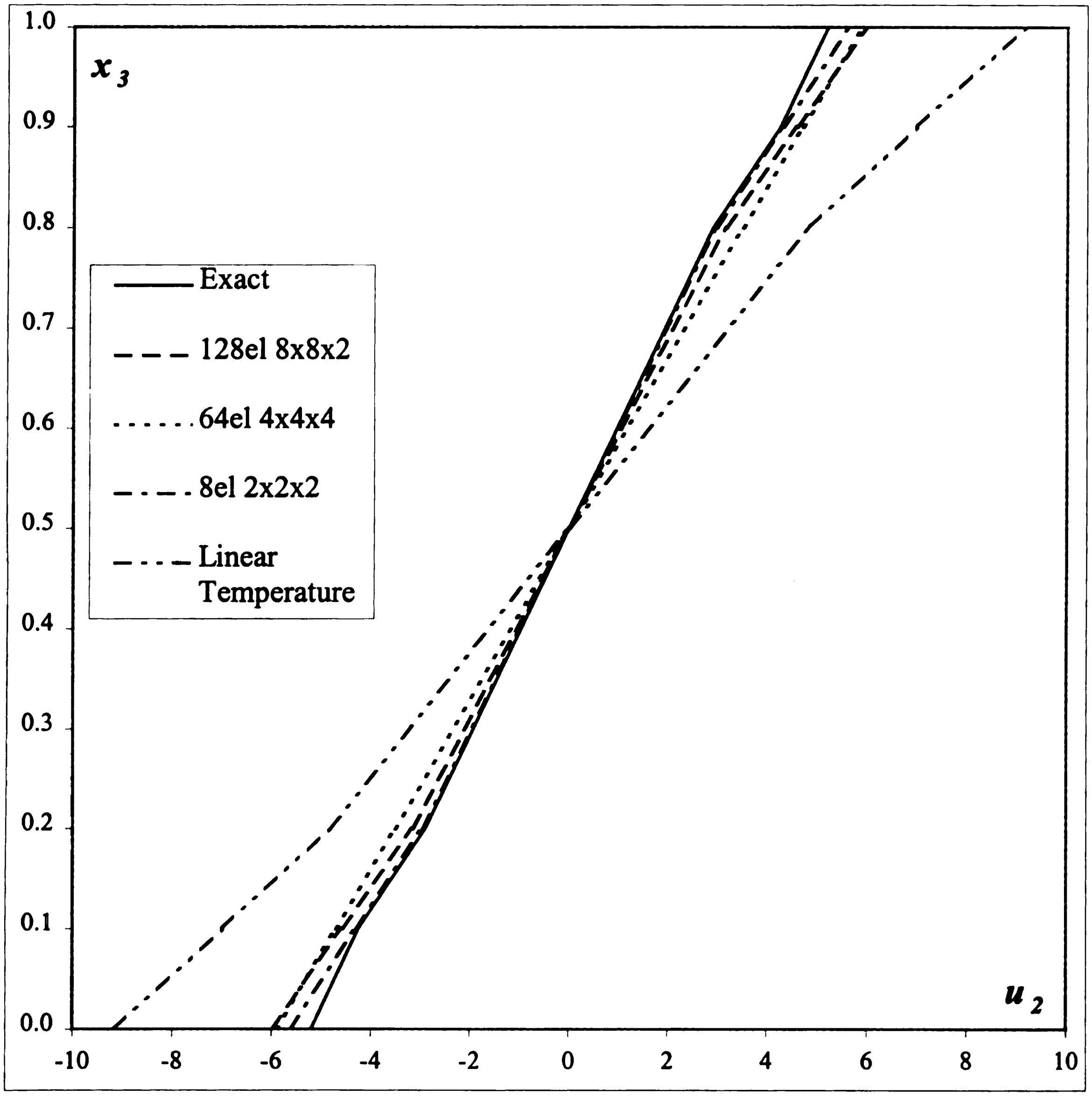


Figure 44 ***Through-thickness variation of the displacement u_2 at $x_1 = a/2$ and $x_2 = 0$ in the sandwich plate with aspect ratio 4.***

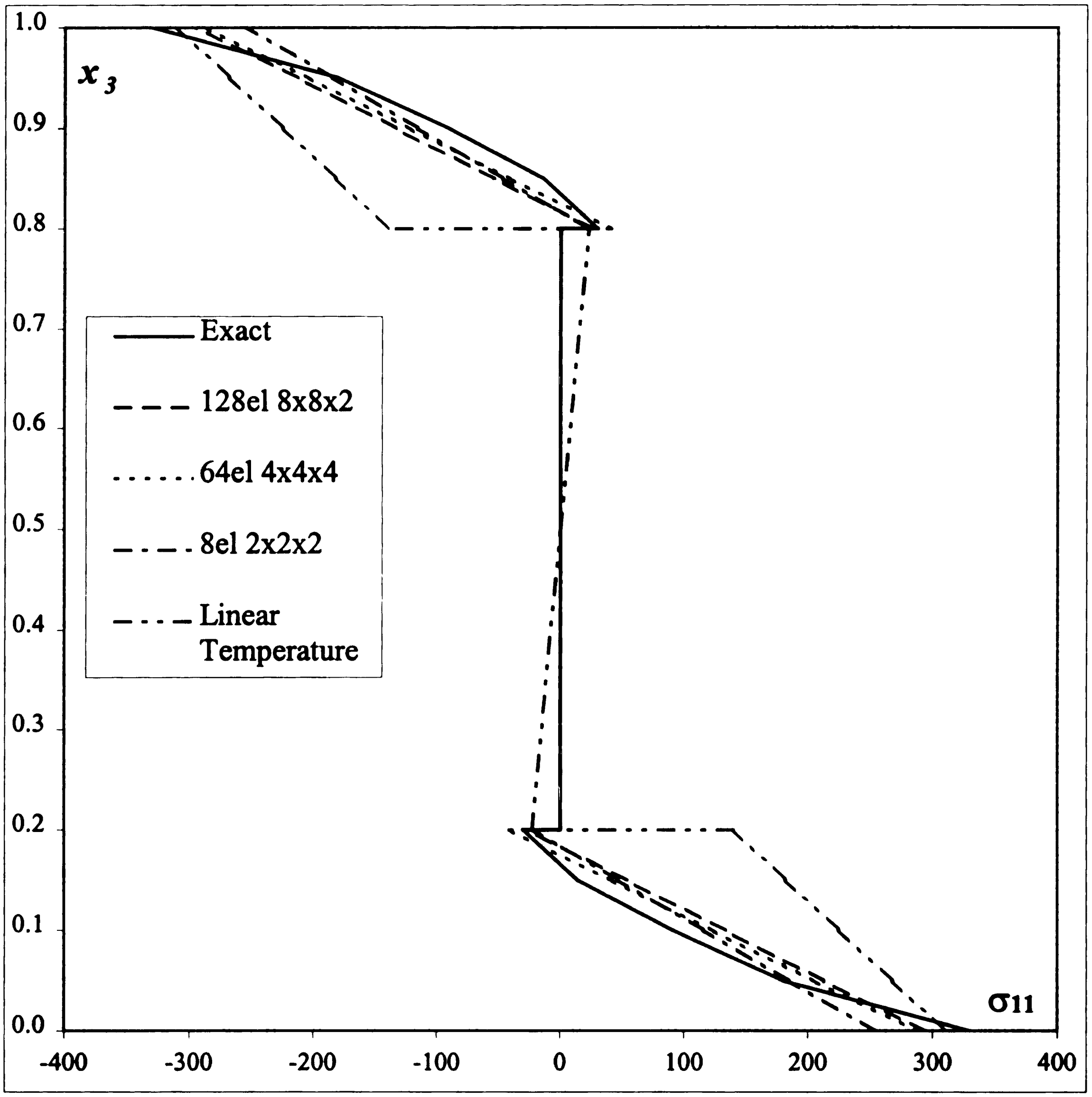


Figure 45 ***Through-thickness variation of the stress σ_{11} at $x_1 = a/2$ and $x_2 = a/2$ in the sandwich plate with aspect ratio 4.***

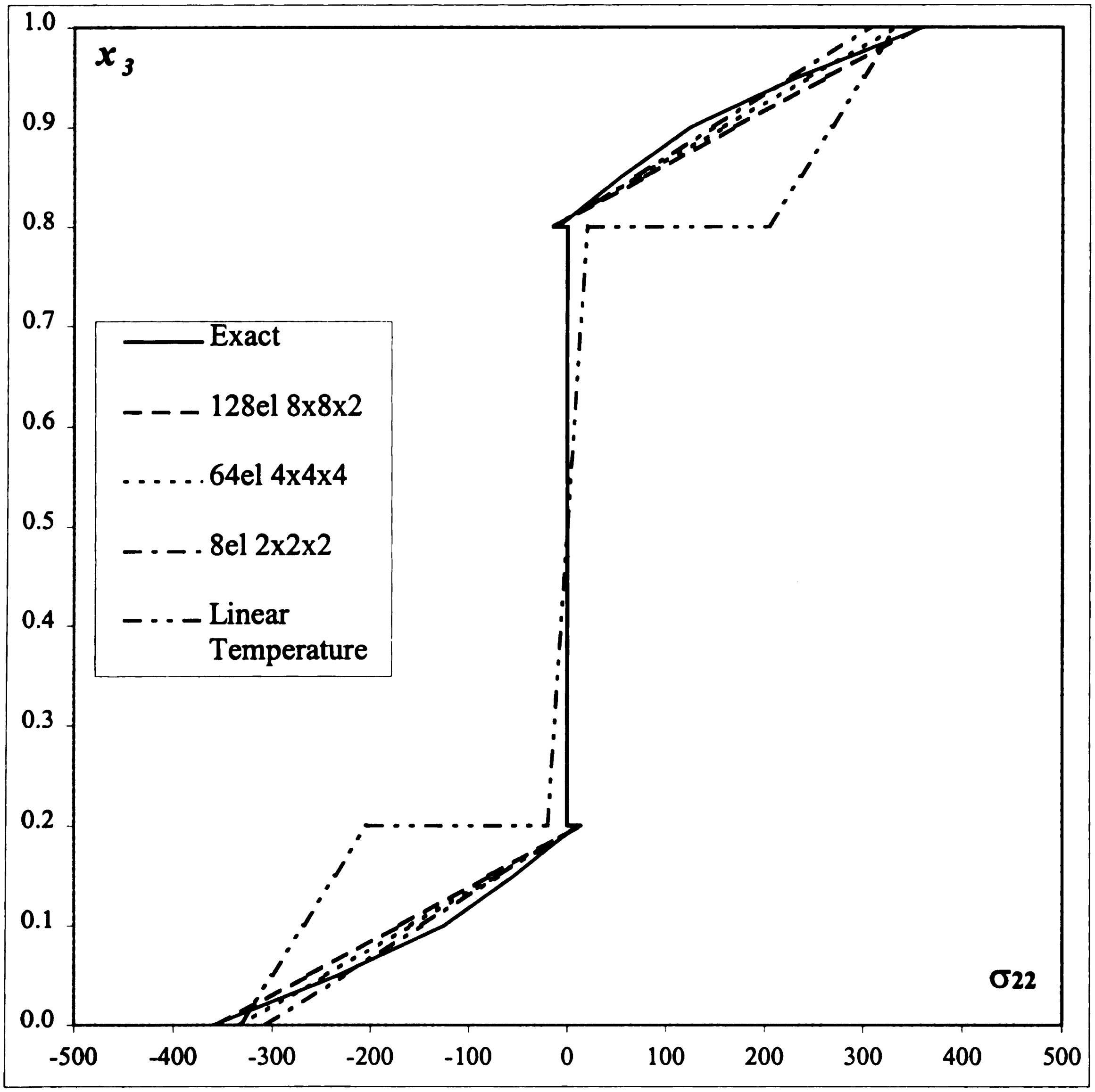


Figure 46 ***Through-thickness variation of the stress σ_{22} at $x_1 = a/2$ and $x_2 = a/2$ in the sandwich plate with aspect ratio 4.***

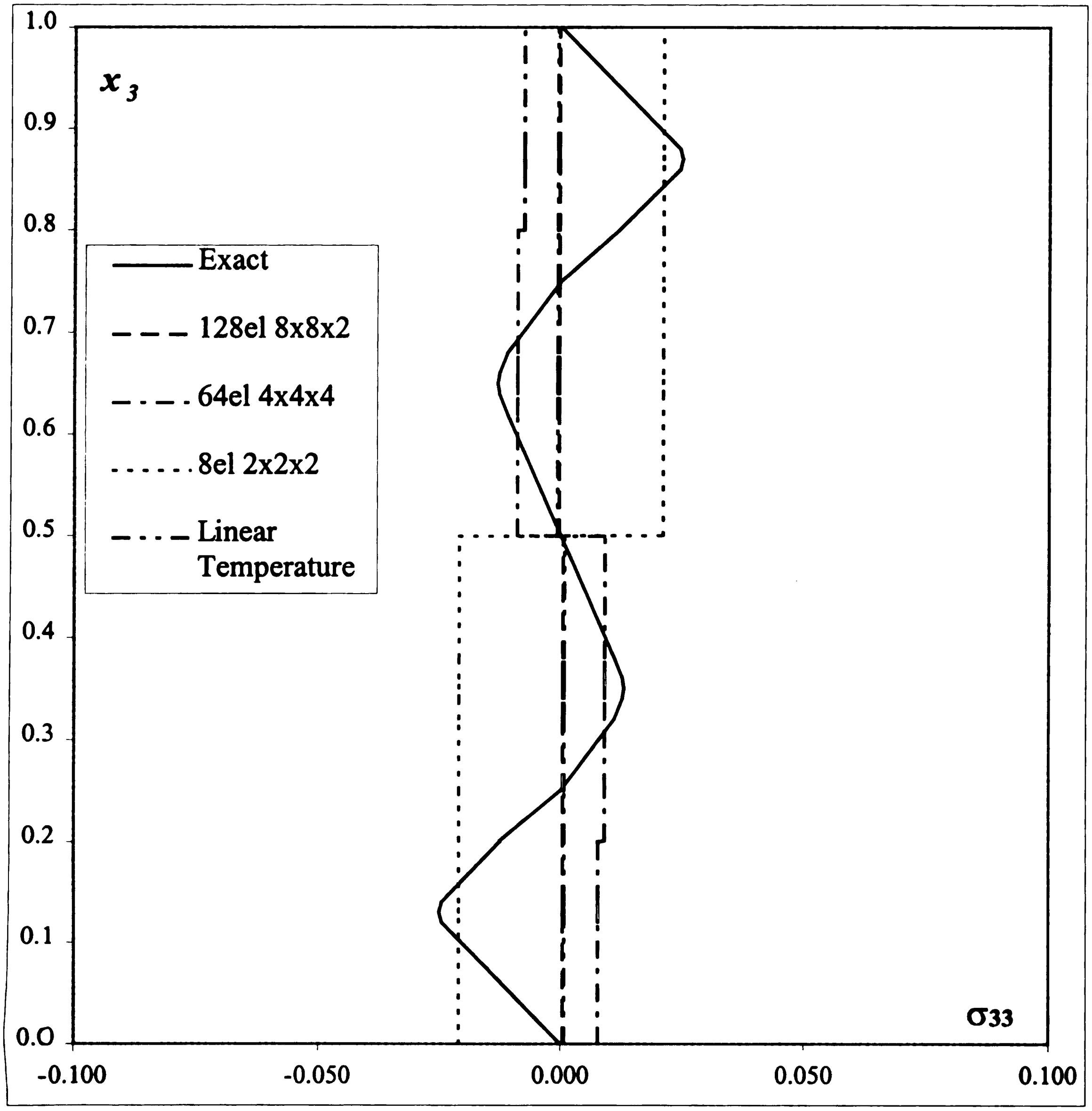


Figure 47 ***Through-thickness variation of the stress σ_{33} at $x_1 = a/2$ and $x_2 = a/2$ in the sandwich plate with aspect ratio 4.***

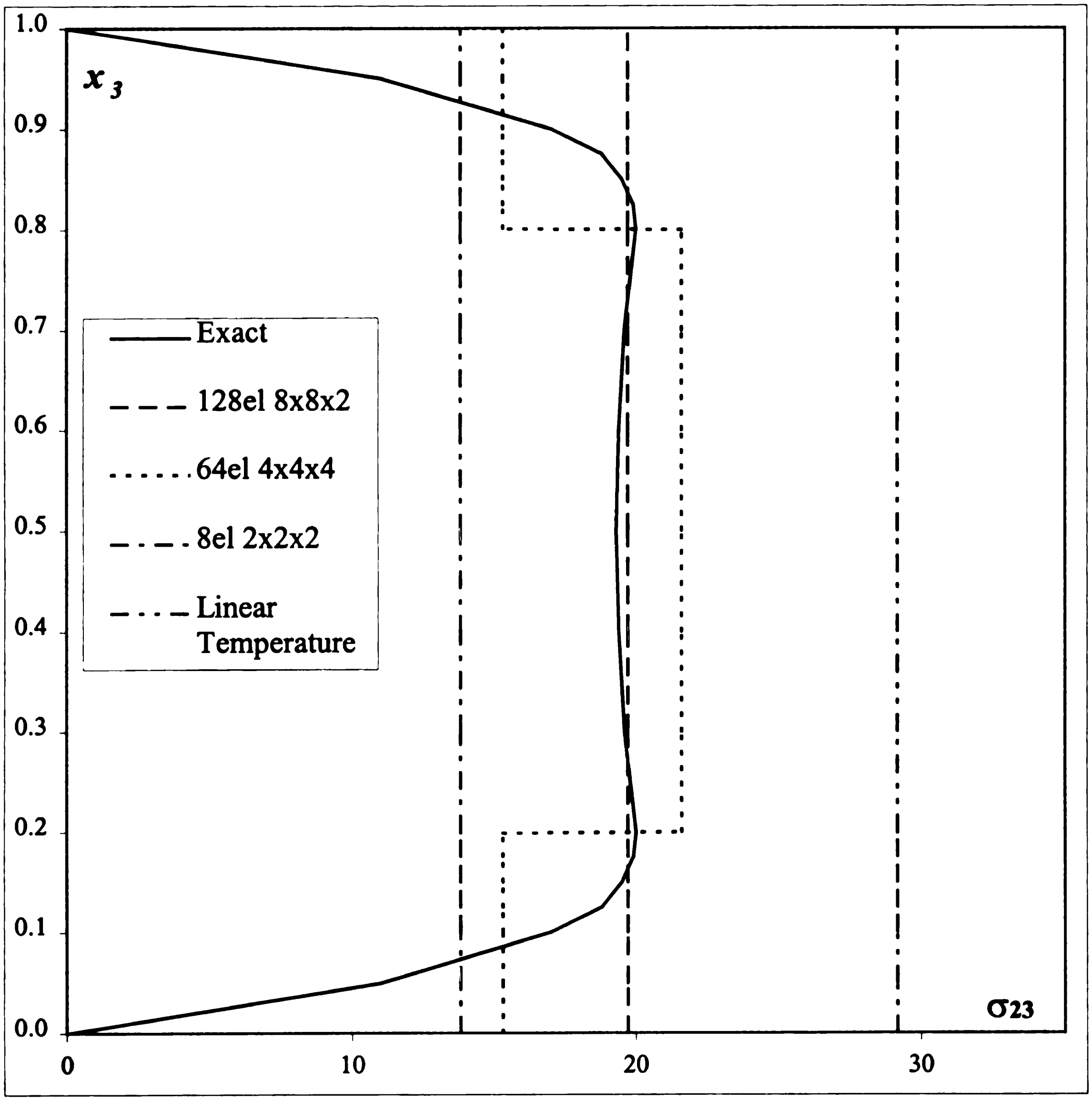


Figure 48 ***Through-thickness variation of the stress σ_{23} at $x_1 = a/2$ and $x_2 = 0$ in the sandwich plate with aspect ratio 4.***

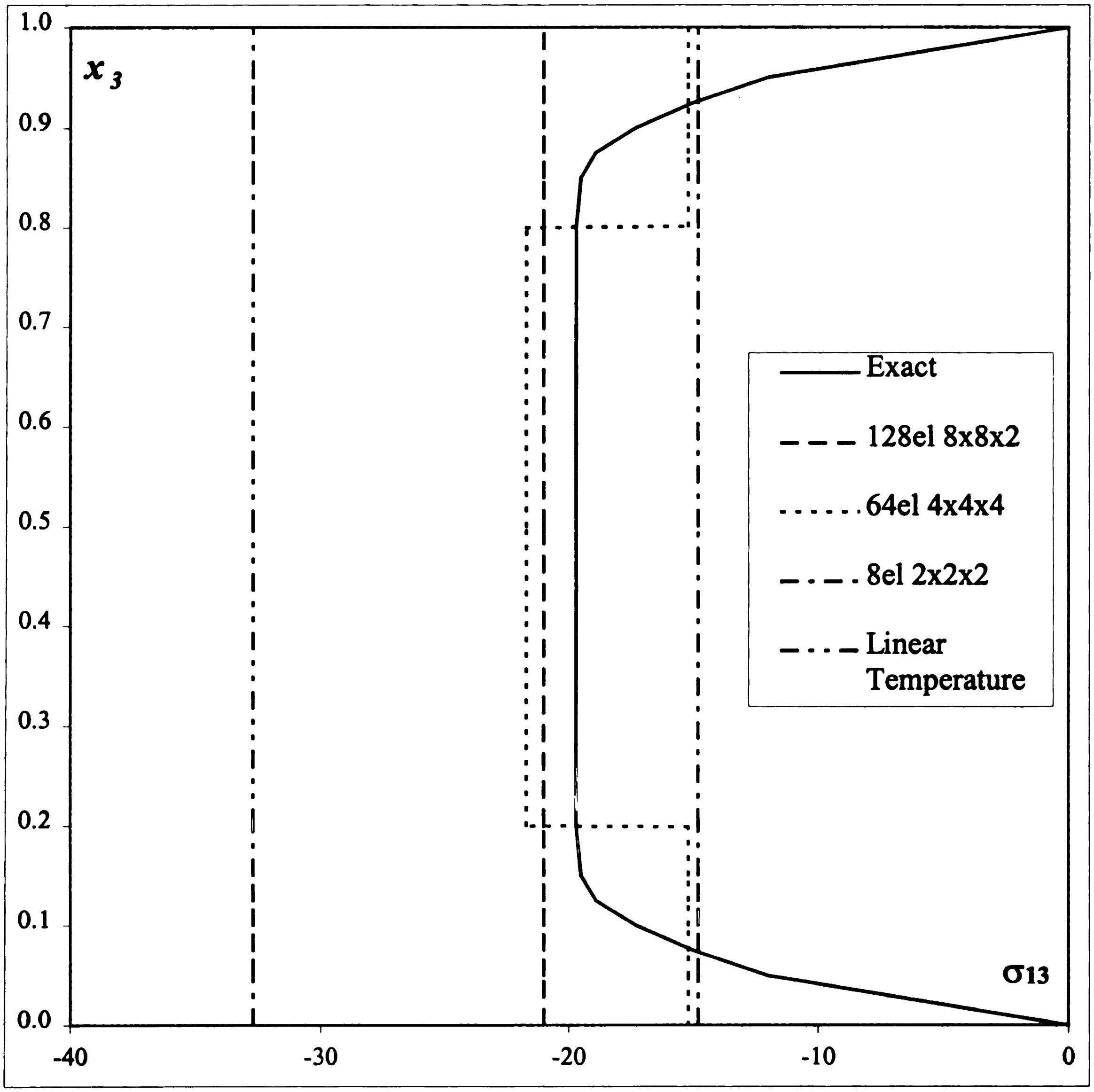


Figure 49 ***Through-thickness variation of the stress σ_{13} at $x_1 = 0$ and $x_2 = a/2$ in the sandwich plate with aspect ratio 4.***

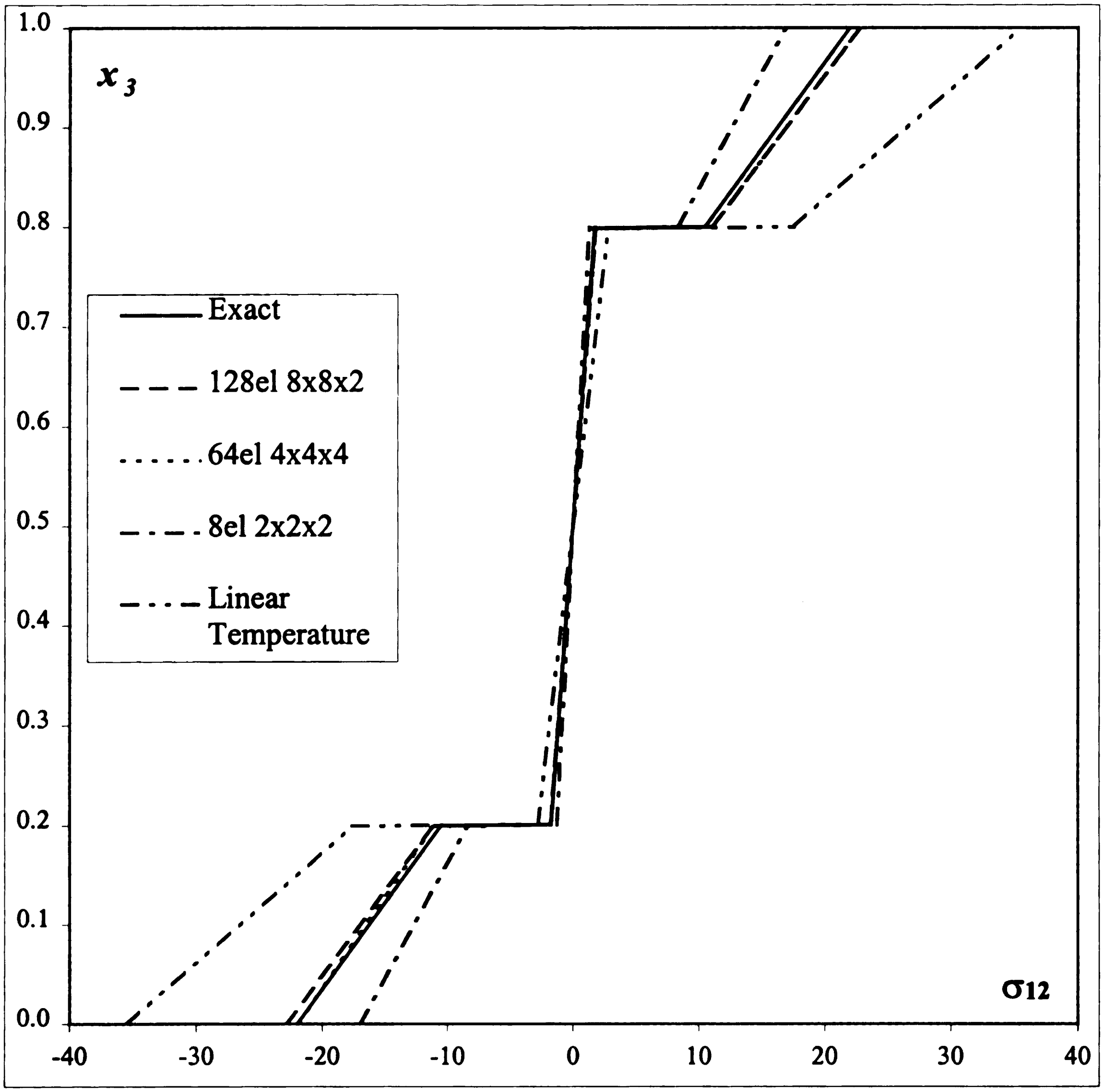


Figure 50 ***Through-thickness variation of the stress σ_{12} at $x_1 = 0$ and $x_2 = 0$ in the sandwich plate with aspect ratio 4.***

5.2.2 Four-Layered Antisymmetric Cross-Ply Laminate

In this case the 4 layers have equal thickness, $0.25h$, where h is the thickness of the plate. The physical properties of the material are those typical of high-modulus fibrous composites:

$$\begin{aligned}&\frac{E_L}{E_T}=15,\ \ \frac{G_{LT}}{E_T}=0.5,\ \ \frac{G_{TT}}{E_T}=0.3378,\ \ \nu_{LT}=0.3,\ \ \nu_{TT}=0.48,\\&\alpha_L=0.139\times10^{-6},\ \ \alpha_T=9\times10^{-6}\end{aligned}\tag{65}$$

Where subscripts L and T stand for the directions parallel and perpendicular to the fibers. The fibers of the top layer are parallel to the x_1 axis. Results are presented for an aspect ratio of 10.

For this problem the exact solution is presented in [13,14] only for the transverse shear stresses σ_{23} and σ_{13}. However, even though the others stresses and the inplane displacements cannot be compared to an exact solution, they are reported here. All displacements and stresses are evaluated at the same positions in the plate as for the previous problem and presented in Figures 51-57.

The average values of the transverse shear stresses are well captured by the two finer meshes, whereas the coarse 2x2x2 mesh produces slightly underestimated predictions.

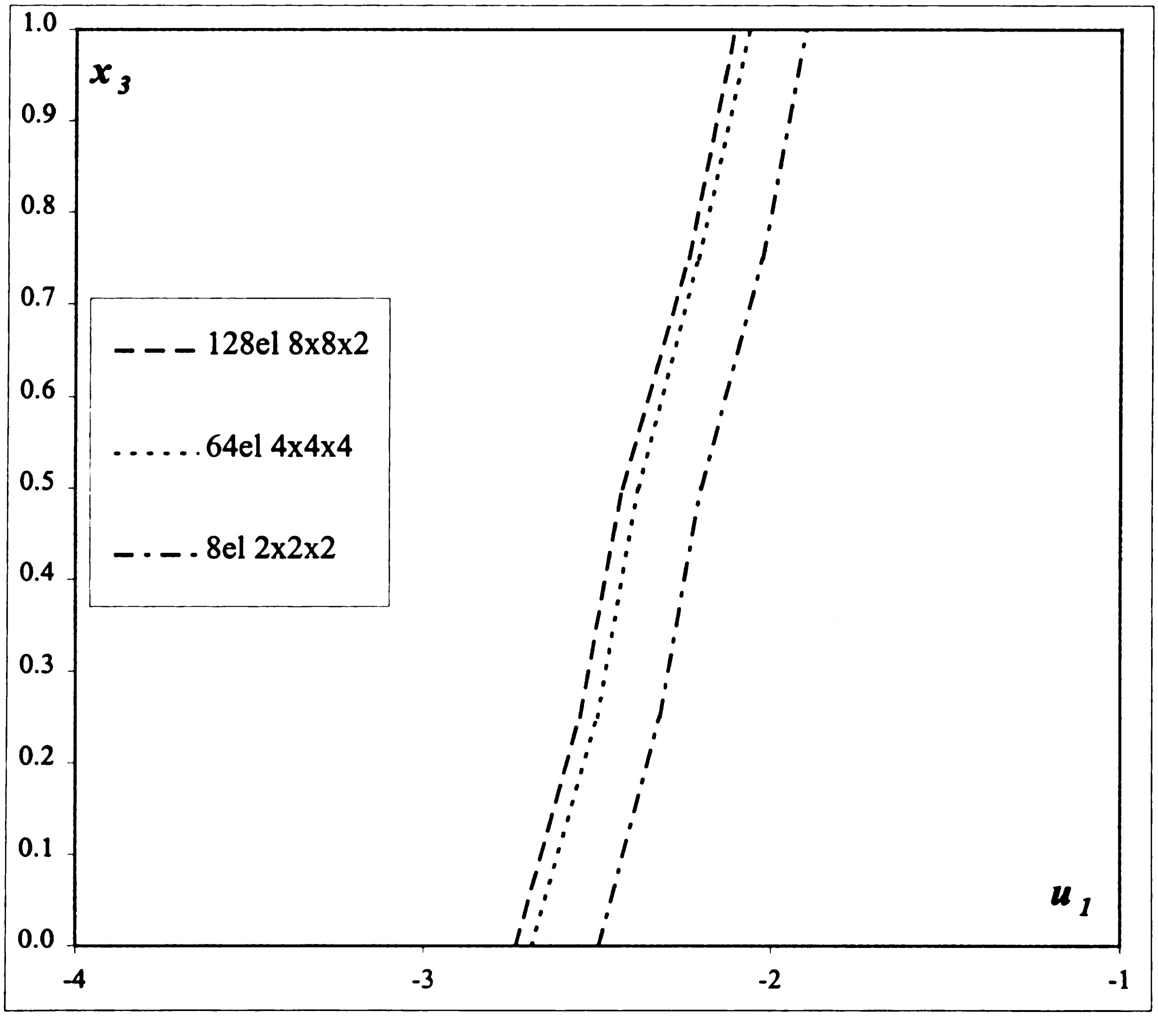


Figure 51 ***Through-thickness variation of the displacement u_1 at $x_1 = 0$ and $x_2 = a/2$ in the 4-layer antisymmetric cross-ply laminate.***

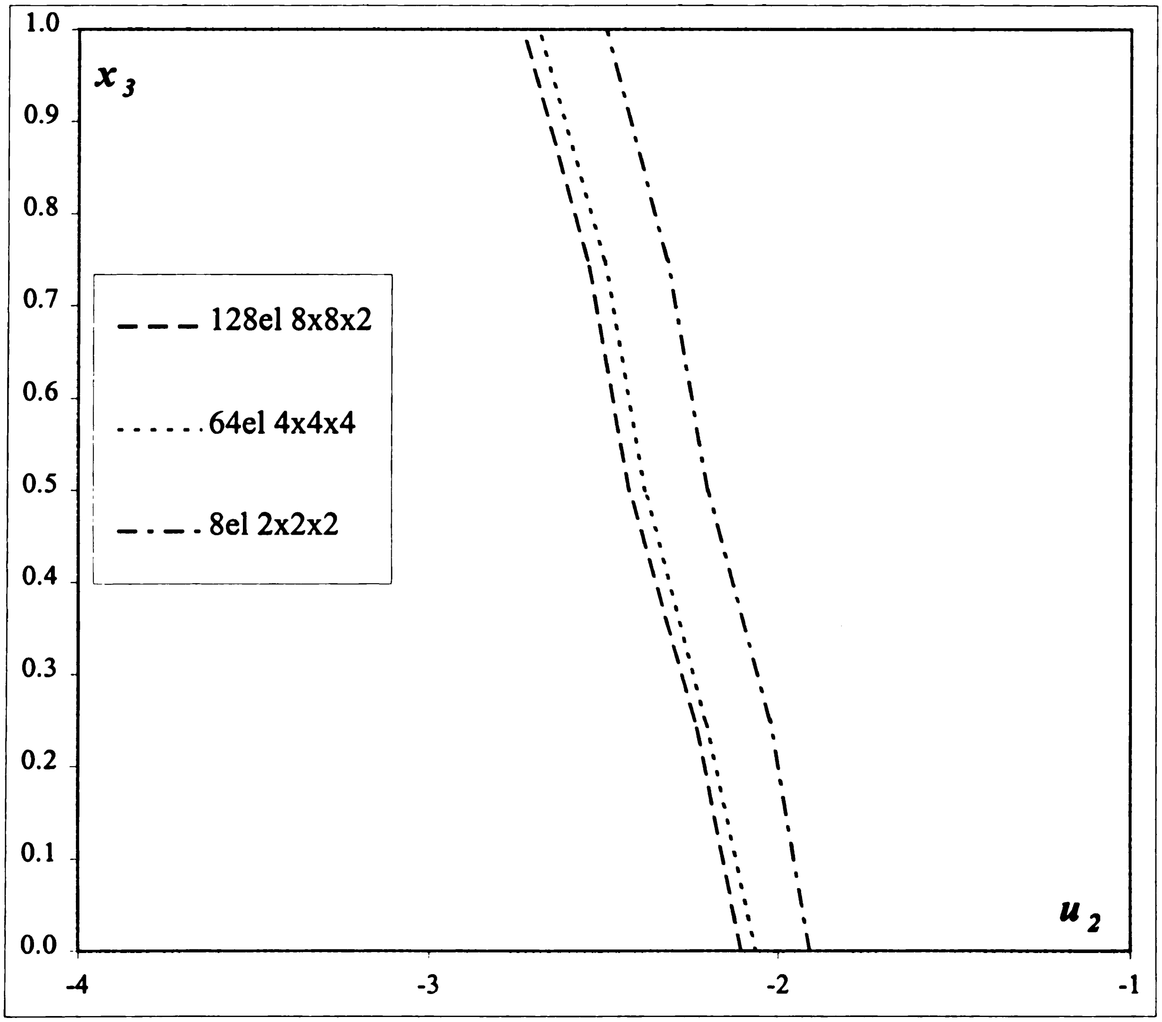


Figure 52 ***Through-thickness variation of the displacement u_2 at $x_1 = a/2$ and $x_2 = 0$ in the 4-layer antisymmetric cross-ply laminate.***

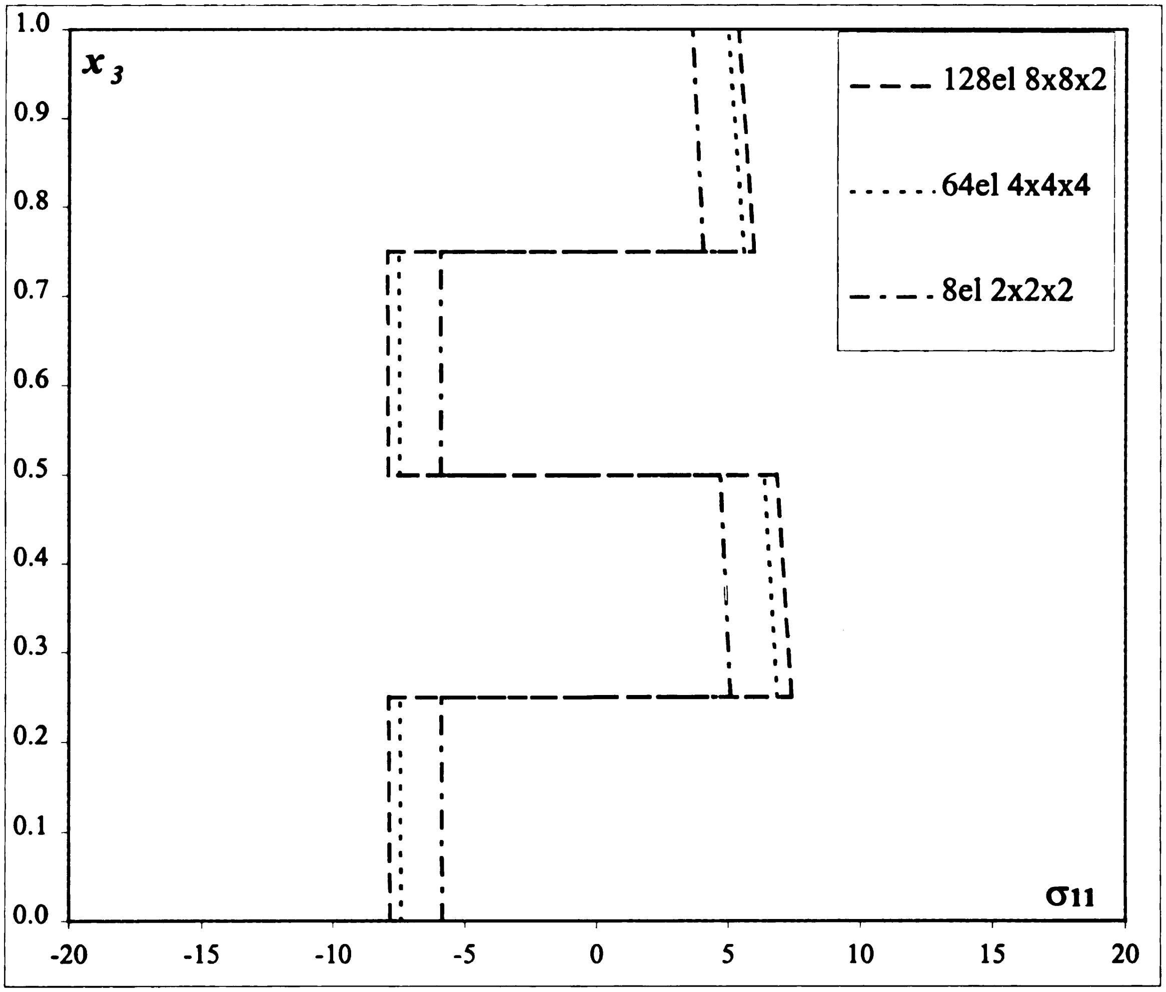


Figure 53 ***Through-thickness variation of the stress σ_{11} at $x_1 = a/2$ and $x_2 = a/2$ in the 4-layer antisymmetric cross-ply laminate.***

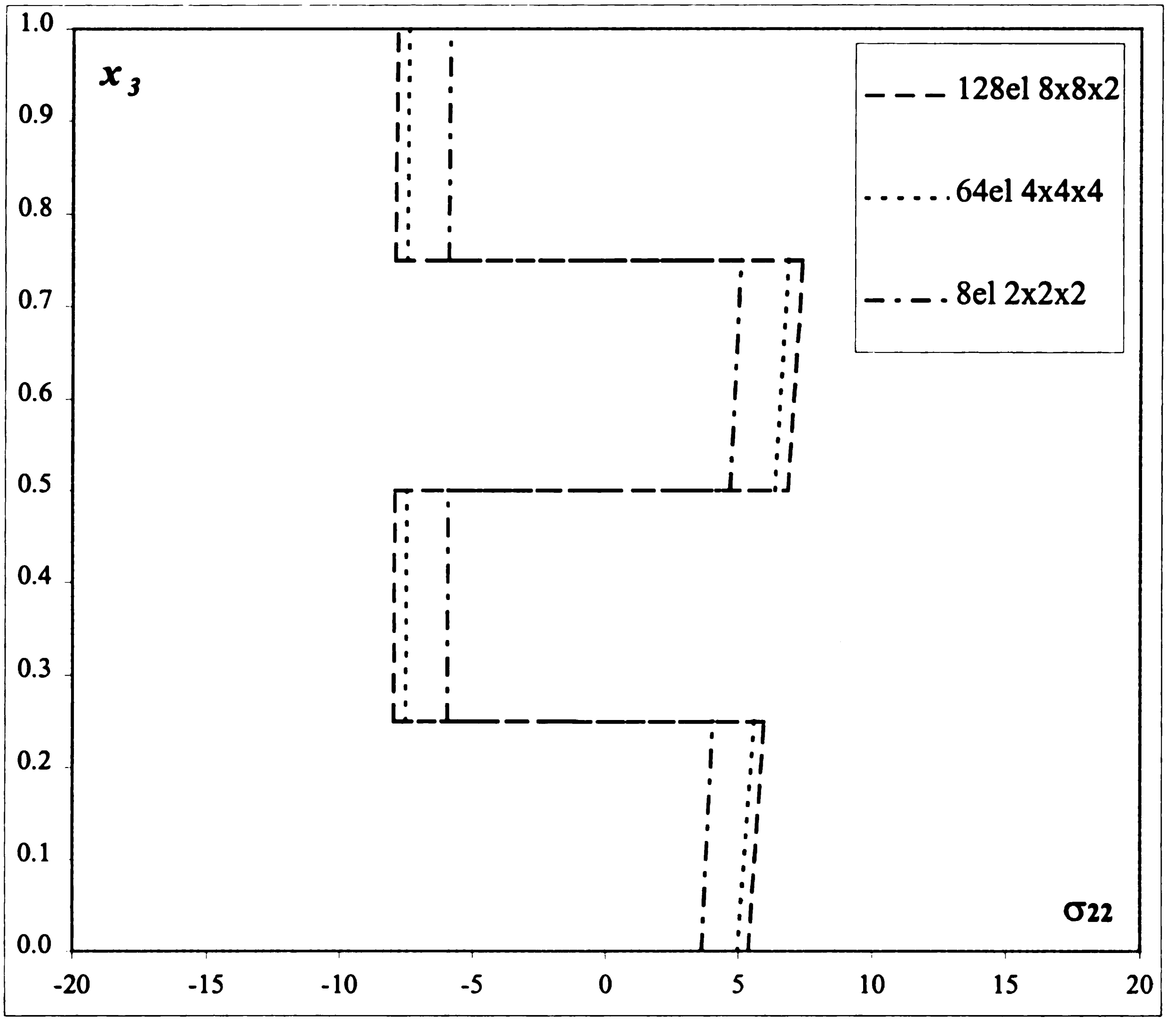


Figure 54 ***Through-thickness variation of the stress σ_{22} at $x_1 = a/2$ and $x_2 = a/2$ in the 4-layer antisymmetric cross-ply laminate.***

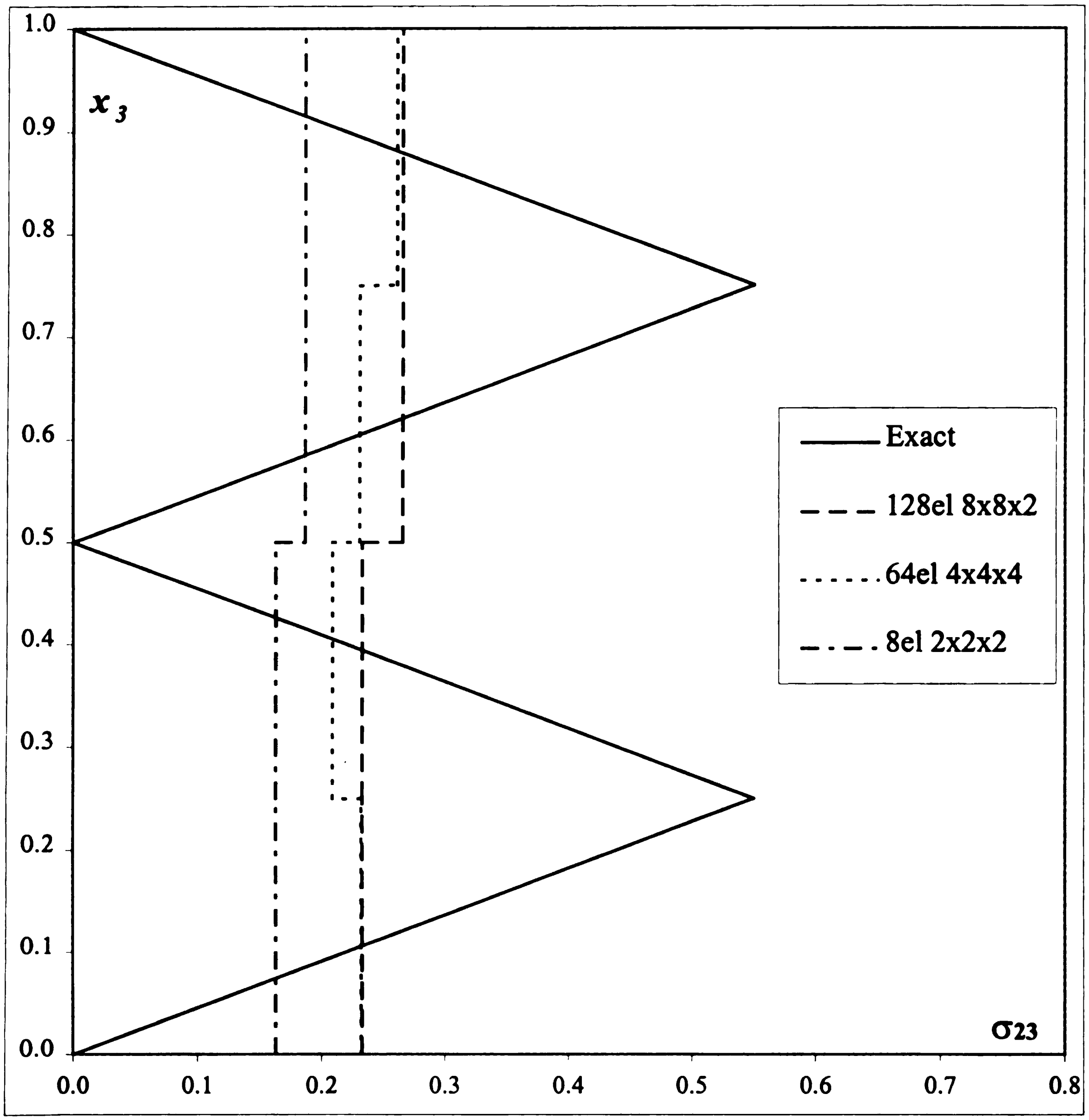


Figure 55 ***Through-thickness variation of the stress σ_{23} at $x_1 = a/2$ and $x_2 = 0$ in the 4-layer antisymmetric cross-ply laminate.***

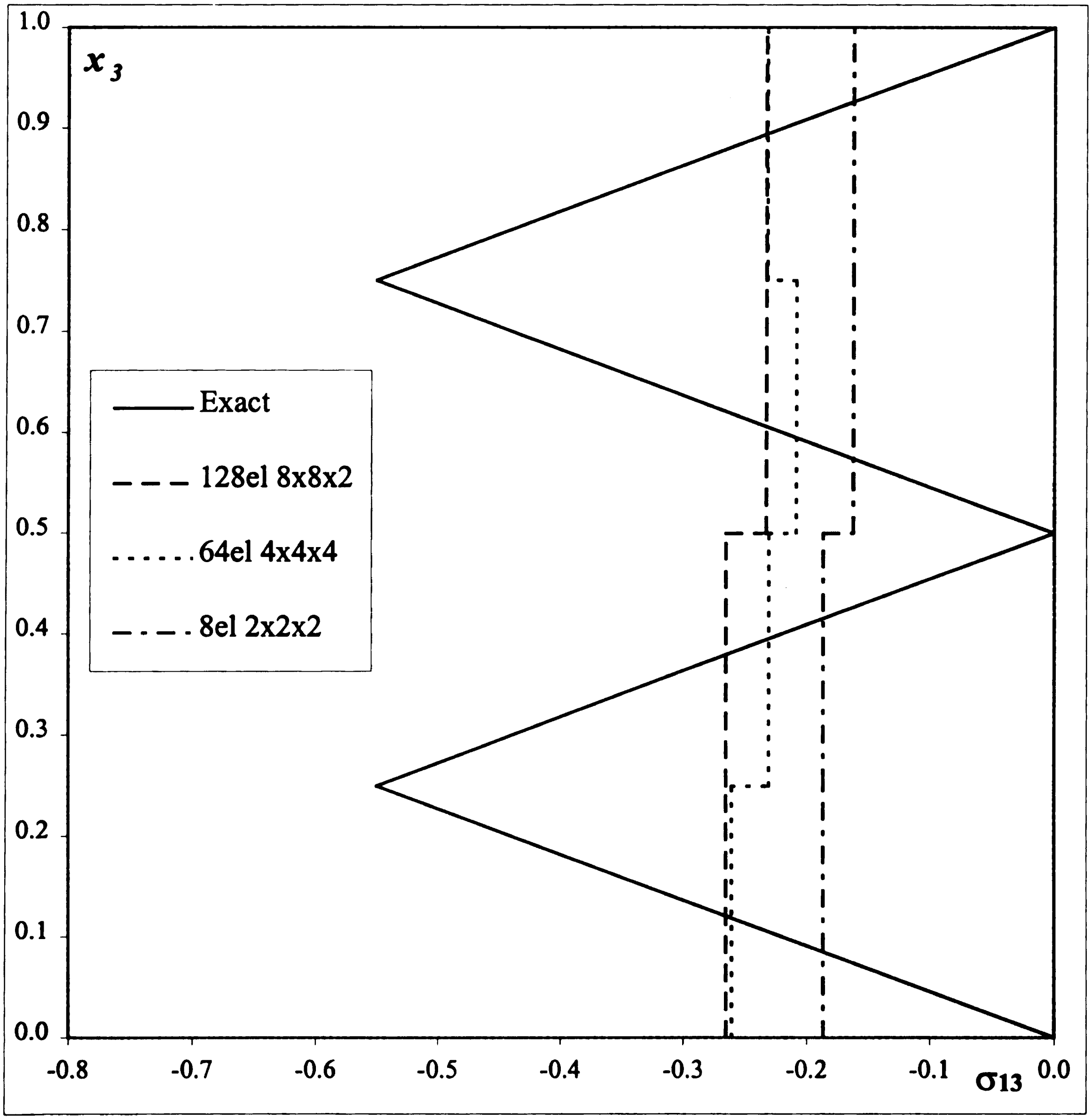


Figure 56 ***Through-thickness variation of the stress σ_{13} at $x_1 = 0$ and $x_2 = a/2$ in the 4-layer antisymmetric cross-ply laminate.***

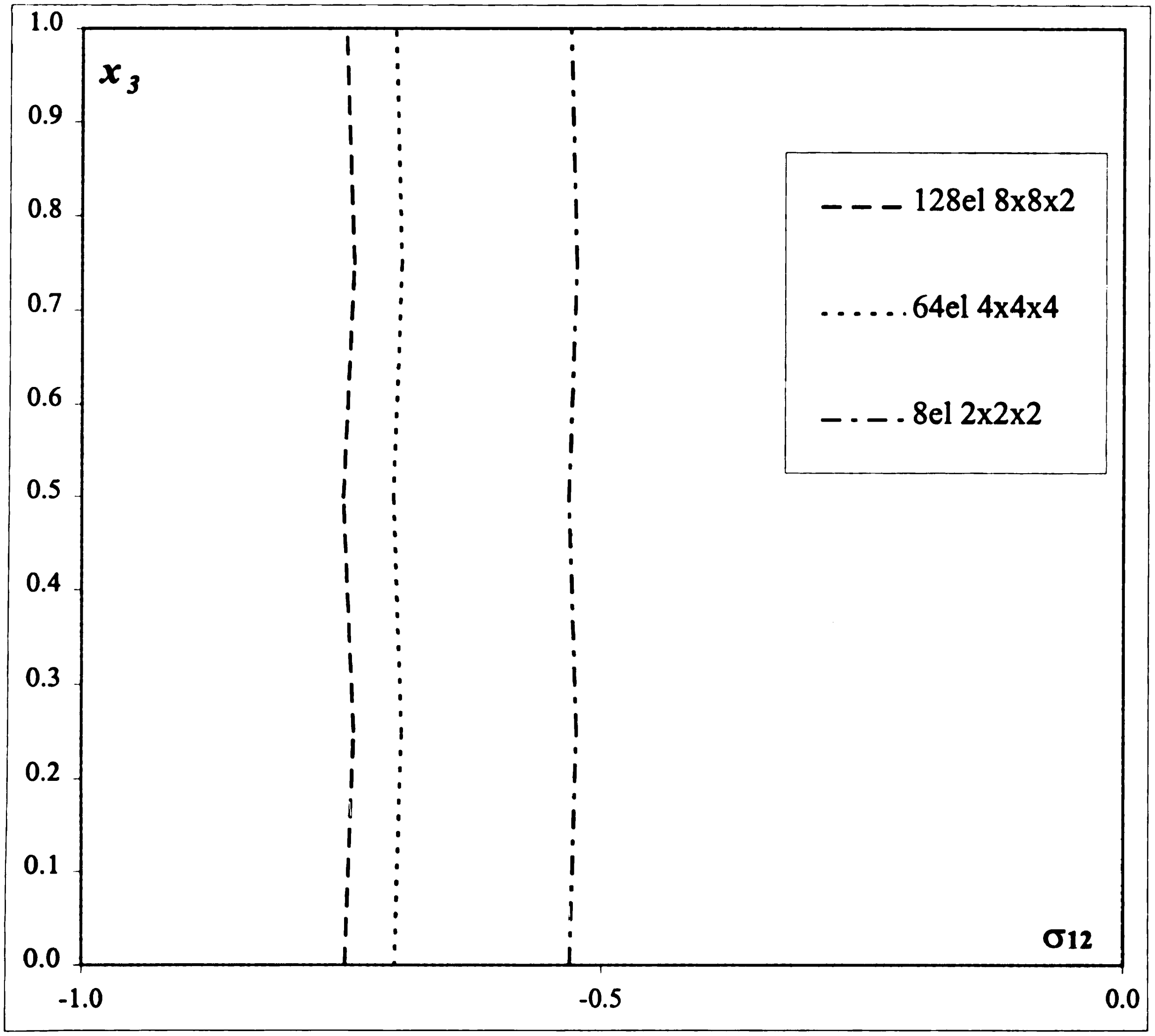


Figure 57 ***Through-thickness variation of the stress σ_{12} at $x_1 = 0$ and $x_2 = 0$ in the 4-layer antisymmetric cross-ply laminate.***

CHAPTER 6 CONCLUSIONS

An improved thermal lamination theory has been presented that is both accurate and effective for predicting the distribution of temperature in laminated structures wherein the plies have dissimilar thermal properties. The theory lends itself well to several forms of finite element models, including two-dimensional shell-type models and three-dimensional solid-type models. In the latter case, a sublaminate form of the theory is utilized that allows the thickness of a laminate to be discretized, if necessary, to increase solution accuracy. The four-noded two-dimensional sublaminate finite element model used to test the thermal theory behaved exceptionally well even in the presence of material non-linearity, wherein thermal conductivity coefficients of the individual layers are defined as quadratic functions of the temperature.

Starting from the proposed thermal lamination theory and a zig-zag sublaminate plate theory [31] that has been previously proven to be very effective, accurate thermal stress analysis of laminated structures can be performed. With respect to others theories, the novel features of the formulation allow superior results in predicting the distribution of temperatures, displacements and stresses in laminated plates wherein the plies have dissimilar thermal and/or structural properties. The three-dimensional solid-type models permit a sublaminate form of the theory in order to increase solution accuracy by refining the mesh through the thickness when needed. However, the accuracy of the first-order zig-zag sublaminate kinematics approximation minimizes the need for multiple sublaminates for many problems of practical interest. The finite element model is "regenerated" in the form of an eight-noded three-dimensional element with one DOF (temperature) per node for the thermal analysis and five (three translations and two rotations) for the structural one. Thus, it is suitable for implementation into commercial finite element codes. Numerical results demonstrate that the current model is accurate, efficient, and robust for analysis of a wide variety of thick or thin laminated plates and sandwich panels.

The presented theory could be applied to other fields of engineering. For example, models for moisture absorption would take the same form as above, with temperature being replaced by moisture content. For the case of electric fields, Maxwell's equations governing electric fields are similar in form to those of heat conduction, and so a model essentially equivalent to that given above could readily be developed in terms of the electric scalar potential.

APPENDICES

APPENDIX A FORTRAN CODE, THERM2D

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C                              THERM2D                                 C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C                             I N P U T                                C
C                                                                      C
C     >  TITLE                                                         C
C     >  PRTSTF                                                        C
C     >  DOM_X, DOM_Z, NELM_X, NELM_Z                                  C
C                                                                      C
C     IF (IMESH = 0) THEN                                              C
C        DO LOOP: I = 1 TO NELMSH                                      C
C        >  NODE(I,J), J = 1 TO NODELM                                 C
C        END OF LOOP                                                   C
C        >  X(I), Z(I)  I = 1 TO NODMSH                                C
C     ELSE                                                             C
C        DO LOOP: I = 1 TO NELM_X                                      C
C        >  DX(I)                                                      C
C        END OF LOOP                                                   C
C        DO LOOP: I = 1 TO NELM_Z                                      C
C        >  DZ(I)                                                      C
C        END OF LOOP                                                   C
C     END IF                                                           C
C                                                                      C
C     >  NSGD    NSGF                                                  C
C     >  ISGD(I,J), J = 1 TO 2, I = 1 TO NSGD                          C
C     >  VSGD(I), I = 1 TO NSGD                                        C
C     >  ISGF(I,J), J = 1 TO 2, I = 1 TO NSGF                          C
C     >  VSGF(I), I = 1 TO NSGF                                        C
C                                                                      C
C      DO LOOP I=1, NELMSH                                             C
C         NLAYER(I)                                                    C
C         DO LOOP J=1, NLAYER(I)                                       C
C         KX(I,J), KZ(I,J), Z1LAY(I,J), Z2LAY(I,J)                     C
C        END OF LOOP                                                           C
C      END OF LOOP                                                     C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C                         V A R I A B L E S                            C
C                                                                      C
C     PARAMETERS:                                                      C
C                                                                      C
C        IUNIT  - INTEGER, UNIT NUMBER OF INPUT FILE.                  C
C        OUNIT  - INTEGER, UNIT NUMBER OF OUTPUT FILE.                 C
C                                                                      C
C        BNDMAX - INTEGER, COLUMN DIMENSION OF GLOBAL STIFFNESS:       C
C                 GSTIF IS BANDED IF IMASS = 0,                        C
C                 GSTIF IS SQUARE IF IMASS = 1.                        C
C        EDFMAX - INTEGER, MAXIMUM NUMBER OF D.O.F. PER ELEMENT.       C
C        NBCMAX - INTEGER, MAXIMUM NUMBER OF BOUNDARY CONDITIONS.      C
C        NDFMAX - INTEGER, MAXIMUM NUMBER OF D.O.F. PER NODE.          C
C        NEMMAX - INTEGER, MAXIMUM NUMBER OF ELEMENTS IN THE MESH.     C
C        NEQMAX - INTEGER, MAXIMUM NUMBER OF EQUATIONS (OR MAXIMUM     C
C                 NUMBER OF D.O.F. IN THE MESH).                       C
C        NGPMAX - INTEGER, MAXIMUM NUMBER OF GAUSS POINTS.             C
C        NNMMAX - INTEGER, MAXIMUM NUMBER OF NODES IN THE MESH.        C
C        NPEMAX - INTEGER, MAXIMUM NUMBER OF NODES PER ELEMENT.        C
C        NLAMAX - INTEGER, MAXIMUM NUMBER OF LAYERS PER ELEMENT.       C
C                                                                      C
C     CONTROL VARIABLES:                                               C
```

```
C                                                                        C
C        PRTGRD - INTEGER, FLAG FOR PRINTING GRADIENTS:                  C
C                 PRTGRD = 0 FOR NO PRINT,                               C
C                 PRTGRD = 1 FOR PRINTING.                               C
C        PRTSTF - INTEGER, FLAG FOR PRINTING STIFFNESS MATRICES:         C
C                 PRTSTF = 0 FOR NO PRINT,                               C
C                 PRTSTF = 1 FOR PRINTING STIFFNESS OF ELEMENT ONE,      C
C                 PRTSTF = 2 FOR PRINTING GLOBAL STIFFNESS.              C
C                                                                        C
C     ELEMENT PROPERTIES:                                                C
C                                                                        C
C        NDFELM - INTEGER, NUMBER OF D.O.F. PER ELEMENT.                 C
C        NODELM - INTEGER, NUMBER OF NODES PER ELEMENT.                  C
C        NLAYER - INTEGER, NUMBER OF LAYERS PER ELEMENT.                 C
C                                                                        C
C     MESH:                                                              C
C                                                                        C
C        BNDWTH - INTEGER, HALF BANDWIDTH OF GLOBAL STIFFNESS MATRIX     C
C                 FOR BENDING PROBLEMS.                                  C
C        EX     - DOUBLE PRECISION(NPEMAX), EX(I) IS THE X-COORDINATE    C
C                 OF THE I-TH NODE OF THE ELEMENT UNDER CONSIDERATION.   C
C        IMESH  - INTEGER, FLAG TO INDICATE HOW MESH WILL BE INPUT:      C
C                 IMESH = 0 IF THE MESH IS INPUT BY HAND,                C
C                 IMESH = 1 IF THE MESH IS COMPUTED BY THE PROGRAM.      C
C        NDFMSH - INTEGER, NUMBER OF D.O.F. IN THE MESH.                 C
C        NDFNOD - INTEGER, NUMBER OF D.O.F. PER NODE.                    C
C        NELMSH - INTEGER, NUMBER OF ELEMENTS IN THE MESH.               C
C        NODE   - INTEGER(NEMMAX,NPEMAX), CONNECTIVITY MATRIX:           C
C                 NODE(I,J) IS THE J-TH NODE OF THE I-TH ELEMENT.        C
C        NODMSH - INTEGER, NUMBER OF NODES IN THE MESH.                  C
C        X      - DOUBLE PRECISION(NNMMAX), X(I) IS THE GLOBAL           C
C                 COORDINATE OF THE I-TH NODE.                           C
C        NELM_X - INTEGER, NUMBER OF ELEMENTS IN THE MESH IN X DIR.      C
C        NELM_Z - INTEGER, NUMBER OF ELEMENTS IN THE MESH IN Z DIR.      C
C        DOM_X  - DOUBLE PRECISION, DOMAIN IN X DIRECTION                C
C        DOM_Z  - DOUBLE PRECISION, DOMAIN IN Z DIRECTION                C
C        DX     - DOUBLE PRECISION, DIMENSION OF ELEMENTS                C
C                 IN X DIRECTION STARTING FROM THE LEFT                  C
C        DZ     - DOUBLE PRECISION, DIMENSION OF ELEMENTS                C
C                 IN Z DIRECTION STARTING FROM THE BOTTOM                C
C        KX     - DOUBLE PRECISION, TERMAL CONDUCTIVITY COEFFICIENT OF C
C                 THE LAYER ALONG X DIRECTION                            C
C        KZ     - DOUBLE PRECISION, TERMAL CONDUCTIVITY COEFFICIENT OF C
C                 THE LAYER ALONG Z DIRECTION                            C
C        Z1LAY  - DOUBLE PRECISION, Z COORDINATE OF THE BOTTOM OF THE  C
C                 LAYER                                                  C
C        Z2LAY  - DOUBLE PRECISION, Z COORDINATE OF THE TOP OF THE       C
C                 LAYER                                                  C
C                                                                        C
C     BOUNDARY CONDITIONS:                                               C
C                                                                        C
C        ISGD   - INTEGER(NBCMAX,2), ISGD(I,J) INDICATES THAT THE J-TH C
C                 DISPLACEMENT D.O.F. OF THE I-TH NODE IS SPECIFIED.   C
C        ISGF   - INTEGER(NBCMAX,2), ISGF(I,J) INDICATES THAT THE J-TH C
C                 FORCE COMPONENT OF THE I-TH NODE IS SPECIFIED.         C
C        NSGD   - INTEGER, NUMBER OF DISPLACEMENT BOUNDARY CONDITIONS. C
C        NSGF   - INTEGER, NUMBER OF NODAL FORCE BOUNDARY CONDITIONS.  C
C        VSGD   - DOUBLE PRECISION(NBCMAX), VSGD(I) IS THE SPECIFIED   C
C                 VALUE OF THE I-TH DISPLACEMENT BOUNDARY CONDITION.   C
C        VSGF   - DOUBLE PRECISION(NBCMAX), VSGF(I) IS THE SPECIFIED   C
C                 VALUE OF THE I-TH NODAL FORCE BOUNDARY CONDITION.    C
C                                                                        C
C     FINITE ELEMENT MODEL:                                              C
```

```
C                                                                          C
C       DSF    - DOUBLE PRECISION(NPEMAX), DSF(I) IS THE LOCAL (XI)        C
C                DERIVATIVE OF THE I-TH SHAPE FUNCTION.                    C
C       EF     - DOUBLE PRECISION(EDFMAX), ELEMENT FORCE VECTOR.           C
C       ESTIF  - DOUBLE PRECISION(EDFMAX,EDFMAX), ELEMENT STIFFNESS.       C
C       EU     - DOUBLE PRECISION(EDFMAX), ELEMENT EXTENSION VECTOR.       C
C       GAUSPT - DOUBLE PRECISION(NGPMAX,NGPMAX) GAUSPT(I,J) IS THE        C
C                I-TH GAUSS POINT OF J-TH GAUSS RULE.                      C
C       GAUSWT - DOUBLE PRECISION(NGPMAX,NGPMAX) GAUSWT(I,J) IS THE        C
C                I-TH GAUSS WEIGHT OF J-TH GAUSS RULE.                     C
C       GDSF   - DOUBLE PRECISION(NPEMAX), GDSF(I) IS THE GLOBAL (X)       C
C                DERIVATIVE OF THE I-TH SHAPE FUNCTION.                    C
C       GF     - DOUBLE PRECISION(NEQMAX), GLOBAL FORCE VECTOR,            C
C                WHICH, ON RETURN, CONTAINS THE SOLUTION VECTOR;           C
C                ALSO USED AS A DUMMY VECTOR IN EIGENPROBLEMS.             C
C       GSTIF  - DOUBLE PRECISION(NEQMAX,BNDMAX), GLOBAL STIFFNESS.        C
C       NGPSTF - INTEGER, NUMBER OF GAUSS POINTS TO BE USED TO             C
C                INTEGRATE THE STIFFNESS MATRIX.                           C
C       NGPGRD - INTEGER, NUMBER OF GAUSS POINTS AT WHICH THE              C
C                GRADIENTS ARE TO BE EVALUATED.                            C
C       SF     - INTEGER(NPEMAX), SF(I) IS THE I-TH SHAPE FUNCTION.        C
C                                                                          C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      IMPLICIT LOGICAL (A-Z)

      INTEGER            IUNIT , OUNIT , NEMMAX, NPEMAX, NDFMAX, NGPMAX,
     .                   NNMMAX, NBCMAX, NEQMAX, EDFMAX, BNDMAX,
     .                       NELM_X, NELM_Z, NLAMAX

      PARAMETER         (IUNIT  = 10, OUNIT  = 12, NEMMAX = 400,
     .                   NPEMAX =  4, NDFMAX =  1, NGPMAX =  5,
     .                   BNDMAX = 400, NLAMAX = 40,
     .                   EDFMAX = NDFMAX * NPEMAX,
     .                   NNMMAX = NEMMAX * (NPEMAX - 1) + 1,
     .                   NEQMAX = NNMMAX * NDFMAX,
     .                   NBCMAX = NEMMAX*4)



      INTEGER            PRTSTF, PRTGRD, nodelm, ndfelm, ngpstf, ngpgrd,
     .                   nelmsh, ndfmsh, nodmsh, ndfnod, node  , bndwth,
     .                   nsgd  , nsgf  , isgd  , isgf,
     .                   NLAYER

      DOUBLE PRECISION   X     ,   vsgd, vsgf  , ex    , DOM_X, DOM_Z,
     .                   GAUSPT, GAUSWT, ESTIF , EF    , GSTIF ,
     .                   GF    , DSF   , GDSF  ,
     .                    Z     , KX    , KZ,    Z1LAY,    Z2LAY, EZ,
     .                    AK    , FI_D  , D_FI_N, D_FI,   SF2, GUL


      DIMENSION  NODE(NEMMAX,NPEMAX), ISGD(NBCMAX,2),
     .           ISGF(NBCMAX,2), X(NNMMAX)          , EX(NPEMAX)    ,
     .           VSGD(NBCMAX)   , VSGF(NBCMAX)      ,
     .           EF(EDFMAX)     , GF(NEQMAX)        ,
     .           DSF(NPEMAX)         , GDSF(NPEMAX)  ,
     .           GAUSPT(NGPMAX,NGPMAX), GAUSWT(NGPMAX,NGPMAX),
     .           ESTIF(EDFMAX,EDFMAX) ,
     .           GSTIF(NEQMAX,BNDMAX),
     .           Z(NNMMAX), NLAYER(NEMMAX),
     .           KX(nemmax,NLAMAX), KZ(nemmax,NLAMAX),
```

```
     .           Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .           EZ(NPEMAX), AK(nemmax,NLAMAX), FI_D(nemmax),
     .           D_FI_N(nemmax,NLAMAX), D_FI(nemmax,NLAMAX),
      .           SF2(2,2),GUL(nemmax,2,NLAMAX)


       open(unit=10,file='TERM2D.data',status='old')
      open(unit=12,file='TERM2D.out', status='unknown')

C --------------------
C --- PREPROCESSOR ---
C --------------------

C --- INPUT OPTIONS.

      CALL INPOPT (IUNIT , OUNIT , PRTSTF, PRTGRD)

C --- INPUT ELEMENT DATA.

      call INPELM (OUNIT , NODELM, NDFELM, NGPSTF, NGPGRD,
     .             NDFNOD)

C --- INPUT MESH PARAMETERS AND FORM THE MESH.

      call INPMSH (IUNIT , OUNIT , NEMMAX, NPEMAX, NNMMAX, NODELM,
     .             NDFNOD, DOM_X, DOM_Z, NELM_X, NELM_Z,
     .             NELMSH, NDFMSH, NODMSH, NODE, BNDWTH, X, Z )

C --- INPUT NODAL BOUNDARY CONDITIONS.

      call INPNBC (IUNIT , OUNIT , NBCMAX, NNMMAX, NODMSH, NDFNOD,
     .             NSGD  , NSGF  , ISGD  , ISGF  , X    , Z, VSGD,
     .             VSGF  )

C --- INPUT ELEMENT LAYERS

      call INPLAY (IUNIT , OUNIT , NELMSH, NLAYER, NLAMAX, NEMMAX,
      .              KX    , KZ    , Z1LAY , Z2LAY)


C --- INITIALIZE THE ARRAYS WHICH CONTAIN THE GAUSS POINTS AND WEIGHTS.

      CALL GQUAD  (NGPMAX, GAUSPT, GAUSWT)

C -----------------
C --- PROCESSOR ---
C -----------------

      call Procsr_T     (OUNIT , NEMMAX, NPEMAX, NGPMAX, NNMMAX, NEQMAX,
     .                   BNDMAX, NBCMAX, EDFMAX, NDFNOD, NDFELM, NODELM,
     .                   NDFMSH, NELMSH, NODE  , NGPSTF, PRTSTF, BNDWTH,
     .                   NSGD  , ISGD  , NSGF  , ISGF  , X     , EX    ,
     .                   GAUSPT, GAUSWT, DSF   , GDSF  ,
     .                   ESTIF , EF    , GSTIF , GF    , VSGD  , VSGF  ,
```

```
     .                   Z      , EZ , KX    , KZ,    Z1LAY,     Z2LAY,
     .                   NLAMAX, NLAYER, AK , FI_D, D_FI_N, D_FI, SF2 )


C --------------------
C --- POSTPROCESSOR ---
C --------------------


      call PRTGU  (OUNIT , NEQMAX, NNMMAX, NDFNOD, NODMSH, GF,
     .             X, Z  )


      call  PR_L_T (OUNIT, NPEMAX,NLAMAX,NELMSH, NEMMAX,
     .              NLAYER, NEQMAX, Z1LAY, Z2LAY,
     .              AK, FI_D, NODE, GUL, GF)


      STOP
      END


C--------------------------------------------------------------------C
C                                                                    C
C                           SUBROUTINES                              C
C                                                                    C
C--------------------------------------------------------------------C



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
C     SUBROUTINE:  ASMBLE                                            C
C     PURPOSE:     ASSEMBLE THE GLOBAL MATRICES IN BANDED FORM.      C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE ASMBLE (NEQMAX, BNDMAX, EDFMAX, NEMMAX, NPEMAX, NODELM,
     .                   NDFNOD, NODE  , ESTIF , EF    , GSTIF , GF    ,
     .                   ELMNUM)

      IMPLICIT LOGICAL (A-Z)

      INTEGER            NEQMAX, BNDMAX, EDFMAX, NEMMAX, NPEMAX, NODELM,
     .                   NDFNOD, NODE  , ELMNUM, INOD  , JNOD  , IDOF  ,
     .                   JDOF  , ROW   , COL   , COL1  , EROW  , ECOL

      DOUBLE PRECISION   ESTIF , EF    , GSTIF , GF

      DIMENSION NODE(NEMMAX,NPEMAX) , ESTIF(EDFMAX,EDFMAX), EF(EDFMAX),
     .          GSTIF(NEQMAX,BNDMAX), GF(NEQMAX)


      DO 40 INOD = 1, NODELM
         ROW = NDFNOD * (NODE(ELMNUM,INOD) - 1)

         DO 30 IDOF = 1, NDFNOD
            ROW  = ROW + 1
            EROW = (INOD - 1) * NDFNOD + IDOF
            GF(ROW) = GF(ROW) + EF(EROW)
```

```
         DO 20 JNOD = 1, NODELM
            COL1 = NDFNOD * (NODE(ELMNUM,JNOD) - 1)

            DO 10 JDOF = 1, NDFNOD
               COL  = COL1 - ROW + JDOF + 1
               ECOL = (JNOD - 1) * NDFNOD + JDOF
               IF (COL .GT. 0) THEN
                  GSTIF(ROW,COL) = GSTIF(ROW,COL) + ESTIF(EROW,ECOL)
               END IF
10          CONTINUE
20       CONTINUE
30    CONTINUE
40 CONTINUE
   RETURN
   END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     SUBROUTINE:  CR_FI                                              C
C     PURPOSE:     EVALUATE FUNCTIONS FI AND ITS DERIVATIVES          C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE CR_FI(NLAMAX, NELMSH, NEMMAX,
     .                 NLAYER, KZ, Z1LAY, Z2LAY,
     .                 AK, FI_D, D_FI_N, D_FI)


      IMPLICIT LOGICAL (A-Z)

      integer              NLAMAX, NELMSH,NEMMAX,
     .                     NLAYER, I, K, L , N

      double precision     KZ, Z1LAY, Z2LAY, AK, FI_D,
     .                     H, R, D_FI_N, D_FI, SUM_AK


      DIMENSION  NLAYER(NEMMAX), KZ(nemmax,NLAMAX),
     .           Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .             AK(nemmax,NLAMAX), FI_D(nemmax), D_FI_N(nemmax,NLAMAX),
     .             D_FI(nemmax,NLAMAX)


      DO 100 I = 1, NELMSH

      if (NLAYER(I) .gt. 1) then
        SUM_AK = 0
        AK(I,1) = ( KZ(I,1)/KZ(I,2) - 1 )
        SUM_AK = AK(I,1)

        DO 30 K = 2,  (NLAYER(I)-1)
        AK(I,K) = ( KZ(I,K)/KZ(I,K+1) - 1 )*( 1 + SUM_AK)
        SUM_AK  = SUM_AK + AK(I,K)

30      CONTINUE

       END IF


       H = ( Z2LAY(I,NLAYER(I)) - Z1LAY(I,1) )
```

```
      FI_D(I) = H

         DO 40 R = 1, (NLAYER(I)-1)
         FI_D(I) = FI_D(I) + ( H - (Z2LAY(I,R)- Z1LAY(I,1)) )*AK(I,R)
 40     CONTINUE

       D_FI_N(I,1) = 1

         DO 50 L = 2, NLAYER(I)
         D_FI_N(I,L) = D_FI_N(I,L-1) +  AK(I,L-1)
 50      CONTINUE

         DO 60  N = 1, NLAYER(I)
         D_FI(I,N) = D_FI_N(I,N) /  FI_D(I)
 60      CONTINUE


100   CONTINUE

      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  procsr_T                                            C
C     PURPOSE:     DRIVER ROUTINE TO SOLVE THE equations.              C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE procsr_T (OUNIT , NEMMAX, NPEMAX, NGPMAX, NNMMAX,
     .                        NEQMAX,
     .                   BNDMAX, NBCMAX, EDFMAX, ndfnod, ndfelm, nodelm,
     .                   ndfmsh, nelmsh, node  , ngpstf, prtstf, bndwth,
     .                   nsgd  , isgd  , nsgf  , isgf  , X     , EX,
     .                   GAUSPT, GAUSWT, dsf   , GDSF  , ESTIF,
     .                   EF    , GSTIF , gf    , vsgd  , VSGF  ,
     .                   Z     , EZ, KX    , KZ,   Z1LAY,  Z2LAY,
     .                   NLAMAX, NLAYER,AK, FI_D, D_FI_N, D_FI,
     .                   SF2)

      IMPLICIT LOGICAL (A-Z)

      integer            OUNIT , NEMMAX, NPEMAX, NGPMAX, NNMMAX, NEQMAX,
     .                   BNDMAX, NBCMAX, EDFMAX, ndfnod, ndfelm, nodelm,
     .                   ndfmsh, nelmsh, node  , ngpstf, prtstf, bndwth,
     .                   nsgd  , isgd  , nsgf  , isgf  ,
     .                   ELMNUM, ENOD  , GNOD  , IRES  , I     , J,
     .                   NLAMAX, NLAYER

      double precision   X     , EX    , GAUSPT, GAUSWT, dsf   ,
     .                   GDSF  , ESTIF , EF    , GSTIF , gf    , vsgd  ,
     .                   VSGF  ,
     .                   Z,    EZ, KX    , KZ,   Z1LAY,  Z2LAY,
     .                     AK,   FI_D,  D_FI_N, D_FI, SF2


      DIMENSION ISGD(NBCMAX,2), ISGF(NBCMAX,2),
     .          X(NNMMAX)      , EX(NPEMAX)     , EF(EDFMAX)     ,
     .          DSF(NPEMAX)    , GDSF(NPEMAX)   ,
     .          VSGD(NBCMAX)   , VSGF(NBCMAX)   , GF(NEQMAX)     ,
     .          GAUSPT(NGPMAX,NGPMAX)   , GAUSWT(NGPMAX,NGPMAX),
```

```
     .           ESTIF(EDFMAX,EDFMAX)     ,
     .           GSTIF(NEQMAX,BNDMAX)     , NODE(NEMMAX,NPEMAX),
     .                 Z(NNMMAX)       , EZ(NPEMAX), NLAYER(NEMMAX),
     .           KX(nemmax,NLAMAX), KZ(nemmax,NLAMAX),
     .           Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .             AK(nemmax,NLAMAX), FI_D(nemmax), D_FI_N(nemmax,NLAMAX),
     .             D_FI(nemmax,NLAMAX), SF2(2,2)


C --- INITIALIZE THE GLOBAL STIFFNESS ARRAY AND FORCE VECTOR.

      DO 20 I = 1, NDFMSH
         GF(I) = 0.000
         DO 10 J = 1, BNDWTH
            GSTIF(I,J) = 0.000
   10    CONTINUE
   20 CONTINUE


      CALL  CR_FI(NLAMAX, NELMSH, NEMMAX,
     .            NLAYER, KZ, Z1LAY, Z2LAY,
     .            AK, FI_D, D_FI_N, D_FI)


C --- BEGIN DO-LOOP ON NUMBER OF ELEMENTS TO CALCULATE THE ELEMENT
C --- MATRICES AND TO ASSEMBLE THE GLOBAL MATRICES.

      DO 50 ELMNUM = 1, NELMSH

C --- DETERMINE THE LOCAL (ELEMENT) DATA.

      DO 30 ENOD = 1, NODELM
         GNOD = NODE(ELMNUM,ENOD)
         EX(ENOD) = X(GNOD)
          EZ(ENOD) = Z(GNOD)
   30 CONTINUE


      call STIFF_T(EDFMAX, NPEMAX, NGPMAX, ndfelm,
     .              NGPstf, ELMNUM,
     .              DSF, GDSF, EX, ESTIF, EF, GAUSPT, GAUSWT,
     .             EZ, KX   , KZ,   Z1LAY,    Z2LAY, NEMMAX,
     .             NLAMAX, NLAYER,AK, FI_D, D_FI, SF2)


C --- PRINT STIFFNESS MATRIX AND FORCE VECTOR FOR ELEMENT ONE.

      IF (PRTSTF .EQ. 1 .AND. ELMNUM .EQ. 1) THEN
         WRITE(OUNIT,1000)
         DO 40 I = 1, NDFELM
            WRITE(OUNIT,1010) I
            WRITE(OUNIT,1020) (ESTIF(I,J), J = 1, NDFELM)
   40    CONTINUE
         WRITE(OUNIT,1030)
         WRITE(OUNIT,1020) (EF(I), I = 1, NDFELM)
      END IF
```

```
C --- ASSEMBLE THE GLOBAL MATRICES IN BANDED FORM.

      CALL ASMBLE    (NEQMAX, BNDMAX, EDFMAX, NEMMAX, NPEMAX, NODELM,
     .                NDFNOD, NODE  , ESTIF , EF    , GSTIF , GF    ,
     .                ELMNUM)

C --- END OF LOOP TO COMPUTE GLOBAL STIFFNESS AND FORCE MATRICES.

   50 CONTINUE

C --- IMPOSE SECONDARY BOUNDARY CONDITIONS.

      IF (NSGF .GT. 0) THEN
         CALL FBNDRY    (NEQMAX, NBCMAX, NDFNOD, NSGF  , ISGF  , VSGF  ,
     .                   GF    )
      END IF

C --- PRINT GLOBAL STIFFNESS MATRIX AND FORCE VECTOR.

      IF (PRTSTF .EQ. 2) THEN
         WRITE(OUNIT,1040)
         DO 60 I = 1, NDFMSH
            WRITE(OUNIT,1010) I
            WRITE(OUNIT,1020) (GSTIF(I,J), J = 1, BNDWTH)
   60    CONTINUE
         WRITE(OUNIT,1050)
         WRITE(OUNIT,1020) (GF(I), I = 1, NDFMSH)
      END IF

C --- IMPOSE PRIMARY BOUNDARY CONDITIONS.

      CALL UBNDRY    (NEQMAX, BNDMAX, NBCMAX, NDFNOD, NDFMSH, BNDWTH,
     .                NSGD  , ISGD  , VSGD  , GSTIF , GF    )

C --- SOLVE THE SYSTEM OF EQUATIONS.

      IRES = 0
      CALL SOLVE  (NEQMAX, BNDMAX, NDFMSH, BNDWTH, GSTIF , GF    ,
     .             IRES  )
 1000 FORMAT(' ',//3X,'STIFFNESS MATRIX FOR ELEMENT ONE: ')
 1010 FORMAT(' ',/5X,'ROW:  ',I2,/)
 1020 FORMAT(' ',5X,6E12.4)
 1030 FORMAT(' ',//3X,'FORCE VECTOR FOR ELEMENT ONE: ',/)
 1040 FORMAT(' ',//3X,'GLOBAL STIFFNESS MATRIX: ',/)
 1050 FORMAT(' ',//3X,'GLOBAL FORCE VECTOR: ',/)
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  FBNDRY                                              C
C     PURPOSE:     IMPOSE THE PRESCRIBED FORCE B.C.'S ON THE           C
C                  BANDED SYMMETRIC STIFFNESS MATRIX AND MODIFY        C
C                  THE FORCE VECTOR.                                   C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE FBNDRY (NEQMAX, NBCMAX, NDFNOD, NSGF  , ISGF  , VSGF  ,
```

```
     .                   GF     )

      IMPLICIT LOGICAL (A-Z)

      INTEGER            NEQMAX, NBCMAX, NDFNOD, NSGF  , ISGF  , IBC   ,
     .                   IDOF

      DOUBLE PRECISION   VSGF  , GF

      DIMENSION          ISGF(NBCMAX,2), VSGF(NBCMAX), GF(NEQMAX)

      DO 10 IBC = 1, NSGF
         IDOF = NDFNOD * (ISGF(IBC,1) - 1) + ISGF(IBC,2)
         GF(IDOF) = GF(IDOF) + VSGF(IBC)
   10 CONTINUE
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  GQUAD                                               C
C     PURPOSE:     ASSIGN VALUES TO THE ARRAYS "GAUSPT" AND "GAUSWT"   C
C                  USED IN GAUSSIAN QUADRATURE.                        C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE GQUAD  (NGPMAX, GAUSPT, GAUSWT )

      IMPLICIT LOGICAL (A-Z)

       INTEGER            NGPMAX, I     , J
        DOUBLE PRECISION   GAUSPT, GAUSWT

       DIMENSION          GAUSPT(NGPMAX,NGPMAX), GAUSWT(NGPMAX,NGPMAX)


C ---  INITIALIZE ARRAYS.

       DO 20 I = 1,NGPMAX
        DO 10 J = 1,NGPMAX
         GAUSPT(I,J) = 0.000
         GAUSWT(I,J) = 0.000
   10 CONTINUE
   20 CONTINUE

C --- GAUSSIAN QUADRATURE OF ORDER 1.

      GAUSPT(1,1) = 0.000
      GAUSWT(1,1) = 2.000

C --- GAUSSIAN QUADRATURE OF ORDER 2.

      GAUSPT(1,2) = -1.000/SQRT(3.000)
      GAUSPT(2,2) = -GAUSPT(1,2)
      GAUSWT(1,2) =  1.000
      GAUSWT(2,2) =  GAUSWT(1,2)

C --- GAUSSIAN QUADRATURE OF ORDER 3.

      GAUSPT(1,3) = -SQRT(3.000/5.000)
      GAUSPT(2,3) =  0.000
      GAUSPT(3,3) = -GAUSPT(1,3)
```

```
      GAUSWT(1,3) =  5.000/9.000
      GAUSWT(2,3) =  8.000/9.000
      GAUSWT(3,3) =  GAUSWT(1,3)

C --- GAUSSIAN QUADRATURE OF ORDER 4.

      GAUSPT(1,4) = -0.8611363116
      GAUSPT(2,4) = -0.3399810436
      GAUSPT(3,4) = -GAUSPT(2,4)
      GAUSPT(4,4) = -GAUSPT(1,4)
      GAUSWT(1,4) =  0.3478548451
      GAUSWT(2,4) =  0.6521451549
      GAUSWT(3,4) =  GAUSWT(2,4)
      GAUSWT(4,4) =  GAUSWT(1,4)

      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  INPELM                                              C
C     PURPOSE:     READ ELEMENT DATA.                                  C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INPELM (OUNIT , nodelm, ndfelm, ngpstf, ngpgrd,
     .                   ndfnod)

      IMPLICIT LOGICAL (A-Z)

      integer         OUNIT , nodelm, ndfelm, ngpstf, ngpgrd, ndfnod

      NDFNOD = 1
      NDFELM = 4
       NGPSTF = 2
      NGPGRD = 2
       NODELM = 4

      WRITE(OUNIT,1000)
      WRITE(OUNIT,1010) NODELM, ndfnod, ndfelm, ngpstf, ngpgrd

 1000 FORMAT(' ',//3X,'ELEMENT DATA:')
 1010 FORMAT(' ',
     . //10X,'NUMBER OF NODES PER ELEMENT, ............. NODELM =',I4,
     .  /10X,'NUMBER OF D.O.F. IN EACH NODE, ............ NDFNOD =',I4,
     .  /10X,'NUMBER OF D.O.F. IN EACH ELEMENT, ......... NDFELM =',I4,
     . //10X,'INTEGRATION RULE , ........................ NGPSTF =',I4,
     .  /10X,'INTEGRATION POINTS , ...................... NGPGRD =',I4)
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  INPLAY                                              C
C     PURPOSE:     INPUT ELEMENT LAYERS                                C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INPLAY (IUNIT , OUNIT , NELMSH, NLAYER,NLAMAX,NEMMAX,
     .                    KX    , KZ    , Z1LAY , Z2LAY)
```

```
      IMPLICIT LOGICAL (A-Z)

      INTEGER          I,J, IUNIT , OUNIT , NELMSH, NLAYER, NLAMAX,NEMMAX

      DOUBLE PRECISION   KX     , KZ     , Z1LAY , Z2LAY

      DIMENSION           NLAYER(NEMMAX),
     .                      KX(nemmax,NLAMAX),     KZ(nemmax,NLAMAX),
     .                    Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX)


      DO 80 I=1, NELMSH
          READ(IUNIT,*) NLAYER(I)
          DO 70 J=1, NLAYER(I)
             READ(IUNIT,*) KX(I,J), KZ(I,J), Z1LAY(I,J), Z2LAY(I,J)
70        CONTINUE
80    CONTINUE



      WRITE(OUNIT,1000)

       DO 100         I=1, NELMSH

      WRITE(OUNIT,1010)
       WRITE(OUNIT,1020) I, NLAYER(I)
      WRITE(OUNIT,1030)
                DO 90 J=1, NLAYER(I)
             WRITE(OUNIT,1040) J,KX(I,J),KZ(I,J),Z1LAY(I,J),Z2LAY(I,J)

90        CONTINUE
100    CONTINUE



C --- FORMAT STATEMENTS.

 1000 FORMAT(' ',///3X,'NUMBERS AND CHARACTERISTICS OF LAYERS: ')
 1010 FORMAT(' ',//6X,'ELEMENT',3X,'N LAYER',/)
 1020 FORMAT(' ',9X,I3,7X,I3)
 1030 FORMAT(' ',/6X,'N LAYER',5X,'KX',12X,'KZ',12X,'Z1LAY',9X,
     .                  'Z2LAY',5X,/)
 1040 FORMAT(' ',9X,I3,4X,E10.4,4X,E10.4,4X,E10.4,4X,E10.4)

      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
C     SUBROUTINE:  INPMSH                                            C
C     PURPOSE:     INPUT MESH PARAMETERS AND FORM THE MESH.          C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INPMSH (IUNIT , OUNIT , NEMMAX, NPEMAX, NNMMAX, nodelm,
     .                   ndfnod,DOM_X, DOM_Z, NELM_X, NELM_Z,
     .                   NELMSH, NDFMSH, nodmsh, node, BNDWTH, X, Z)

      IMPLICIT LOGICAL (A-Z)
```

```
      integer            IUNIT , OUNIT , NEMMAX, NPEMAX, NNMMAX, nodelm,
     .                   ndfnod, NELM_X, NELM_Z, nodmsh, BNDWTH,
     .                   NDFMSH,   node, IMESH, EORDER, NXINCR, NZINCR,
     .                   I     , J     ,       N, NELMSH, NW, NPE

       DOUBLE PRECISION   X , Z, DOM_X, DOM_Z, DX, DZ

      DIMENSION      NODE(NEMMAX,NPEMAX), X(NNMMAX), Z(NNMMAX),
       .               DX(NNMMAX)     , DZ(NNMMAX)

      READ(IUNIT,*) IMESH, DOM_X, DOM_Z, NELM_X, NELM_Z

      NODMSH = (NELM_X + 1)*(NELM_Z + 1)
      NELMSH =  NELM_X * NELM_Z

      IF (IMESH .EQ. 0) THEN
         DO 10 I = 1, NELMSH
            READ(IUNIT,*) (NODE(I,J), J=1,NODELM)
 10      CONTINUE
         DO 20 I = 1, NODMSH
         READ(IUNIT,*) X(I), Z(I)
 20      CONTINUE

      ELSE

         EORDER = 1
         NXINCR = NELM_X * EORDER
         NZINCR = NELM_Z * EORDER
         READ(IUNIT,*) (DX(I), I = 1, NXINCR)
         READ(IUNIT,*) (DZ(I), I = 1, NZINCR)
         NPE = NODELM
         CALL MESH   (NPEMAX, NEMMAX, NNMMAX, EORDER, NODE   , NPE    ,
     .                NODMSH, NELMSH, NELM_X, NELM_Z, X      , Z      ,
     .                DX     , DZ)
      END IF

      NDFMSH = NDFNOD * NODMSH

      BNDWTH = 0
       DO 1      N = 1,NELMSH
         DO 2    I = 1,NODELM
           DO 3  J = 1,NODELM
             NW = ( ABS( NODE(N,I) - NODE(N,J) ) + 1 )
            IF (BNDWTH .LT. NW) THEN
            BNDWTH = NW
             END IF
  3        CONTINUE
  2      CONTINUE
  1    CONTINUE

      WRITE(OUNIT,1000)
      WRITE(OUNIT,1010) NELM_X, NELM_Z, NODMSH, NDFMSH, BNDWTH
      WRITE(OUNIT,1020)
      WRITE(OUNIT,1030)
      DO 90 N = 1,NELMSH
         WRITE(OUNIT,1040) N, (NODE(N,I), I=1,NODELM)
 90   CONTINUE

1000  FORMAT(' ',//3X,'MESH DATA:')
1010  FORMAT(' ',
     . //10X,'N. OF ELEMENTS IN THE MESH IN X DIRECTION, .NELMSH =',I4,
     .   /10X,'N. OF ELEMENTS IN THE MESH IN Z DIRECTION, .NELMSH =',I4,
```

```
     . /10X,'NUMBER OF NODES IN THE MESH, ............. NODMSH =',I4,
     . /10X,'NUMBER OF D.O.F. IN THE MESH, ............ NDFMSH =',I4,
     . /10X,'HALF BAND WIDTH OF GLOBAL STIFFNESS MATRIX, BNDWTH =',I4)
 1020 FORMAT(' ',//10X,'BOOLEAN (CONNECTIVITY) MATRIX: ')
 1030 FORMAT(' ',/10X,'ELEMENT NO.',6X,'NODE(1,2,3,4)',/)
 1040 FORMAT(' ',11X,I3,10X,9(I3,1X))
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     SUBROUTINE:  MESH                                               C
C     PURPOSE:     GENERATE THE CONNECTIVITY ARRAY NODE(I,J), THE     C
C                  COORDINATES X(I), Z(I), AND MESH INFORMATION       C
C                  (NODMSH, NELMSH, NODELM) FOR RECTANGULAR DOMAINS.  C
C                  THE DOMAIN IS DIVIDED INTO RECTANGULAR ELEMENTS    C
C                  WITH FOUR, EIGHT, OR NINE NODES (NUMELX BY NUMELY  C
C                  NONUNIFORM MESH IN GENERAL).                       C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE MESH   (NPEMAX, NEMMAX, NNMMAX, IEL   , NOD   , NPE   ,
     .                   NNM   , NEM   , NX    , NZ    , X     , Z     ,
     .                   DX    , DZ    )

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

      INTEGER  NPEMAX, NEMMAX, NNMMAX, IEL   , NOD   , NPE   ,
     .         NNM   , NEM   , NX    , NZ

      DOUBLE PRECISION X     , Z     , DX    , DZ

      DIMENSION X(NNMMAX) , Z(NNMMAX) , NOD(NEMMAX,NPEMAX),
     .          DX(NNMMAX), DZ(NNMMAX)

      NEX1=NX+1
      NEZ1=NZ+1
      NXX=IEL*NX
      NZZ=IEL*NZ
      NXX1=NXX+1
      NZZ1=NZZ+1
      NEM=NX*NZ
      NNM=NXX1*NZZ1-(IEL-1)*NX*NZ
      IF(NPE.EQ.9)NNM=NXX1*NZZ1

C --- GENERATE THE CONNECTIVITY ARRAY NOD(I,J).

      K0=0
      IF(NPE.EQ.9)K0=1
      NOD(1,1) = 1
      NOD(1,2) = IEL+1
      NOD(1,3) = NXX1+(IEL-1)*NEX1+IEL+1
      IF (NPE .EQ. 9)NOD(1,3)=4*NX+5
      NOD(1,4) = NOD(1,3) - IEL
      IF(NPE .EQ. 4)GO TO 200
      NOD(1,5) = 2
      NOD(1,6) = NXX1 + (NPE-6)
      NOD(1,7) = NOD(1,3) - 1
      NOD(1,8) = NXX1+1
      IF (NPE .EQ. 9)NOD(1,9)=NXX1+2
  200 IF(NZ .EQ. 1)GOTO 230
```

```
      M = 1
      DO 220 N = 2,NZ
      L = (N-1)*NX + 1
      DO 210 I = 1,NPE
  210 NOD(L,I) = NOD(M,I)+NXX1+(IEL-1)*NEX1+K0*NX
  220 M=L
  230 IF(NX .EQ .1)GO TO 270
      DO 260 NI = 2,NX
      DO 240 I = 1,NPE
      K1 = IEL
      IF(I .EQ. 6 .OR. I .EQ. 8)K1=1+K0
  240 NOD(NI,I) = NOD(NI-1,I)+K1
      M = NI
      DO 260 NJ = 2,NZ
      L = (NJ-1)*NX+NI
      DO 250 J = 1,NPE
  250 NOD(L,J) = NOD(M,J)+NXX1+(IEL-1)*NEX1+K0*NX
  260 M = L

C --- GENERATE THE COORDINATES X(I) AND Z(I)

  270 DX(NXX1)=0.0
      DZ(NZZ1)=0.0
      ZC=0.0
      IF (NPE .EQ. 9) GOTO 310
      DO 300 NI = 1, NEZ1
      I = (NXX1+(IEL-1)*NEX1)*(NI-1)+1
      J = (NI-1)*IEL+1
      X(I) = 0.0
      Z(I) = ZC
      DO 280 NJ = 1,NXX
      I=I+1
      X(I) = X(I-1)+DX(NJ)
  280 Z(I) = ZC

      IF(NI.GT.NZ .OR. IEL.EQ.1)GO TO 300
      J = J+1
      ZC = ZC+DZ(J-1)
      I = I+1
      X(I) = 0.0
      Z(I) = ZC
      DO 290 II = 1, NX
      K = 2*II-1
      I = I+1
      X(I) = X(I-1)+DX(K)+DX(K+1)
  290 Z(I) = ZC

  300 ZC = ZC+DZ(J)
      RETURN

  310 DO 330 NI=1,NZZ1
      I=NXX1*(NI-1)
      XC=0.0
      DO 320 NJ=1,NXX1
      I = I+1
      X(I) = XC
      Z(I) = ZC
  320 XC = XC + DX(NJ)
  330 ZC = ZC + DZ(NI)

      RETURN
      END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  INPNBC                                              C
C     PURPOSE:     INPUT NODAL BOUNDARY CONDITIONS.                    C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INPNBC (IUNIT , OUNIT , NBCMAX, NNMMAX, NODMSH, ndfnod,
     .                   NSGD  , NSGF  , ISGD  , ISGF  , X,   Z , vsgd  ,
     .                   VSGF  )

      IMPLICIT LOGICAL (A-Z)

      INTEGER            IUNIT , OUNIT , NBCMAX, NNMMAX, NODMSH, ndfnod,
     .                   NSGD  , NSGF  , ISGD  , ISGF  ,
     .                   I     , J     , K     , IBC   , NOD   , DOF

      DOUBLE PRECISION   X  ,    Z, VSGD  , VSGF

      DIMENSION          ISGD(NBCMAX,2), ISGF(NBCMAX,2), IBC(4),
     .                   VSGD(NBCMAX)  , VSGF(NBCMAX)  , X(NNMMAX),
     .                   Z(NNMMAX)

      READ(IUNIT,*) NSGD, NSGF
      IF (NSGD .GT. 0) THEN
         READ(IUNIT,*)  ((ISGD(I,J),J=1,2),I=1,NSGD)
         READ(IUNIT,*)  (VSGD(I),I=1,NSGD)
      END IF
      IF (NSGF .GT. 0) THEN
         READ(IUNIT,*)  ((ISGF(I,J),J=1,2),I=1,NSGF)
         READ(IUNIT,*)  (VSGF(I),I=1,NSGF)
      END IF

      WRITE(OUNIT,1000)
      WRITE(OUNIT,1010) NSGD
      WRITE(OUNIT,1020)
      DO 30 I = 1,NODMSH
         DO 10 K = 1,NDFNOD
            IBC(K) = 0
   10    CONTINUE
         DO 20 J = 1,NSGD
            NOD = ISGD(J,1)
            DOF = ISGD(J,2)
            IF (NOD .EQ. I) IBC(DOF) = DOF
   20    CONTINUE
         WRITE(OUNIT,1030) I, X(I), Z(I), (IBC(K), K = 1, NDFNOD)
   30 CONTINUE
      WRITE(OUNIT,1035)
      WRITE(OUNIT,1070) (VSGD(I), I = 1, NSGD)

      IF (NSGF .NE. 0) THEN
         WRITE(OUNIT,1040) NSGF
         WRITE(OUNIT,1050)
         DO 60 I = 1,NODMSH
         DO 40 K = 1,NDFNOD
            IBC(K) = 0
   40    CONTINUE
         DO 50 J = 1,NSGF
            NOD = ISGF(J,1)
            DOF = ISGF(J,2)
            IF (NOD .EQ. I) IBC(DOF) = DOF
   50    CONTINUE
         WRITE(OUNIT,1030) I, X(I), Z(I), (IBC(K), K = 1, NDFNOD)
```

```
   60     CONTINUE
          WRITE(OUNIT,1060)
          WRITE(OUNIT,1070) (VSGF(I), I = 1, NSGF)
       END IF

C --- FORMAT STATEMENTS.

 1000 FORMAT(' ',//3X,'NODAL BOUNDARY CONDITIONS: ',/)
 1010 FORMAT(' ',//10X,'NUMBER OF SPECIFIED TEMPERATURE D.O.F =',I4)
 1020 FORMAT(' ',//10X,'NODE',4X,'X-COORD.',6X,'Z-COORD.'5X,
     .                 'SPECIFIED TEMPERATURE D.O.F.',/)
 1030 FORMAT(' ',9X,I3,4X,E10.4,4X,E10.4,3X,12(I2,1X))
 1035 FORMAT(' ',/10X,'VALUES OF THE SPECIFIED TEMPERATURE D.O.F.:')
 1040 FORMAT(' ',//10X,'NUMBER OF SPECIFIED GENERALIZED FORCES =',I4)
 1050 FORMAT(' ',//10X,'NODE',4X,'X-COORD.',4X,'Z-COORD.',5X,
     .                 'SPECIFIED GENERALIZED FORCE',/)
 1060 FORMAT(' ',/10X,'VALUES OF THE SPECIFIED GENERALIZED FORCES:')
 1070 FORMAT(' ',/10X,6(E10.4,2X))
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  INPOPT                                              C
C     PURPOSE:     READ OPTION PARAMETERS FROM THE INPUT FILE.         C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INPOPT (IUNIT , OUNIT , PRTSTF, PRTGRD)

      IMPLICIT LOGICAL (A-Z)

      INTEGER            IUNIT , OUNIT , PRTSTF, PRTGRD

      CHARACTER*72       TITLE

      READ(IUNIT,1000)   TITLE
       READ(IUNIT,*)      PRTSTF
C      READ(IUNIT,*)      PRTSTF, PRTGRD

       PRTGRD = 0

      WRITE(OUNIT,1010)
      WRITE(OUNIT,1020)  TITLE

 1000 FORMAT(A72)
 1010 FORMAT(' ',//3X,
     .   'RESULTS OF "TERM2D VERS. 1.0" - WRITTEN BY ANTONIO PANTANO')
 1020 FORMAT(' ',//3X,A72)
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  PRTGRD                                              C
C     PURPOSE:     WRITE THE SOLUTION GRADIENTS TO THE OUTPUT FILE.    C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine prtdu  (ounit , npemax, neqmax, nnmmax, edfmax, nemmax,
     .                   ngpmax, nodelm, ndfnod, ndfelm, nelmsh, ngpgrd,
     .                   node  ,
     .                   gauspt, gu    , eu    , ex    , sf    ,
```

```
     .                   gdsf  , a0    , a1    , a2    , x     )

      IMPLICIT LOGICAL (A-Z)

      integer            ounit , npemax, neqmax, nnmmax, edfmax, nemmax,
     .                   ngpmax, nodelm, ndfnod, ndfelm, nelmsh, ngpgrd,
     .                   node  ,
     .                   elmnum, idof  , igp   , enod  , gnod  , edof  ,
     .                   gdof

      double precision   gauspt, gu    , eu    , ex    , sf    ,
     .                   gdsf  , a0    , a1    , a2    , x     ,
     .                   xc    , dudx  , adudx , a     , xi    , elenth


      dimension          gu(neqmax)    , eu(edfmax)    , ex(npemax)    ,
     .                   sf(npemax)    , gdsf(npemax)  ,
     .                   a0(nemmax)    , a1(nemmax)    , a2(nemmax)    ,
     .                   x(nnmmax)     ,
     .                   gauspt(ngpmax,ngpmax), node(nemmax,npemax)

       WRITE(OUNIT,1000)
       WRITE(OUNIT,1010)

C --- BEGIN LOOP OVER NUMBER OF ELEMENTS IN THE MESH.

      DO 50 ELMNUM = 1, NELMSH

C --- DETERMINE THE LOCAL (ELEMENT) SOLUTION VECTOR AND NODAL
C --- COORDINATES.

      gdof = (node(elmnum,1) - 1) * ndfnod
      do 10 edof = 1, ndfelm
         eu(edof) = gu(gdof+edof)
   10 continue
      do 20 enod = 1, nodelm
         gnod = node(elmnum,enod)
         ex(enod) = x(gnod)
   20 continue
      elenth = ex(nodelm) - ex(1)

C --- BEGIN LOOP ON NUMBER OF GAUSS POINTS (NGPGRD).

      DO 40 IGP = 1, NGPGRD
      XI = GAUSPT(IGP,NGPGRD)

C --- COMPUTE THE SOLUTION GRADIENTS AT THE GAUSS POINTS.

CA      CALL SHPL1D (NPEMAX, NODELM, XI    , ELENTH, SF    , DSF   ,
CA    .                 GDSF  , JACOBN)

      DUDX = 0.000
      XC   = 0.000
      DO 30 IDOF = 1, NODELM
         DUDX = DUDX + EU(IDOF) * GDSF(IDOF)
         XC   = XC   + EX(IDOF) * SF(IDOF)
   30 CONTINUE

      A     = A0(elmnum) + A1(elmnum)*XC + A2(elmnum)*XC*XC
      ADUDX = A*DUDX

      WRITE(OUNIT,1030) ELMNUM, XC, DUDX, ADUDX
```

```
   40 CONTINUE
   50 CONTINUE

C --- FORMAT STATEMENTS.

 1000 FORMAT(' ',/3X,'SOLUTION GRADIENTS:  ',/)
 1010 FORMAT(' ',/10X,'ELEMENT NO.  X-COORD.      DU/DX       A*DU/DX',/)
 1030 FORMAT(' ', 11X,I3,6X,4(E10.4,2X))
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  PRTGU                                               C
C     PURPOSE:     WRITE THE SOLUTION VECTOR TO THE OUTPUT FILE.       C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine prtgu  (OUNIT , NEQMAX, NNMMAX, NDFNOD, NODMSH, GU    ,
     .                   X, Z )

      IMPLICIT LOGICAL (A-Z)

      INTEGER            OUNIT , NEQMAX, NNMMAX, NDFNOD, NODMSH,
     .                   U1    , U2    , I     , J

      DOUBLE PRECISION   GU    , X,    Z

      DIMENSION          GU(NEQMAX)    , X(NNMMAX),  Z(NNMMAX)

      WRITE(OUNIT,1000)

C --- OUTPUT OF SOLUTION VECTOR.

      WRITE(OUNIT,1010)
      U2 = 0
      DO 20 I = 1, NODMSH
         U1 = U2 + 1
         U2 = U2 + ndfNOD
         WRITE(OUNIT,1020) I, X(I), Z(I), (GU(J), J = U1, U2)
   20 CONTINUE

C --- FORMAT STATEMENTS.

 1000 FORMAT(' ',///3X,'SOLUTION:  ',/)
 1010 FORMAT(' ',10X,'NODE',7X,'X-COORD.',5X,'Z-COORD. '6X, 'U',/)
 1020 FORMAT(' ', 10X,I3,6X,E12.4,2X,E12.4,2X,E12.6)
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  SHPL1D_T                                            C
C     PURPOSE:     EVALUATE THE 1-D LAGRANGIAN INTERPOLATION FUNCTIONS C
C                  AND THEIR GLOBAL DERIVATIVES AT THE GAUSS POINTS.   C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE SHPL1D_T (NPEMAX, XI , ELENTH_X, DSF,
     .                     GDSF  , JACOBN, SF2, IGP)
```

```
      IMPLICIT LOGICAL (A-Z)

      INTEGER              NPEMAX,  I,  IGP

      DOUBLE PRECISION     XI   , ELENTH_X, DSF   , GDSF  , JACOBN,
     .                       SF2

      DIMENSION            DSF(NPEMAX), GDSF(NPEMAX), SF2(2,2)

C --- LINEAR LAGRANGE INTERPOLATION FUNCTIONS FOR 2-NODED ELEMENTS.

         SF2(1,IGP)  =  0.5000 * (1.000 - XI)
         SF2(2,IGP)  =  0.5000 * (1.000 + XI)
         DSF(1) = -0.5000
         DSF(2) =  0.5000

C --- COMPUTE THE GLOBAL DERIVATIVES OF SF(I).

      JACOBN = ELENTH_X * 0.5000

      DO 10 I = 1, 2
         GDSF(I) = DSF(I) / JACOBN
   10 CONTINUE
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SOLVE:                                                           C
C     THIS PROGRAM SOLVES A BANDED SYMMETRIC SYSTEM OF EQUATIONS.       C
C     THE BANDED MATRIX IS INPUT THROUGH BAND(NEQNS,NBW), AND          C
C     RHS IS THE RIGHT HAND SIDE (FORCE VECTOR) OF THE EQUATION.       C
C     NEQNS IS THE NO. OF EQUATIONS AND NBW IS THE HALF BAND WIDTH.    C
C     IN RESOLVING, IRES .GT. 0, LHS ELIMINATION IS SKIPPED.           C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C         CALL SOLVE  (NEQMAX, BNDMAX, NDFMSH, BNDWTH,GSTIF, GF, IRES )
       SUBROUTINE SOLVE  (NRM ,  NCM  , NEQNS , NBW   , BAND, RHS, IRES)

       IMPLICIT DOUBLE PRECISION (A-H,O-Z)

      INTEGER              NRM   , NCM   , NEQNS , NBW   , IRES

      DOUBLE PRECISION     BAND  , RHS

      DIMENSION            BAND(NRM,NCM), RHS(NRM)


      MEQNS=NEQNS-1
      IF (IRES.GT.0) GO TO 40
      DO 30 NPIV=1,MEQNS
      NPIVOT=NPIV+1
      LSTSUB=NPIV+NBW-1
      IF (LSTSUB.GT.NEQNS) LSTSUB=NEQNS
      DO 20 NROW=NPIVOT,LSTSUB


C --- INVERT ROWS AND COLUMNS FOR ROW FACTOR.

      NCOL=NROW-NPIV+1
```

```
      FACTOR=BAND(NPIV,NCOL)/BAND(NPIV,1)
      DO 10 NCOL=NROW,LSTSUB
      ICOL=NCOL-NROW+1
      JCOL=NCOL-NPIV+1
   10 BAND(NROW,ICOL)=BAND(NROW,ICOL)-FACTOR*BAND(NPIV,JCOL)
   20 RHS(NROW)=RHS(NROW)-FACTOR*RHS(NPIV)
   30 CONTINUE
      GO TO 70
   40 DO 60 NPIV=1,MEQNS
      NPIVOT=NPIV+1
      LSTSUB=NPIV+NBW-1
      IF (LSTSUB.GT.NEQNS) LSTSUB=NEQNS
      DO 50 NROW=NPIVOT,LSTSUB
      NCOL=NROW-NPIV+1
      FACTOR=BAND(NPIV,NCOL)/BAND(NPIV,1)
   50 RHS(NROW)=RHS(NROW)-FACTOR*RHS(NPIV)
   60 CONTINUE

C --- BACK SUBSTITUTION.

   70 DO 90 IJK=2,NEQNS
      NPIV=NEQNS-IJK+2
      RHS(NPIV)=RHS(NPIV)/BAND(NPIV,1)
      LSTSUB=NPIV-NBW+1
      IF (LSTSUB.LT.1) LSTSUB=1
      NPIVOT=NPIV-1
      DO 80 JKI=LSTSUB,NPIVOT
      NROW=NPIVOT-JKI+LSTSUB
      NCOL=NPIV-NROW+1
      FACTOR=BAND(NROW,NCOL)
   80 RHS(NROW)=RHS(NROW)-FACTOR*RHS(NPIV)
   90 CONTINUE
      RHS(1)=RHS(1)/BAND(1,1)
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  STIFF_T                                             C
C     PURPOSE:     COMPUTE THE ELEMENTAL STIFFNESS AND FORCE ARRAYS    C
C                                                                      C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE STIFF_T(EDFMAX, NPEMAX, NGPMAX, ndfelm,
     .                     NGPstf, ELMNUM,
     .                     DSF, GDSF, EX, ESTIF, EF, GAUSPT, GAUSWT,
     .                   EZ, KX    , KZ,    Z1LAY,     Z2LAY, NEMMAX,
     .                   NLAMAX, NLAYER,AK, FI_D, D_FI,
     .                   SF2)


      IMPLICIT LOGICAL (A-Z)

      integer            EDFMAX, NPEMAX, NGPMAX,  ndfelm,
     .                   NGPstf, I     , J     , igp,
     .                   NLAMAX, NLAYER, NEMMAX, ELMNUM, K

      double precision   DSF   , GDSF  , EX    , ESTIF , EF    ,
     .                   GAUSPT, GAUSWT,
     .                   xi    , const , jacobn,
     .                   ELENTH_X, ELENTH_Z,
```

```
     .                    EZ, KX   , KZ,   Z1LAY,  Z2LAY,
     .                      AK,   FI_D,  D_FI, SF2, K_LAY,
     .                    FI, D_FI_F, FI_N, H_LAYER, FXI, DXI


      dimension  dsf(npemax), gdsf(npemax),
     .           EX(NPEMAX), ef(edfmax),
     .           GAUSPT(NGPMAX,NGPMAX), GAUSWT(NGPMAX,NGPMAX) ,
     .           ESTIF(EDFMAX,EDFMAX), EZ(NPEMAX), NLAYER(NEMMAX),
     .           KX(nemmax,NLAMAX), KZ(nemmax,NLAMAX),
     .           Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .           AK(nemmax,NLAMAX), FI_D(nemmax),
     .           D_FI(nemmax,NLAMAX), SF2(2,2), K_LAY(4,4,NLAMAX),
     .             FI(2,3), D_FI_F(2), FXI(2,3), DXI(2)



      ELENTH_X = EX(2) - EX(1)
       ELENTH_Z = EZ(4) - EZ(1)

C --- INITIALIZE THE ARRAYS.

      DO 2 I= 1, 4
         DO 3 J= 1, 4
           DO 4 K= 1, NLAYER(ELMNUM)
              K_LAY(I,J,K) = 0.000
    4      CONTINUE
    3    CONTINUE
    2 CONTINUE


      DO 20 I = 1, NDFELM
         EF(I) = 0.000
         DO 10 J = 1, NDFELM
            ESTIF(I,J) = 0.000
   10    CONTINUE
   20 CONTINUE

C --- BEGIN GAUSS QUADRATURE LOOP FOR FULL INTEGRATION.

      DO 21 IGP = 1, NGPstf

      XI = GAUSPT(IGP,NGPstf)

      CALL SHPL1D_T (NPEMAX, XI, ELENTH_X, DSF,
     .                GDSF  , JACOBN, SF2, IGP)

        CONST = JACOBN*GAUSWT(IGP,NGPstf)
   21 CONTINUE


      DO 22 I= 1, 4
         DO 23 J= 1, 4
           DO 24 K= 1, NLAYER(ELMNUM)

          CALL    FI_AT_N (I, J, K, ELMNUM, Z1LAY, Z2LAY, AK,
     .                        NEMMAX, NLAMAX, npemax, FI_D, D_FI,
     .                        D_FI_F, FI, FI_N, FXI, DXI, SF2, gdsf)

             H_LAYER  = ( Z2LAY(ELMNUM,K) - Z1LAY(ELMNUM,K) ) / 2

       K_LAY(I,J,K)= H_LAYER/3 *              (
     .        JACOBN * KX(ELMNUM,K) * ( 2*DXI(1)*DXI(2) ) *
```

```
     .       ( FI(1,1)*FI(2,1) + 4*FI(1,2)*FI(2,2) + FI(1,3)*FI(2,3) )+
    .       6 * JACOBN * KZ(ELMNUM,K) *
    .       ( FXI(1,1)*FXI(2,1) + FXI(1,2)*FXI(2,2) )*
    .       ( D_FI_F(1) * D_FI_F(2) )    )

         ESTIF(I,J) = ESTIF(I,J) + K_LAY(I,J,K)
   24     CONTINUE
   23   CONTINUE

   22 CONTINUE




   50 CONTINUE
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                    C
C     SUBROUTINE:  UBNDRY                                            C
C     PURPOSE:     IMPOSE THE PRESCRIBED DISPLACEMENT B.C.'S ON THE  C
C                  BANDED SYMMETRIC STIFFNESS MATRIX AND MODIFY      C
C                  THE FORCE VECTOR.                                 C
C                                                                    C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE UBNDRY (NEQMAX, BNDMAX, NBCMAX, NDFNOD, NDFMSH, BNDWTH,
     .                   NSGD  , ISGD  , VSGD  , GSTIF , GF    )

      IMPLICIT LOGICAL (A-Z)

      INTEGER            NEQMAX, BNDMAX, NBCMAX, NDFNOD, NDFMSH, BNDWTH,
     .                   NSGD  , ISGD  , IBC   , IDOF  , BWM1  , ROW   ,
     .                   COL   , K

      DOUBLE PRECISION   VSGD  , GSTIF , GF    , UVALUE

      DIMENSION ISGD(NBCMAX,2), VSGD(NBCMAX), GSTIF(NEQMAX,BNDMAX),
     .          GF(NEQMAX)


      DO 40 IBC = 1, NSGD
         IDOF   = NDFNOD * (ISGD(IBC,1) - 1) + ISGD(IBC,2)
         UVALUE = VSGD(IBC)
         BWM1   = BNDWTH - 1
         ROW    = IDOF - BNDWTH

         DO 20 K = 1, BWM1
            ROW = ROW + 1
            IF (ROW .GT. 0) THEN
               COL = IDOF - ROW + 1
               GF(ROW) = GF(ROW) - GSTIF(ROW,COL) * UVALUE
               GSTIF(ROW,COL) = 0.000
            END IF
   20    CONTINUE

         GSTIF(IDOF,1) = 1.000
         GF(IDOF) = UVALUE
```

```
        ROW = IDOF

        DO 30 K = 2, BNDWTH
           ROW = ROW + 1
           IF (ROW .LE. NDFMSH) THEN
              GF(ROW) = GF(ROW) - GSTIF(IDOF,K) * UVALUE
             GSTIF(IDOF,K) = 0.000
         END IF
   30   CONTINUE
   40 CONTINUE
      RETURN
       END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                       C
C     SUBROUTINE:  FI_AT_N                                              C
C     PURPOSE:     CHOOSE THE FUNCTION FI AND XI TO BE USED IN THE      C
C                  EVALUATION OF THE ACTUAL K_LAY                       C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE  FI_AT_N (I, J, K, ELMNUM, Z1LAY, Z2LAY, AK,
      .                     NEMMAX, NLAMAX, npemax, FI_D, D_FI,
      .                     D_FI_F, FI, FI_N, FXI, DXI, SF2, gdsf)


      IMPLICIT LOGICAL (A-Z)

      integer              I, J, K, R, Q,
     .                     NLAMAX, NEMMAX, ELMNUM, npemax

      double precision     Z1LAY,  Z2LAY, AK,  FI_D,  D_FI,
     .                     FI, D_FI_F, FI_N, ZZ, FXI, DXI,
      .                      gdsf,SF2


      dimension  Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .           AK(nemmax,NLAMAX), FI_D(nemmax),
     .           D_FI(nemmax,NLAMAX),
      .            FI(2,3), D_FI_F(2), FXI(2,3), DXI(2),
     .           gdsf(npemax), SF2(2,2)


      DO 90 R = 1,2

            IF (I .EQ. 1 .OR. I .EQ. 4) THEN
         DXI(1)    = gdsf(1)
            FXI(1,R) = SF2(1,R)
            ELSE
         DXI(1)    = gdsf(2)
            FXI(1,R) = SF2(2,R)
            END IF

         IF (J .EQ. 1 .OR. J .EQ. 4) THEN
         DXI(2)    = gdsf(1)
            FXI(2,R) = SF2(1,R)
         ELSE
         DXI(2)    = gdsf(2)
            FXI(2,R) = SF2(2,R)
         END IF

   90  CONTINUE
```

```
        DO 110 R = 1,3

         FI_N =  0.000
           IF      (R .EQ. 1) THEN
           ZZ= Z1LAY(ELMNUM,K)-Z1LAY(ELMNUM,1)
           ELSE IF (R .EQ. 2) THEN
           ZZ=( (Z2LAY(ELMNUM,K)-Z1LAY(ELMNUM,1)) +
     .           (Z1LAY(ELMNUM,K)-Z1LAY(ELMNUM,1))   )/2
           ELSE IF (R .EQ. 3) THEN
           ZZ= Z2LAY(ELMNUM,K)-Z1LAY(ELMNUM,1)
         END IF

         FI_N =  ZZ

          DO 100 Q = 2, K
          FI_N = FI_N +(ZZ - (Z2LAY(ELMNUM,Q-1)-Z1LAY(ELMNUM,1)))
     .                *AK(ELMNUM,Q-1)


 100      CONTINUE

           IF (I .LT. 3) THEN
         D_FI_F(1) = -D_FI(ELMNUM,K)
           FI(1,R)= (1 - FI_N/FI_D(ELMNUM) )
           ELSE
         D_FI_F(1) = D_FI(ELMNUM,K)
           FI(1,R) = FI_N/FI_D(ELMNUM)
           END IF

         IF (J .LT. 3) THEN
         D_FI_F(2) = -D_FI(ELMNUM,K)
           FI(2,R) = (1 - FI_N/FI_D(ELMNUM) )
         ELSE
         D_FI_F(2) = D_FI(ELMNUM,K)
           FI(2,R) = FI_N/FI_D(ELMNUM)
         END IF

 110  CONTINUE

      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     SUBROUTINE:  PR_L_T                                             C
C     PURPOSE:     EVALUATE FUNCTIONS FI AND ITS DERIVATIVES          C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE PR_L_T(OUNIT, NPEMAX,NLAMAX,NELMSH, NEMMAX,
     .                  NLAYER, NEQMAX, Z1LAY, Z2LAY,
     .                  AK, FI_D, NODE, GUL, GF)


      IMPLICIT LOGICAL (A-Z)

      integer            OUNIT , NPEMAX, NLAMAX, NELMSH,NEMMAX,
     .                     NEQMAX, NLAYER, I , J, K1, K2, NODE

      double precision   Z1LAY, Z2LAY, AK, FI_D,
     .                   GUL, GF, T_B,
```

```
     .                   T_T, FI_N, FI_T

      DIMENSION  NLAYER(NEMMAX),        NODE(NEMMAX,NPEMAX),
     .           Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .             AK(nemmax,NLAMAX),    FI_D(nemmax),
     .           GUL(nemmax,2,NLAMAX), GF(NEQMAX)

      DO 10 I = 1, NELMSH

        DO 20 J = 1, 2

         IF (J .EQ. 1) THEN
         T_B = GF(NODE(I,1))
         T_T = GF(NODE(I,4))
         FI_T = (Z2LAY(I,1)- Z1LAY(I,1)) / FI_D(I)
         GUL(I,1,1) = T_B * (1 - FI_T) + T_T * FI_T
         FI_T = 0.0000
           DO 30 K1 = 2, NLAYER(I)
             FI_N = Z2LAY(I,K1) - Z1LAY(I,1)
              DO 40 K2 = 2, K1
                FI_N = FI_N +(Z2LAY(I,K1) - Z2LAY(I,K2-1))*AK(I,K2-1)
   40    CONTINUE
             FI_T = FI_N / FI_D(I)
              GUL(I,1,K1) = T_B * (1 - FI_T) + T_T * FI_T
   30  CONTINUE

         ELSE

         T_B = GF(NODE(I,2))
         T_T = GF(NODE(I,3))
         FI_T = (Z2LAY(I,1)- Z1LAY(I,1))/FI_D(I)
         GUL(I,2,1) = T_B * (1 - FI_T) + T_T * FI_T
         FI_T = 0.0000
           DO 50 K1 = 2, NLAYER(I)
             FI_N = Z2LAY(I,K1) - Z1LAY(I,1)
              DO 60 K2 = 2, K1
                FI_N = FI_N +(Z2LAY(I,K1) - Z2LAY(I,K2-1))*AK(I,K2-1)
   60    CONTINUE
             FI_T = FI_N / FI_D(I)
              GUL(I,2,K1) = T_B * (1 - FI_T) + T_T * FI_T
   50  CONTINUE

         END IF
   20   CONTINUE
   10 CONTINUE

       WRITE(OUNIT,1000)

       DO 100        I=1, NELMSH

      WRITE(OUNIT,1010)
       WRITE(OUNIT,1020) I, NLAYER(I)
      WRITE(OUNIT,1030)
                DO 90 J=1, NLAYER(I)
            WRITE(OUNIT,1040) J, Z2LAY(I,J), GUL(I,1,J), GUL(I,2,J)

90         CONTINUE
100     CONTINUE
```

```
C --- FORMAT STATEMENTS.

 1000 FORMAT(' ',///3X,'SOLUTION AT EACH LAYER: ')
 1010 FORMAT(' ',//6X,'ELEMENT',3X,'LAYER',/)
 1020 FORMAT(' ',9X,I3,7X,I3)
 1030 FORMAT(' ',/6X,'LAYER N.',5X,'Z LAYER',6X,'SIDE 1-4',6X,
     .                'SIDE 2-3',/)
 1040 FORMAT(' ',9X,I3,4X, E12.4,4X, E12.6,4X, E12.6)

      RETURN
      END
```

APPENDIX B FORTRAN CODE, SINLOAD

```
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C                                                                      C
C                          S Y N L O A D                               C
C                                                                      C
C                                                                      C
C     This program solves the two-dimensional the steady-state         C
C     heat conduction problem in 2-d for laminated plates              C
C     subjected to a synusoidal temperature distribution               C
C     on the top and bottom surfaces.                                  C
C                                                                      C
C                                                                      C
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
C                                                                      C
C                                INPUT:                                C
C                                                                      C
C     >  TITLE (A70)                                                   C
C     >  NMATER  NLAYER                                                C
C                                                                      C
C     DO LOOP: I = 1 TO NMATER                                         C
C     >  MATNAM(I)                                                     C
C     >  K1(I)    K3(I)                                                C
C     END OF LOOP                                                      C
C                                                                      C
C     DO LOOP: I = 1 TO NLAYER                                         C
C     >  MATNUM(I)  Z(I)                                               C
C     END OF LOOP                                                      C
C                                                                      C
C     >  XLENTH                                                        C
C                                                                      C
C     >  T_B  T_T                                                      C
C     DO LOOP: I = 1 TO P_SOL                                          C
C     >  PX(I)                                                         C
C     END OF LOOP                                                      C
C                                                                      C
C                                                                      C
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      implicit none

      INTEGER             IUNIT , OUNIT , LAYMAX, matmax, neqmax

      PARAMETER          (IUNIT  = 10, OUNIT  = 12, laymax = 20,
     .                    matmax = 10, NEQMAX = 2 * laymax)

      integer             nmater, nlayer,  matnum, P_SOL,
     .                     ipivot

      double precision    Z, K1, K3, K1L, K3L, L, XLENTH, T_B, T_T, PX,
     .                     tthick,  a     , f

      character           matnam*50


        dimension       Z(laymax+1), K1(matmax), K3(matmax),
     .                   K1L(laymax), K3L(laymax), L(laymax), PX(5),
     .                  matnum(laymax), MATNAM(MATMAX),
     .                 f(neqmax), a(neqmax,neqmax),  ipivot(neqmax)

      open(unit=10,file='Synload.data',status='old')
      open(unit=12,file='Synload.out',status='unknown')

C --------------------
```

```
C --- PREPROCESSOR ---
C --------------------

      call inpdat       (iunit , ounit , laymax, matmax, nmater, nlayer,
     .                   matnum, matnam, P_SOL,
     .                   Z, K1, K3, XLENTH, T_B, T_T, PX,
     .                   tthick)



C -----------------
C --- PROCESSOR ---
C -----------------

      call Solve        (ounit , neqmax,laymax, matmax,
      .                   nlayer, matnum, P_SOL, ipivot, Z, K1, K3,
     .                   K1L, K3L, L, XLENTH, T_B, T_T, PX, a, f)



C ---------------------
C --- POSTPROCESSOR ---
C ---------------------

      STOP
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  INPDAT                                              C
C     PURPOSE:     Read input data from a file.                        C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

       SUBROUTINE INPDAT (iunit , ounit , laymax, matmax, nmater,
     .                   nlayer, matnum, matnam, P_SOL,
     .                   Z, K1, K3,  XLENTH, T_B, T_T, PX,
     .                   tthick)

          IMPLICIT none

       integer          iunit , ounit , laymax, matmax, nmater, nlayer,
     .                   matnum,  P_SOL, i

      double precision   Z, K1, K3, XLENTH, T_B, T_T, PX,
     .                   tthick

       CHARACTER          TITLE*70      , MATNAM*50

       dimension          Z(laymax+1), K1(matmax), K3(matmax),
     .                   PX(5), matnum(laymax), MATNAM(MATMAX)


C ---  READ TITLE AND CONTROL DATA.

      READ(iunit,5000) TITLE
      READ(iunit,*) nmater, nlayer

C --- READ MATERIAL PROPERTIES.

      DO 10 I = 1,nmater
      READ(iunit,5010) MATNAM(I)
```

```
      READ(iunit,*) K1(I), K3(I)
   10 CONTINUE

C --- READ LAMINATION SCHEME.

      tthick = 0.0D0
      DO 20 I = 1,nlayer
      READ(iunit,*) MATNUM(I), Z(I)
   20 CONTINUE
      tthick = Z(nlayer)
      READ(iunit,*) xlenth
       READ(iunit,*) T_B, T_T
       READ(iunit,*) P_SOL
      DO 30 I = 1, P_SOL
      READ(iunit,*) PX(I)
   30 CONTINUE



C --- PRINT INPUT DATA.

      WRITE(ounit,4900)
      WRITE(ounit,5020) TITLE
      WRITE(ounit,5025)
      WRITE(ounit,5030)
      DO 40 I = 1, NMATER
      WRITE(ounit,5040) I,MATNAM(I), K1(I), K3(I)
   40 CONTINUE
      WRITE(ounit,5070)
      WRITE(ounit,5080)
      DO 50 I = 1,NLAYER
      WRITE(ounit,5090) I, MATNUM(I), Z(I)
   50 CONTINUE
      WRITE(ounit,5100) tthick, xlenth
      WRITE(ounit,5200) T_B, T_T
      WRITE(ounit,5300)
      DO 60 I = 1, P_SOL
      WRITE(ounit,5400) PX(I)
   60 CONTINUE

C --- FORMAT STATEMENTS.

 4900 FORMAT(' ',//3X,'RESULTS FROM PROGRAM "SYNLOAD" - WRITTEN BY ',
     .             'ANTONIO PANTANO')
 5000 FORMAT(A70)
 5010 FORMAT(A50)
 5020 FORMAT(' ',//3X,A70)
 5025 FORMAT(' ',//3X,'INPUT DATA:')
 5030 FORMAT(' ',//6X,'MATERIAL PROPERTIES:')
 5040 FORMAT(' ',/10X,'MATERIAL NO. ',I1,':  ',A50,
     .            //10X,'K1     = ',E12.4,
     .             /10X,'K3     = ',E12.4)
 5070 FORMAT(' ',//6X,'LAMINATION SCHEME:')
 5080 FORMAT(' ',/10X,'LAYER',3X,'MATERIAL',4X,'Z LAYER',/)
 5090 FORMAT(' ',10X,I2,8X,I1,6X,E12.4,6X,E12.4)
 5100 FORMAT(' ',/10X,'TOTAL THICKNESS OF LAMINATE =  ',E12.4,
     .             /10X,'X-DIMENSION OF LAMINATE =       ',E12.4)
 5200 FORMAT(' ',/10X,'TEMPERATURE AT THE BOTTOM   =  ',E12.4,
     .             /10X,'TEMPERATURE AT THE TOP      =  ',E12.4)
 5300 FORMAT(' ',//6X,'X AT WHICH THE SOLUTION IS REQUIRED:')
 5400 FORMAT(' ',/10X,'COORDINATE X   =  ',E12.4)
      RETURN
      END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                        C
C     Subroutine:  Solve                                                 C
C                                                                        C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine Solve  (ounit , neqmax, laymax, matmax,
       .                  nlayer,  matnum, P_SOL, ipivot, Z, K1, K3,
     .                   K1L, K3L, L, XLENTH, T_B, T_T, PX, a, f)

      implicit none

      integer            ounit , laymax, neqmax, ipivot,
       .                   matmax, nlayer,  matnum, P_SOL,
     .                   i,  j   , k ,  neq, ierror, ii


      double precision   X, Z, K1, K3, K1L, K3L, L, XLENTH, T_B,T_T, PX,
       .                   T,   a,  f , pi , p

      dimension          Z(laymax+1), K1(matmax), K3(matmax),
     .                   K1L(laymax), K3L(laymax), L(laymax), PX(5),
     .                   matnum(laymax),
     .                   a(neqmax,neqmax), f(neqmax), ipivot(neqmax)


       DO 5 I = 1, NLAYER
       K1L(I) = K1(MATNUM(I))
       K3L(I) = K3(MATNUM(I))
    5 continue

      pi  = 4.000 * datan(1.0d0)
      p   = pi / xlenth

       DO 7 I = 1, NLAYER
       L(I) = P * dsqrt(K1L(I)/K3L(I))
    7 continue

      neq = 2*nlayer

      write(ounit,1000)

C      LOOP FOR X = 1 TO  P_SOL
       DO 25 K = 1, P_SOL

       X = PX(K)


C --- Initialize the arrays.

      do 20 i = 1, neq
      f(i) = 0.000
      do 10 j = 1, neq
      a(i,j) = 0.000
   10 continue
   20 continue


       A(1,1) = 1
```

```
      A(1,2) = 1
      A(NEQ,NEQ-1) = dexp( - L(nlayer) * Z(nlayer))
      A(NEQ,NEQ)   = dexp( L(nlayer) * Z(nlayer))

      DO 30 I = 1, (nlayer-1)
            DO 40 J = 1, 4
      IF (J .EQ. 1) THEN
      II = I*2
     A(II,II-1)= dexp(- L(I) * Z(I))
      A(II+1,II-1)= -K3L(I)*L(I)*dexp( - L(I) * Z(I))
      ELSE IF (J .EQ. 2) THEN
      II = I*2
     A(II,II)= dexp( L(I) * Z(I))
      A(II+1,II)= K3L(I)*L(I)*dexp( L(I) * Z(I))
      ELSE IF (J .EQ. 3) THEN
      II = I*2
     A(II,II+1)= - dexp( - L(I+1) * Z(I))
      A(II+1,II+1)= K3L(I+1)*L(I+1)*dexp( - L(I+1) * Z(I))
      ELSE IF (J .EQ. 4) THEN
      II = I*2
     A(II,II+2)= - dexp( L(I+1) * Z(I))
      A(II+1,II+2)= - K3L(I+1)*L(I+1)*dexp( L(I+1) * Z(I))
      END IF

   40 continue
   30 continue

      F(1)   = T_B * DSIN( X * p )
      F(NEQ) = T_T * DSIN( X * p )


C --- Call a (incore) LINPACK subroutine to solve the equations.

      call genfac (a     , neqmax, neq   , ipivot, ierror)
      if (ierror .ne. 0) write(ounit,1010) ierror
      call genslv (a     , neqmax, neq   , ipivot, f     )


C --- Write the TEMPERATURE (at x=PX(K)) to the output file.
      write(ounit,1100) X
      write(ounit,1200)

      do 90 i = 1, nlayer-1
      IF (I .EQ. 1) THEN
      T = f(1)*dexp(- L(1) * Z(1)) + f(2)*dexp(L(1) * Z(1))
      ELSE
      J = I*2-1
      T = f(J)*dexp(- L(I) * Z(I)) + f(J+1)*dexp(L(I) * Z(I))
      END IF
      write(ounit,1300) i, Z(i), T
   90 continue

 1010 format(' ',/3x,'Error in factorization of coefficient matrix.',
     .           /3x,'Leading minor of order ',i4,
     .              ' is not positive definite.')
 1100 format(' ',//3x,'Temperatures at X=',E12.4)
 1200 format(' ',/7x,'Layer', 8x,   'Z', 15x,  'T',/)
 1300 format(' ',/8x,     i4, 8x, e10.5, 6x, e11.6)



   25 continue
      1000 format(' ',///3x,'SOLUTION')
```

```
      return
       end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C                           G E N S O L V                              C
C                                                                      C
C     THESE SUBROUTINES ARE MODIFIED VERSIONS OF LINPACK ROUTINES      C
C     USED TO SOLVE THE DOUBLE PRECISION GENERAL SYSTEM A * X = B.     C
C     THE ROUTINES "DGEFA" AND "DGESL" ARE RENAMED "GENFAC" AND        C
C     "GENSLV", RESPECTIVELY.  THE MODIFICATIONS TO THE SUBROUTINES    C
C     ARE PRIMARILY COSMETIC, MAKING THEM UNIFORM WITH OTHER PROGRAMS. C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  GENFAC                                              C
C     PURPOSE:     FACTOR A DOUBLE PRECISION MATRIX BY GAUSSIAN        C
C                  ELIMINATION.                                        C
C                                                                      C
C     ON ENTRY                                                         C
C                                                                      C
C        A       DOUBLE PRECISION(LDA,LDA)                             C
C                THE MATRIX TO BE FACTORED.                            C
C                                                                      C
C        LDA     INTEGER                                               C
C                THE LEADING DIMENSION OF THE ARRAY  A .               C
C                                                                      C
C        N       INTEGER                                               C
C                THE ORDER OF THE MATRIX  A .                          C
C                                                                      C
C     ON RETURN                                                        C
C                                                                      C
C        A       AN UPPER TRIANGULAR MATRIX AND THE MULTIPLIERS        C
C                WHICH WERE USED TO OBTAIN IT.                         C
C                THE FACTORIZATION CAN BE WRITTEN  A = L*U  WHERE      C
C                L  IS A PRODUCT OF PERMUTATION AND UNIT LOWER         C
C                TRIANGULAR MATRICES AND  U  IS UPPER TRIANGULAR.      C
C                                                                      C
C        IPVT    INTEGER(LDA)                                          C
C                AN INTEGER VECTOR OF PIVOT INDICES.                   C
C                                                                      C
C        INFO    INTEGER                                               C
C                = 0  NORMAL VALUE.                                    C
C                = K  IF  U(K,K) .EQ. 0.0 .  THIS IS NOT AN ERROR      C
C                     CONDITION FOR THIS SUBROUTINE, BUT IT DOES       C
C                     INDICATE THAT DGESL OR DGEDI WILL DIVIDE BY ZERO C
C                     IF CALLED.  USE  RCOND  IN DGECO FOR A RELIABLE  C
C                     INDICATION OF SINGULARITY.                       C
C                                                                      C
C     LINPACK. THIS VERSION DATED 08/14/78 .                           C
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LAB.     C
C                                                                      C
C     MODIFIED COSMETICALLY BY RON AVERILL, 03/14/91.                  C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE GENFAC (A     , LDA    , N      , IPVT   , INFO   )

      INTEGER            LDA   , N      , IPVT   ,INFO    ,
     .                   IDAMAX, J      , K      , KP1    , L      , NM1

      DOUBLE PRECISION   A     , T
```

```
      DIMENSION          A(LDA,LDA)    , IPVT(LDA)

C --- GAUSSIAN ELIMINATION WITH PARTIAL PIVOTING.

      INFO = 0
      NM1 = N - 1
      IF (NM1 .LT. 1) GO TO 70
      DO 60 K = 1, NM1
         KP1 = K + 1

C --- FIND L = PIVOT INDEX.

         L = IDAMAX(N-K+1,A(K,K),1) + K - 1
         IPVT(K) = L

C --- ZERO PIVOT IMPLIES THIS COLUMN ALREADY TRIANGULARIZED.

         IF (A(L,K) .EQ. 0.0D0) GO TO 40

C --- INTERCHANGE IF NECESSARY.

            IF (L .EQ. K) GO TO 10
               T = A(L,K)
               A(L,K) = A(K,K)
               A(K,K) = T
   10 CONTINUE

C ---  COMPUTE MULTIPLIERS.

            T = -1.0D0/A(K,K)
            CALL DSCAL(N-K,T,A(K+1,K),1)

C --- ROW ELIMINATION WITH COLUMN INDEXING.

            DO 30 J = KP1, N
               T = A(L,J)
               IF (L .EQ. K) GO TO 20
                  A(L,J) = A(K,J)
                  A(K,J) = T
   20          CONTINUE
               CALL DAXPY(N-K,T,A(K+1,K),1,A(K+1,J),1)
   30       CONTINUE
         GO TO 50
   40    CONTINUE
            INFO = K
   50    CONTINUE
   60 CONTINUE
   70 CONTINUE
      IPVT(N) = N
      IF (A(N,N) .EQ. 0.0D0) INFO = N

      RETURN

      END
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  GENSLV                                              C
C     PURPOSE:     SOLVE THE DOUBLE PRECISION SYSTEM A * X = B USING   C
C                  THE FACTORS COMPUTED BY GENFAC.                     C
C                                                                      C
C     ON ENTRY                                                         C
C                                                                      C
C        A       DOUBLE PRECISION(LDA, N)                              C
C                THE OUTPUT FROM GENFAC.                               C
C                                                                      C
C        LDA     INTEGER                                               C
C                THE LEADING DIMENSION OF THE ARRAY  A .               C
C                                                                      C
C        N       INTEGER                                               C
C                THE ORDER OF THE MATRIX  A .                          C
C                                                                      C
C        IPVT    INTEGER(N)                                            C
C                THE PIVOT VECTOR FROM GENFAC.                         C
C                                                                      C
C        B       DOUBLE PRECISION(N)                                   C
C                THE RIGHT HAND SIDE VECTOR.                           C
C                                                                      C
C        JOB     INTEGER                                               C
C                = 0         TO SOLVE  A*X = B ,                       C
C                = NONZERO   TO SOLVE  TRANS(A)*X = B  WHERE           C
C                            TRANS(A)  IS THE TRANSPOSE.               C
C                (HARDWIRED TO ZERO IN THE PROGRAM)                    C
C                                                                      C
C     ON RETURN                                                        C
C                                                                      C
C        B       THE SOLUTION VECTOR  X .                              C
C                                                                      C
C     ERROR CONDITION                                                  C
C                                                                      C
C        A DIVISION BY ZERO WILL OCCUR IF THE INPUT FACTOR CONTAINS A  C
C        ZERO ON THE DIAGONAL.  TECHNICALLY THIS INDICATES SINGULARITY C
C        BUT IT IS OFTEN CAUSED BY IMPROPER ARGUMENTS OR IMPROPER      C
C        SETTING OF LDA .                                              C
C                                                                      C
C     LINPACK. THIS VERSION DATED 08/14/78 .                           C
C     CLEVE MOLER, UNIVERSITY OF NEW MEXICO, ARGONNE NATIONAL LAB.     C
C                                                                      C
C     MODIFIED COSMETICALLY BY RON AVERILL, 03/14/91.                  C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE GENSLV (A     , LDA   , N     , IPVT  , B     )

      INTEGER             LDA   , N     , IPVT  , JOB,
     .                    K     , KB    , L     , NM1

      DOUBLE PRECISION    A     , B     , DDOT  , T

      DIMENSION           A(LDA,LDA)    , B(LDA), IPVT(LDA)

      JOB = 0
      NM1 = N - 1
      IF (JOB .NE. 0) GO TO 50

C --- IF JOB = 0 , SOLVE  A*X = B.
C --- FIRST SOLVE  L*Y = B.
```

```
      IF (NM1 .LT. 1) GO TO 30
      DO 20 K = 1, NM1
         L = IPVT(K)
         T = B(L)
         IF (L .EQ. K) GO TO 10
            B(L) = B(K)
            B(K) = T
   10    CONTINUE
         CALL DAXPY(N-K,T,A(K+1,K),1,B(K+1),1)
   20 CONTINUE
   30 CONTINUE

C --- NOW SOLVE  U*X = Y.

      DO 40 KB = 1, N
         K = N + 1 - KB
         B(K) = B(K)/A(K,K)
         T = -B(K)
         CALL DAXPY(K-1,T,A(1,K),1,B(1),1)
   40 CONTINUE
      GO TO 100
   50 CONTINUE

C --- IF JOB = NONZERO, SOLVE  TRANS(A) * X = B.
C --- FIRST SOLVE  TRANS(U)*Y = B.

      DO 60 K = 1, N
         T = DDOT(K-1,A(1,K),1,B(1),1)
         B(K) = (B(K) - T)/A(K,K)
   60 CONTINUE

C --- NOW SOLVE TRANS(L)*X = Y.

      IF (NM1 .LT. 1) GO TO 90
      DO 80 KB = 1, NM1
         K = N - KB
         B(K) = B(K) + DDOT(N-K,A(K+1,K),1,B(K+1),1)
         L = IPVT(K)
         IF (L .EQ. K) GO TO 70
            T = B(L)
            B(L) = B(K)
            B(K) = T
   70    CONTINUE
   80 CONTINUE
   90 CONTINUE
  100 CONTINUE
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  DAXPY                                               C
C     PURPOSE:     MULTIPLY A CONSTANT TIMES A VECTOR PLUS A VECTOR.   C
C                  USES UNROLLED LOOPS FOR INCREMENTS EQUAL TO ONE.    C
C     AUTHOR:      JACK DONGARRA, LINPACK, 3/11/78.                    C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE DAXPY  (N     , DA    , DX    , INCX  , DY    , INCY  )

      DOUBLE PRECISION   DX(1) , DY(1) , DA

      INTEGER            I     , INCX  , INCY  , M     , MP1   ,
```

```
     .                     N

      IF (N .LE. 0) RETURN
      IF (DA .EQ. 0.0D0) RETURN
      IF (INCX .EQ. 1 .AND. INCY .EQ. 1) GO TO 20

C --- CODE FOR UNEQUAL INCREMENTS OR EQUAL INCREMENTS NOT EQUAL TO 1.

      IX = 1
      IY = 1
      IF (INCX .LT. 0) IX = (-N+1)*INCX + 1
      IF (INCY .LT. 0) IY = (-N+1)*INCY + 1
      DO 10 I = 1,N
        DY(IY) = DY(IY) + DA*DX(IX)
        IX = IX + INCX
        IY = IY + INCY
   10 CONTINUE
      RETURN

C --- CODE FOR BOTH INCREMENTS EQUAL TO 1.
C --- CLEAN-UP LOOP.

   20 M = MOD(N,4)
      IF (M .EQ. 0) GO TO 40
      DO 30 I = 1,M
        DY(I) = DY(I) + DA*DX(I)
   30 CONTINUE
      IF (N .LT. 4) RETURN
   40 MP1 = M + 1
      DO 50 I = MP1,N,4
        DY(I) = DY(I) + DA*DX(I)
        DY(I + 1) = DY(I + 1) + DA*DX(I + 1)
        DY(I + 2) = DY(I + 2) + DA*DX(I + 2)
        DY(I + 3) = DY(I + 3) + DA*DX(I + 3)
   50 CONTINUE
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                       C
C     FUNCTION:  DDOT                                                   C
C     PURPOSE:   FORM THE DOT PRODUCT OF TWO VECTORS.                   C
C                USES UNROLLED LOOPS FOR INCREMENTS EQUAL TO ONE.       C
C     AUTHOR:    JACK DONGARRA, LINPACK, 3/11/78.                       C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      DOUBLE PRECISION FUNCTION DDOT    (N     , DX    , INCX  , DY    ,
     .                                   INCY  )

      DOUBLE PRECISION   DX(1) , DY(1) , DTEMP

      INTEGER            I     , INCX  , INCY  , IX    , IY    , M     ,
     .                   MP1   , N

      DDOT = 0.0D0
      DTEMP = 0.0D0
      IF (N .LE. 0) RETURN
      IF (INCX .EQ. 1 .AND. INCY .EQ. 1) GO TO 20

C --- CODE FOR UNEQUAL INCREMENTS OR EQUAL INCREMENTS NOT EQUAL TO 1.

      IX = 1
```

```
      IY = 1
      IF (INCX .LT. 0) IX = (-N+1)*INCX + 1
      IF (INCY .LT. 0) IY = (-N+1)*INCY + 1
      DO 10 I = 1,N
        DTEMP = DTEMP + DX(IX)*DY(IY)
        IX = IX + INCX
        IY = IY + INCY
   10 CONTINUE
      DDOT = DTEMP
      RETURN

C --- CODE FOR BOTH INCREMENTS EQUAL TO 1.
C --- CLEAN-UP LOOP.

   20 M = MOD(N,5)
      IF (M .EQ. 0) GO TO 40
      DO 30 I = 1,M
        DTEMP = DTEMP + DX(I)*DY(I)
   30 CONTINUE
      IF (N .LT. 5) GO TO 60
   40 MP1 = M + 1
      DO 50 I = MP1,N,5
        DTEMP = DTEMP + DX(I)*DY(I) + DX(I + 1)*DY(I + 1) +
     *   DX(I + 2)*DY(I + 2) + DX(I + 3)*DY(I + 3) + DX(I + 4)*DY(I + 4)
   50 CONTINUE
   60 DDOT = DTEMP
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  DSCAL                                               C
C     PURPOSE:     SCALE A VECTOR BY A CONSTANT.                       C
C                  USES UNROLLED LOOPS FOR INCREMENT EQUAL TO ONE.     C
C     AUTHOR:      JACK DONGARRA, LINPACK, 3/11/78.                    C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE DSCAL  (N     , DA    , DX    , INCX   )

      DOUBLE PRECISION   DA    , DX(1)

      INTEGER            I     , INCX  , M     , MP1   , N     , NINCX

      IF (N .LE. 0) RETURN
      IF (INCX .EQ. 1) GO TO 20

C --- CODE FOR INCREMENT NOT EQUAL TO 1.

      NINCX = N*INCX
      DO 10 I = 1,NINCX,INCX
        DX(I) = DA*DX(I)
   10 CONTINUE
      RETURN

C --- CODE FOR INCREMENT EQUAL TO 1.
C --- CLEAN-UP LOOP.

   20 M = MOD(N,5)
      IF (M .EQ. 0) GO TO 40
      DO 30 I = 1,M
        DX(I) = DA*DX(I)
   30 CONTINUE
```

```
      IF (N .LT. 5) RETURN
   40 MP1 = M + 1
      DO 50 I = MP1,N,5
        DX(I) = DA*DX(I)
        DX(I + 1) = DA*DX(I + 1)
        DX(I + 2) = DA*DX(I + 2)
        DX(I + 3) = DA*DX(I + 3)
        DX(I + 4) = DA*DX(I + 4)
   50 CONTINUE
      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     FUNCTION:  IDAMAX                                                C
C     PURPOSE:   FIND THE INDEX OF ELEMENT HAVING MAX. ABSOLUTE VALUE. C
C     AUTHOR:    JACK DONGARRA, LINPACK, 3/11/78.                      C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      INTEGER FUNCTION IDAMAX  ( N     , DX    , INCX  )

      DOUBLE PRECISION   DX(1) , DMAX

      INTEGER            I     , INCX  , IX    , N

      IDAMAX = 0
      IF (N .LT. 1) RETURN
      IDAMAX = 1
      IF (N .EQ. 1) RETURN
      IF (INCX .EQ. 1) GO TO 20

C --- CODE FOR INCREMENT NOT EQUAL TO 1.

      IX = 1
      DMAX = DABS(DX(1))
      IX = IX + INCX
      DO 10 I = 2,N
         IF (DABS(DX(IX)) .LE. DMAX) GO TO 5
         IDAMAX = I
         DMAX = DABS(DX(IX))
    5    IX = IX + INCX
   10 CONTINUE
      RETURN

C --- CODE FOR INCREMENT EQUAL TO 1.

   20 DMAX = DABS(DX(1))
      DO 30 I = 2,N
         IF (DABS(DX(I)) .LE. DMAX) GO TO 30
         IDAMAX = I
         DMAX = DABS(DX(I))
   30 CONTINUE
      RETURN
      END
```

APPENDIX C FORTRAN CODE, THERMNL

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C                        THERM 2 D - NON LINEAR                        C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C                             I N P U T                                C
C                                                                      C
C     >  TITLE                                                         C
C     >  PRTSTF, PRTLAY, INPLAY, PRTITR                                C
C     >  NLSTEP, ITRMAX, EPSILN                                        C
C     >  IMESH, DOM_X, DOM_Z, NELM_X, NELM_Z                           C
C                                                                      C
C     IF (IMESH = 0) THEN                                              C
C        DO LOOP: I = 1 TO NELMSH                                      C
C        >  NODE(I,J), J = 1 TO NODELM                                 C
C        END OF LOOP                                                   C
C        >  X(I), Z(I)  I = 1 TO NODMSH                                C
C     ELSE                                                             C
C        DO LOOP: I = 1 TO NELM_X                                      C
C        >  DX(I)                                                      C
C        END OF LOOP                                                   C
C        DO LOOP: I = 1 TO NELM_Z                                      C
C        >  DZ(I)                                                      C
C        END OF LOOP                                                   C
C     END IF                                                           C
C                                                                      C
C     >  NSGD    NSGF                                                  C
C     >  ISGD(I,J), J = 1 TO 2, I = 1 TO NSGD                          C
C     >  VSGD(I), I = 1 TO NSGD                                        C
C     >  ISGF(I,J), J = 1 TO 2, I = 1 TO NSGF                          C
C     >  VSGF(I), I = 1 TO NSGF                                        C
C                                                                      C
C     IF (INPLAY = 0) THEN                                             C
C      DO LOOP I=1, NELMSH                                             C
C     >  NLAYER(I)                                                     C
C         DO LOOP J=1, NLAYER(I)                                       C
C     >      KX(I,J), KZ(I,J), Z1LAY(I,J), Z2LAY(I,J)                  C
C        END OF LOOP                                                   C
C      END OF LOOP                                                     C
C     ELSE                                                             C
C      DO LOOP I=1, NELM_Z                                             C
C     >   NLATMP                                                       C
C         DO 70 K = 1, NELM_X                                          C
C          E = E + 1                                                   C
C          NLAYER(E) = NLATMP                                          C
C          DO LOOP J = 1, NLATMP                                       C
C     >    KX0TMP, KZ0TMP, LB1TMP, LB2TMP, Z1LTMP, Z2LTMP              C
C           KX0(E,J)     = KX0TMP                                      C
C           KZ0(E,J)     = KZ0TMP                                      C
C           LAMBD1(E,J) = LB1TMP                                       C
C           LAMBD2(E,J) = LB2TMP                                       C
C          Z1LAY(E,J)  = Z1LTMP                                        C
C          Z2LAY(E,J)  = Z2LTMP                                        C
C         END OF LOOP                                                  C
C        END OF LOOP                                                   C
C       END OF LOOP                                                    C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C                         V A R I A B L E S                            C
C                                                                      C
C     PARAMETERS:                                                      C
```

```
C                                                                       C
C       IUNIT  - INTEGER, UNIT NUMBER OF INPUT FILE.                    C
C       OUNIT  - INTEGER, UNIT NUMBER OF OUTPUT FILE.                   C
C                                                                       C
C       BNDMAX - INTEGER, COLUMN DIMENSION OF GLOBAL STIFFNESS:         C
C                GSTIF IS BANDED IF IMASS = 0,                          C
C                GSTIF IS SQUARE IF IMASS = 1.                          C
C       EDFMAX - INTEGER, MAXIMUM NUMBER OF D.O.F. PER ELEMENT.         C
C       NBCMAX - INTEGER, MAXIMUM NUMBER OF BOUNDARY CONDITIONS.        C
C       NDFMAX - INTEGER, MAXIMUM NUMBER OF D.O.F. PER NODE.            C
C       NEMMAX - INTEGER, MAXIMUM NUMBER OF ELEMENTS IN THE MESH.       C
C       NEQMAX - INTEGER, MAXIMUM NUMBER OF EQUATIONS (OR MAXIMUM       C
C                NUMBER OF D.O.F. IN THE MESH).                         C
C       NGPMAX - INTEGER, MAXIMUM NUMBER OF GAUSS POINTS.               C
C       NNMMAX - INTEGER, MAXIMUM NUMBER OF NODES IN THE MESH.          C
C       NPEMAX - INTEGER, MAXIMUM NUMBER OF NODES PER ELEMENT.          C
C       NLAMAX - INTEGER, MAXIMUM NUMBER OF LAYERS PER ELEMENT.         C
C                                                                       C
C    CONTROL VARIABLES:                                                 C
C                                                                       C
C       PRTSTF - INTEGER, FLAG FOR PRINTING STIFFNESS MATRICES:         C
C                PRTSTF = 0 FOR NO PRINT,                               C
C                PRTSTF = 1 FOR PRINTING STIFFNESS OF ELEMENT ONE,      C
C                PRTSTF = 2 FOR PRINTING GLOBAL STIFFNESS.              C
C       PRTLAY - INTEGER, FLAG FOR PRINTING LAYERS PROPERTY:            C
C                PRTSTF = 0 FOR NO PRINT,                               C
C                PRTSTF = 1 FOR PRINTING.                               C
C       PRTITR - INTEGER, FLAG FOR PRINTING SOLUTIONS AT EACH ITERATION
C                         AT THE CENTRAL NODE OF THE MESH:              C
C                PRTITR = 0 FOR NO PRINT,                               C
C                PRTITR = 1 FOR PRINTING.                               C
C       NLSTEP - INTEGER, NUMBER OF LOAD STEP.                          C
C       DLAM   - DOUBLE PRECISION, LOAD INCREMENT SCALE.                C
C       ITRMAX - INTEGER, MAXIMUM NUMBER OF ITERATIONS.                 C
C       FLAG_D - INTEGER, FLAG FOR DISPLACEMENT OR FORCE B.C.'S:        C
C                FLAG_D = 0 FOR MIXED FORCE AND DISPLACEMENT B.C.'S     C
C                FLAG_D = 1 FOR ONLY DISPLACEMENT B.C.'S                C
C       EPSILN - DOUBLE PRECISION, TOLLERANCE.                          C
C                                                                       C
C    ELEMENT PROPERTIES:                                                C
C                                                                       C
C       NDFELM - INTEGER, NUMBER OF D.O.F. PER ELEMENT.                 C
C       NODELM - INTEGER, NUMBER OF NODES PER ELEMENT.                  C
C       NLAYER - INTEGER, NUMBER OF LAYERS PER ELEMENT.                 C
C                                                                       C
C    MESH:                                                              C
C                                                                       C
C       BNDWTH - INTEGER, HALF BANDWIDTH OF GLOBAL STIFFNESS MATRIX     C
C                FOR BENDING PROBLEMS.                                  C
C       EX     - DOUBLE PRECISION(NPEMAX), EX(I) IS THE X-COORDINATE    C
C                OF THE I-TH NODE OF THE ELEMENT UNDER CONSIDERATION.   C
C       IMESH  - INTEGER, FLAG TO INDICATE HOW MESH WILL BE INPUT:      C
C                IMESH = 0 IF THE MESH IS INPUT BY HAND,                C
C                IMESH = 1 IF THE MESH IS COMPUTED BY THE PROGRAM.      C
C       INPLAY - INTEGER, FLAG TO INDICATE HOW LAYERS WILL BE INPUT:    C
C                IMESH = 0 LAYERS ARE INPUT ELEMENT BY ELEMENT,         C
C                IMESH = 1 LAYERS ARE INPUT BY GROUP OF HORIZZONTAL     C
C                          ELEMENTS.                                    C
C       NDFMSH - INTEGER, NUMBER OF D.O.F. IN THE MESH.                 C
C       NDFNOD - INTEGER, NUMBER OF D.O.F. PER NODE.                    C
C       NELMSH - INTEGER, NUMBER OF ELEMENTS IN THE MESH.               C
C       NODE   - INTEGER(NEMMAX,NPEMAX), CONNECTIVITY MATRIX:           C
C                NODE(I,J) IS THE J-TH NODE OF THE I-TH ELEMENT.        C
```

```
C       NODMSH - INTEGER, NUMBER OF NODES IN THE MESH.                  C
C       X      - DOUBLE PRECISION(NNMMAX), X(I) IS THE GLOBAL           C
C                COORDINATE OF THE I-TH NODE.                           C
C       NELM_X - INTEGER, NUMBER OF ELEMENTS IN THE MESH IN X DIR.      C
C       NELM_Z - INTEGER, NUMBER OF ELEMENTS IN THE MESH IN Z DIR.      C
C       DOM_X  - DOUBLE PRECISION, DOMAIN IN X DIRECTION                C
C       DOM_Z  - DOUBLE PRECISION, DOMAIN IN Z DIRECTION                C
C       DX     - DOUBLE PRECISION, DIMENSION OF ELEMENTS                C
C                IN X DIRECTION STARTING FROM THE LEFT                  C
C       DZ     - DOUBLE PRECISION, DIMENSION OF ELEMENTS                C
C                IN Z DIRECTION STARTING FROM THE BOTTOM                C
C       KX     - DOUBLE PRECISION, TERMAL CONDUCTIVITY COEFFICIENT OF C
C                THE LAYER ALONG X DIRECTION                            C
C       KZ     - DOUBLE PRECISION, TERMAL CONDUCTIVITY COEFFICIENT OF C
C                THE LAYER ALONG Z DIRECTION                            C
C       Z1LAY  - DOUBLE PRECISION, Z COORDINATE OF THE BOTTOM OF THE  C
C                LAYER                                                  C
C       Z2LAY  - DOUBLE PRECISION, Z COORDINATE OF THE TOP OF THE      C
C                LAYER                                                  C
C                                                                       C
C     BOUNDARY CONDITIONS:                                              C
C                                                                       C
C       ISGD   - INTEGER(NBCMAX,2), ISGD(I,J) INDICATES THAT THE J-TH C
C                DISPLACEMENT D.O.F. OF THE I-TH NODE IS SPECIFIED.    C
C       ISGF   - INTEGER(NBCMAX,2), ISGF(I,J) INDICATES THAT THE J-TH C
C                FORCE COMPONENT OF THE I-TH NODE IS SPECIFIED.        C
C       NSGD   - INTEGER, NUMBER OF DISPLACEMENT BOUNDARY CONDITIONS. C
C       NSGF   - INTEGER, NUMBER OF NODAL FORCE BOUNDARY CONDITIONS.  C
C       VSGD   - DOUBLE PRECISION(NBCMAX), VSGD(I) IS THE SPECIFIED   C
C                VALUE OF THE I-TH DISPLACEMENT BOUNDARY CONDITION.   C
C       VSGF   - DOUBLE PRECISION(NBCMAX), VSGF(I) IS THE SPECIFIED   C
C                VALUE OF THE I-TH NODAL FORCE BOUNDARY CONDITION.    C
C                                                                       C
C     FINITE ELEMENT MODEL:                                             C
C                                                                       C
C       DSF    - DOUBLE PRECISION(NPEMAX), DSF(I) IS THE LOCAL (XI)   C
C                DERIVATIVE OF THE I-TH SHAPE FUNCTION.                 C
C       EF     - DOUBLE PRECISION(EDFMAX), ELEMENT FORCE VECTOR.        C
C       ESTIF  - DOUBLE PRECISION(EDFMAX,EDFMAX), ELEMENT STIFFNESS.   C
C       EU     - DOUBLE PRECISION(EDFMAX), ELEMENT EXTENSION VECTOR.   C
C       GAUSPT - DOUBLE PRECISION(NGPMAX,NGPMAX) GAUSPT(I,J) IS THE    C
C                I-TH GAUSS POINT OF J-TH GAUSS RULE.                   C
C       GAUSWT - DOUBLE PRECISION(NGPMAX,NGPMAX) GAUSWT(I,J) IS THE    C
C                I-TH GAUSS WEIGHT OF J-TH GAUSS RULE.                  C
C       GDSF   - DOUBLE PRECISION(NPEMAX), GDSF(I) IS THE GLOBAL (X)  C
C                DERIVATIVE OF THE I-TH SHAPE FUNCTION.                 C
C       GF     - DOUBLE PRECISION(NEQMAX), GLOBAL FORCE VECTOR,         C
C                WHICH, ON RETURN, CONTAINS THE SOLUTION VECTOR;        C
C                ALSO USED AS A DUMMY VECTOR IN EIGENPROBLEMS.          C
C       GSTIF  - DOUBLE PRECISION(NEQMAX,BNDMAX), GLOBAL STIFFNESS.    C
C       NGPSTF - INTEGER, NUMBER OF GAUSS POINTS TO BE USED TO         C
C                INTEGRATE THE STIFFNESS MATRIX.                        C
C       NGPGRD - INTEGER, NUMBER OF GAUSS POINTS AT WHICH THE          C
C                GRADIENTS ARE TO BE EVALUATED.                         C
C       SF     - INTEGER(NPEMAX), SF(I) IS THE I-TH SHAPE FUNCTION.    C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      IMPLICIT LOGICAL (A-Z)

      INTEGER            IUNIT , OUNIT , NEMMAX, NPEMAX, NDFMAX, NGPMAX,
     .                   NNMMAX, NBCMAX, NEQMAX, EDFMAX, BNDMAX,
     .                   NELM_X, NELM_Z, NLAMAX,
```

```
     .                  NLSTEP, ITRMAX, FLAG_D

      PARAMETER        (IUNIT  = 10, OUNIT  = 12, NEMMAX = 400,
     .                  NPEMAX =  4, NDFMAX =  1, NGPMAX =  5,
     .                  BNDMAX = 400, NLAMAX = 40,
     .                  EDFMAX = NDFMAX * NPEMAX,
     .                  NNMMAX = NEMMAX * (NPEMAX - 1) + 1,
     .                  NEQMAX = NNMMAX * NDFMAX,
     .                  NBCMAX = NEMMAX*4)


      INTEGER           PRTSTF, PRTGRD, nodelm, ndfelm, ngpstf, ngpgrd,
     .                  nelmsh, ndfmsh, nodmsh, ndfnod, node  , bndwth,
     .                  nsgd  , nsgf  , isgd  , isgf, PRTLAY, INPLAY,
     .                  NLAYER, FLAGNR, PRTITR


      DOUBLE PRECISION  X     ,  vsgd, vsgf  , ex    , DOM_X, DOM_Z,
     .                  GAUSPT, GAUSWT, ESTIF , EF    , GSTIF ,
     .                  GF    , DSF   , GDSF  ,
     .                  Z     , KX    , KZ,    Z1LAY,    Z2LAY, EZ,
     .                   AK    , FI_D  , D_FI_N, D_FI,   SF2, GUL,
     .                  LAMBD1, LAMBD2, DLAM, EPSILN, GU, GUPLS,
     .                  KX0   , KZ0, T_LAY


      DIMENSION  NODE(NEMMAX,NPEMAX), ISGD(NBCMAX,2),
     .           ISGF(NBCMAX,2), X(NNMMAX)          , EX(NPEMAX)     ,
     .           VSGD(NBCMAX)   , VSGF(NBCMAX)       ,
     .           EF(EDFMAX)     , GF(NEQMAX)         ,
     .           DSF(NPEMAX)        , GDSF(NPEMAX)   ,
     .           GAUSPT(NGPMAX,NGPMAX), GAUSWT(NGPMAX,NGPMAX),
     .           ESTIF(EDFMAX,EDFMAX) ,
     .           GSTIF(NEQMAX,BNDMAX),
     .           Z(NNMMAX), NLAYER(NEMMAX),
     .           KX(nemmax,NLAMAX), KZ(nemmax,NLAMAX),
     .           Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .           EZ(NPEMAX), AK(nemmax,NLAMAX), FI_D(nemmax),
     .           D_FI_N(nemmax,NLAMAX), D_FI(nemmax,NLAMAX),
     .             SF2(2,2),GUL(nemmax,2,NLAMAX),
     .           LAMBD1(nemmax,NLAMAX), LAMBD2(nemmax,NLAMAX),
     .           GU(NEQMAX), GUPLS(NEQMAX), T_LAY(nemmax,NLAMAX),
     .             KX0(nemmax,NLAMAX), KZ0(nemmax,NLAMAX)


       open(unit=10,file='TERMNL.data',status='old')
      open(unit=12,file='TERMNL.out', status='unknown')

C --------------------
C --- PREPROCESSOR ---
C --------------------

C --- INPUT OPTIONS.

      CALL INPOPT (IUNIT, OUNIT, PRTSTF, PRTGRD, PRTLAY, INPLAY, PRTITR)

C --- INPUT ELEMENT DATA.
```

```
      call INPELM (OUNIT , NODELM, NDFELM, NGPSTF, NGPGRD,
     .             NDFNOD)

C --- INPUT NON LINEAR ANALYSIS DATA

      call INP_NL (IUNIT , OUNIT , NLSTEP, DLAM, ITRMAX, EPSILN)


C --- INPUT MESH PARAMETERS AND FORM THE MESH.

      call INPMSH (IUNIT , OUNIT , NEMMAX, NPEMAX, NNMMAX, NODELM,
     .             NDFNOD, DOM_X, DOM_Z, NELM_X, NELM_Z,
     .             NELMSH, NDFMSH, NODMSH, NODE, BNDWTH, X, Z )

C --- INPUT NODAL BOUNDARY CONDITIONS.

      call INPNBC (IUNIT , OUNIT , NBCMAX, NNMMAX, NODMSH, NDFNOD,
     .             NSGD  , NSGF  , ISGD  , ISGF  , X    , Z, VSGD,
     .             VSGF, FLAG_D)

C --- INPUT ELEMENT LAYERS

      call INLAYR (IUNIT , OUNIT , NELMSH, NLAYER, NLAMAX, NEMMAX,
      .             KX0, KZ0, Z1LAY, Z2LAY, LAMBD1, LAMBD2, PRTLAY,
     .             NELM_X, NELM_Z, INPLAY)


C --- INITIALIZE THE ARRAYS WHICH CONTAIN THE GAUSS POINTS AND WEIGHTS.

      CALL GQUAD  (NGPMAX, GAUSPT, GAUSWT)

C ---------------
C --- PROCESSOR ---
C ---------------

       IF (FLAG_D .EQ. 0) THEN

        CALL PROCSR_F (OUNIT , NEMMAX, NPEMAX, NGPMAX, NNMMAX,
      .                    NEQMAX, BNDMAX, NBCMAX, EDFMAX,
     .                   ndfnod, ndfelm, nodelm,
     .                   ndfmsh, nelmsh, node  , ngpstf, prtstf, bndwth,
     .                   nsgd  , isgd  , nsgf  , isgf  , X     , EX,
     .                   GAUSPT, GAUSWT, dsf   , GDSF  , ESTIF,
     .                   EF    , GSTIF , gf    , vsgd  , VSGF  ,
     .                   Z     , EZ, KX   , KZ,   Z1LAY,  Z2LAY,
     .                   NLAMAX, NLAYER,AK, FI_D, D_FI_N, D_FI, SF2,
     .                   NLSTEP, DLAM, ITRMAX, EPSILN,
     .                   GU, GUPLS, FLAGNR, KX0, KZ0, T_LAY,
     .                   NELM_X, NELM_Z, LAMBD1, LAMBD2, PRTITR)

       ELSE

        CALL PROCSR_D (OUNIT , NEMMAX, NPEMAX, NGPMAX, NNMMAX,
      .                   NEQMAX, BNDMAX, NBCMAX, EDFMAX,
     .                   ndfnod, ndfelm, nodelm,
     .                   ndfmsh, nelmsh, node  , ngpstf, prtstf, bndwth,
     .                   nsgd  , isgd  , X     , EX,
     .                   GAUSPT, GAUSWT, dsf   , GDSF  , ESTIF,
     .                   EF    , GSTIF , gf    , vsgd  ,
     .                   Z     , EZ, KX   , KZ,   Z1LAY,  Z2LAY,
     .                   NLAMAX, NLAYER,AK, FI_D, D_FI_N, D_FI, SF2,
     .                   NLSTEP, DLAM, ITRMAX, EPSILN,
     .                   GU, GUPLS, FLAGNR, KX0, KZ0, T_LAY,
```

```
     .                  NELM_X, NELM_Z, LAMBD1, LAMBD2, PRTITR)

       END IF

C ---------------------
C --- POSTPROCESSOR ---
C ---------------------


      CAll PRTGU        (OUNIT , NEQMAX, NNMMAX, NDFNOD, NODMSH, GU    ,
     .                  X, Z  )


      CAll PR_L_T      (OUNIT,  NPEMAX, NLAMAX, NELMSH,
     .                 NEMMAX, NLAYER, NEQMAX,
     .                 Z1LAY, Z2LAY, AK, FI_D, NODE, GUL, GU, T_LAY)


      STOP

       END



C-----------------------------------------------------------------------C
C                                                                       C
C                            SUBROUTINES                                C
C                                                                       C
C-----------------------------------------------------------------------C



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                       C
C     SUBROUTINE:  Procsr_F                                             C
C     PURPOSE:     DRIVER ROUTINE TO SOLVE THE EQUATIONS.               C
C                  FORCE STEPPING                                       C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE PROCSR_F (OUNIT , NEMMAX, NPEMAX, NGPMAX, NNMMAX,
     .                     NEQMAX, BNDMAX, NBCMAX, EDFMAX,
     .                  ndfnod, ndfelm, nodelm,
     .                  ndfmsh, nelmsh, node  , ngpstf, prtstf, bndwth,
     .                  nsgd  , isgd  , nsgf  , isgf  , X     , EX,
     .                  GAUSPT, GAUSWT, dsf   , GDSF  , ESTIF,
     .                  EF    , GSTIF , gf    , vsgd  , VSGF  ,
     .                  Z     , EZ, KX    , KZ,   Z1LAY,  Z2LAY,
     .                  NLAMAX, NLAYER,AK, FI_D, D_FI_N, D_FI, SF2,
     .                  NLSTEP, DLAM, ITRMAX, EPSILN,
     .                  GU, GUPLS, FLAGNR, KX0, KZ0, T_LAY,
     .                  NELM_X, NELM_Z, LAMBD1, LAMBD2, PRTITR)

      IMPLICIT LOGICAL (A-Z)

      integer           OUNIT , NEMMAX, NPEMAX, NGPMAX, NNMMAX, NEQMAX,
     .                  BNDMAX, NBCMAX, EDFMAX, ndfnod, ndfelm, nodelm,
     .                  ndfmsh, nelmsh, node  , ngpstf, prtstf, bndwth,
     .                  nsgd  , isgd  , nsgf  , isgf  ,      I,      J,
     .                  ELMNUM, ENOD  , GNOD  , IRES  , NLAMAX, NLAYER,
     .                  NLSTEP, ITRMAX, ILSTEP, ICNVRG, PRTITR,
     .                  FLAGNR, ITER, F_K0, NELM_X, NELM_Z, CEN, Flag_P
```

```
      double precision  X      , EX     , GAUSPT, GAUSWT, dsf   ,
     .                 GDSF   , ESTIF , EF     , GSTIF , gf     , vsgd  ,
     .                 VSGF   , Z,     EZ, KX     , KZ,    Z1LAY,  Z2LAY,
     .                 AK,    FI_D,  D_FI_N, D_FI, SF2,
     .                 LAMBDA, LAMBD1, LAMBD2, DLAM, EPSILN, GU,GUPLS,
     .                 ERROR, KX0, KZ0, T_LAY, IVSGD, GREF,
     .                 GF2, GU2, UINC, UOLD, DELTA_D



      DIMENSION ISGD(NBCMAX,2), ISGF(NBCMAX,2),
     .          X(NNMMAX)      , EX(NPEMAX)     , EF(EDFMAX)     ,
     .          DSF(NPEMAX)    , GDSF(NPEMAX)   ,
     .          VSGD(NBCMAX)   , VSGF(NBCMAX)   , GF(NEQMAX)     ,
     .          GAUSPT(NGPMAX,NGPMAX)   , GAUSWT(NGPMAX,NGPMAX),
     .          ESTIF(EDFMAX,EDFMAX)    ,
     .          GSTIF(NEQMAX,BNDMAX)    , NODE(NEMMAX,NPEMAX),
     .         Z(NNMMAX)      , EZ(NPEMAX), NLAYER(NEMMAX),
     .          KX(nemmax,NLAMAX), KZ(nemmax,NLAMAX),
     .          Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .          AK(nemmax,NLAMAX), FI_D(nemmax), D_FI_N(nemmax,NLAMAX),
     .          D_FI(nemmax,NLAMAX), SF2(2,2),
     .          LAMBD1(nemmax,NLAMAX), LAMBD2(nemmax,NLAMAX),
     .          GU(NEQMAX), GUPLS(NEQMAX), T_LAY(nemmax,NLAMAX),
     .            KX0(nemmax,NLAMAX), KZ0(nemmax,NLAMAX),
     .          IVSGD(NBCMAX), GREF(NEQMAX),
     .          DELTA_D(NEQMAX)


C ----INITIALIZE LOAD SCALE

      LAMBDA = 0

C ----INITIALIZE SOLUTION VECTOR

      DO 5  I = 1, NEQMAX
            GU(I) = 0.000
         GUPLS(I) = 0.000
    5 CONTINUE

      DO 43 I = 1, NBCMAX
          IVSGD(I) = VSGD(I)
   43 CONTINUE


C-----COMPUTE REFERENCE LOAD

       IF (NSGF .GT. 0) THEN
         CALL FBNDRY (NEQMAX, NBCMAX, NDFNOD, NSGF, ISGF, VSGF, GF)
      END IF

       DO 57 I = 1, NEQMAX
          GREF(I) = GF(I)
   57  CONTINUE

       DO 8 I = 1, NDFMSH
          GF(I) = 0.000
    8  CONTINUE
```

```
C ----LOOP ON N. OF LOAD STEP

      DO 500 ILSTEP = 1, NLSTEP

C ---- INITIALIZE COUNTERS AND FLAG

        ICNVRG = 0
        ITER   = 0
        ERROR  = 1.0
        FLAGNR = 0


C ---- ITERATION LOOP

    1  ITER = ITER + 1


C ---- CHECK FOR ITERMAX

        IF (ITER .GT. ITRMAX) STOP

C ---  INITIALIZE THE GLOBAL STIFFNESS ARRAY AND FORCE VECTOR.

      DO 88 I = 1, NDFMSH
          GF(I) = 0.000
   88  CONTINUE

        IF (FLAGNR .EQ. 0) THEN
       DO 21 I = 1, NDFMSH
         DO 11 J = 1, BNDWTH
            GSTIF(I,J) = 0.000
   11    CONTINUE
   21  CONTINUE
       END IF

C ---  VERIFY IF WE ARE AT THE FIRST ITERATION OF THE FIRST LOAD STEP.

       F_K0 = 0
       IF (ILSTEP .EQ. 1  .AND.  ITER .EQ. 1) THEN
       F_K0 = 1
       END IF


      CALL CALC_T (NPEMAX, NLAMAX, NELMSH, NEMMAX,
     .             NLAYER, NEQMAX, KX, KZ, Z1LAY,  Z2LAY, AK,
     .             FI_D, NODE, GU, T_LAY, KX0, KZ0, F_K0,
     .             LAMBD1, LAMBD2)

       CALL CR_FI(NLAMAX, NELMSH, NEMMAX,
     .           NLAYER, KZ, Z1LAY, Z2LAY,
     .           AK, FI_D, D_FI_N, D_FI)


C --- BEGIN DO-LOOP ON NUMBER OF ELEMENTS TO CALCULATE THE ELEMENT
C --- MATRICES AND TO ASSEMBLE THE GLOBAL MATRICES.

      DO 50 ELMNUM = 1, NELMSH


C --- DETERMINE THE LOCAL (ELEMENT) DATA.

      DO 30 ENOD = 1, NODELM
```

```
         GNOD = NODE(ELMNUM,ENOD)
         EX(ENOD) = X(GNOD)
          EZ(ENOD) = Z(GNOD)
   30 CONTINUE


      CALL STIFF_T       (EDFMAX, NPEMAX, NGPMAX, ndfelm,
     .                    NGPstf, ELMNUM,
     .                    DSF, GDSF, EX, ESTIF, EF, GAUSPT, GAUSWT,
     .                    EZ, KX, KZ, Z1LAY, Z2LAY, NEMMAX,
     .                    NLAMAX, NLAYER, AK, FI_D, D_FI, SF2)


C --- PRINT STIFFNESS MATRIX AND FORCE VECTOR FOR ELEMENT ONE.

      IF (PRTSTF .EQ. 1 .AND. ELMNUM .EQ. 1 .AND. F_K0 .EQ. 1) THEN
         WRITE(OUNIT,1000)
         DO 40 I = 1, NDFELM
            WRITE(OUNIT,1010) I
            WRITE(OUNIT,1020) (ESTIF(I,J), J = 1, NDFELM)
   40    CONTINUE
         WRITE(OUNIT,1030)
         WRITE(OUNIT,1020) (EF(I), I = 1, NDFELM)
      END IF


C --- ASSEMBLE THE GLOBAL MATRICES IN BANDED FORM.

      CALL ASMBLE     (NEQMAX, BNDMAX, EDFMAX, NEMMAX, NPEMAX, NODELM,
     .                 NDFNOD, NODE  , ESTIF , EF    , GSTIF , GF    ,
     .                 ELMNUM)


C --- END OF LOOP TO COMPUTE GLOBAL STIFFNESS AND FORCE MATRICES.

   50 CONTINUE


C ----COMPUTE RESIDUAL LOAD VECTOR

       DO 14 I = 1, NEQMAX
          GF(I) = (LAMBDA + DLAM) * GREF(I)
   14  CONTINUE

C --- PRINT GLOBAL STIFFNESS MATRIX AND FORCE VECTOR. --

C --- IMPOSE PRIMARY BOUNDARY CONDITIONS.

      CALL UBNDRY     (NEQMAX, BNDMAX, NBCMAX, NDFNOD, NDFMSH, BNDWTH,
     .                 NSGD  , ISGD  , VSGD  , GSTIF , GF    )


C --- SOLVE THE SYSTEM OF EQUATIONS.

      IRES = 0
      CALL SOLVE (NEQMAX, BNDMAX, NDFMSH, BNDWTH, GSTIF, GF, IRES)

C --- COMPUTE THE INCREMENT.
       DO 113 I = 1, NEQMAX
           DELTA_D(I) = GF(I) - GU(I)
  113  CONTINUE
```

```
C-----CHECK FOR CONVERGENGE

       GF2 = 0.0000
       GU2 = 0.0000
       DO 63 I = 1, NEQMAX
          GF2 = GF2 + DELTA_D(I)*DELTA_D(I)
           GU2 = GU2 + GU(I)*GU(I)
   63  CONTINUE

      IF (ITER .GT. 1) THEN
      UINC = GF2
       UOLD = GU2
         ERROR = DSQRT(UINC/UOLD)
         IF (ERROR .LT. EPSILN) THEN
         ICNVRG = 1
         END IF
      END IF


C-----UPDATE SOLUTION

        DO 73 I = 1, NEQMAX
           GU(I) = GF(I)
   73  CONTINUE

       IF (ICNVRG .EQ. 1) THEN
       DO 83 I = 1, NEQMAX
           GUPLS(I) = GU(I)
   83  CONTINUE
        END IF

C-----PRINT VALUE OF T AT THE CENTER OF THE MESH OR NEAR

       IF  (Flag_P .EQ. 0) THEN
       WRITE(OUNIT,2000)
      WRITE(OUNIT,2030)
       end if
       Flag_P = 1

       IF  (PRTITR .EQ. 1 .and. ICNVRG .EQ. 1 ) THEN
        CEN  = (NELM_X+1)*(NELM_Z/2) + (NELM_X/2+1)
       WRITE(OUNIT,2040) GU(CEN)
 2000  FORMAT(' ',/3X,'SOLUTION AT THE CENTER OF THE MESH: ')
 2030  FORMAT(' ',8X,'T (CENTER)')
 2040  FORMAT(' ',5X,    E14.8,6X)

       END IF



C ----INCREMENT LOAD

       IF (ICNVRG .EQ. 1) THEN


C---- CONVERGENCE OBTAINED, NEXT LOAD STEP

       LAMBDA = LAMBDA + DLAM
         GOTO 500

       ELSE
```

```
C---- CONVERGENCE NOT OBTAINED, NEXT ITERATION

         GOTO 1

      END IF

  500 CONTINUE



 1000 FORMAT(' ',//3X,'STIFFNESS MATRIX FOR ELEMENT ONE: ')
 1010 FORMAT(' ',/5X,'ROW:  ',I2,/)
 1020 FORMAT(' ',5X,6E12.4)
 1030 FORMAT(' ',//3X,'FORCE VECTOR FOR ELEMENT ONE: ',/)
 1040 FORMAT(' ',//3X,'GLOBAL STIFFNESS MATRIX: ',/)
 1050 FORMAT(' ',//3X,'GLOBAL FORCE VECTOR: ',/)
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                       C
C     SUBROUTINE:  Procsr_D                                             C
C     PURPOSE:     DRIVER ROUTINE TO SOLVE THE EQUATIONS.               C
C                  DISPLACEMENT STEPPING                                C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE PROCSR_D (OUNIT , NEMMAX, NPEMAX, NGPMAX, NNMMAX,
     .                       NEQMAX, BNDMAX, NBCMAX, EDFMAX,
     .                   ndfnod, ndfelm, nodelm,
     .                   ndfmsh, nelmsh, node  , ngpstf, prtstf, bndwth,
     .                   nsgd  , isgd  , X     , EX,
     .                   GAUSPT, GAUSWT, dsf   , GDSF  , ESTIF,
     .                   EF    , GSTIF , gf    , vsgd  ,
     .                   Z     , EZ, KX   , KZ,   Z1LAY,  Z2LAY,
     .                   NLAMAX, NLAYER,AK, FI_D, D_FI_N, D_FI, SF2,
     .                   NLSTEP, DLAM, ITRMAX, EPSILN,
     .                   GU, GUPLS, FLAGNR, KX0, KZ0, T_LAY,
     .                   NELM_X, NELM_Z, LAMBD1, LAMBD2, PRTITR)


      IMPLICIT LOGICAL (A-Z)

      integer            OUNIT , NEMMAX, NPEMAX, NGPMAX, NNMMAX, NEQMAX,
     .                   BNDMAX, NBCMAX, EDFMAX, ndfnod, ndfelm, nodelm,
     .                   ndfmsh, nelmsh, node  , ngpstf, prtstf, bndwth,
     .                   nsgd  , isgd  ,      I,      J,
     .                   ELMNUM, ENOD  , GNOD  , IRES  , NLAMAX, NLAYER,
     .                   NLSTEP, ITRMAX, ILSTEP, ICNVRG, PRTITR,
     .                   FLAGNR, ITER, F_K0, NELM_X, NELM_Z, CEN, Flag_P


        double precision  X     , EX    , GAUSPT, GAUSWT, dsf  ,
     .                   GDSF  , ESTIF , EF    , GSTIF , gf    , vsgd  ,
     .                   Z,    EZ, KX    , KZ,    Z1LAY,  Z2LAY,
     .                   AK,   FI_D,  D_FI_N, D_FI, SF2, DELTA_D, ERROR,
     .                           LAMBDA, LAMBD1, LAMBD2, DLAM, EPSILN,
GU,GUPLS,
```

```
     .                   KX0, KZ0, T_LAY, IVSGD,
     .                   UINC, UOLD, GF2, GU2


      DIMENSION ISGD(NBCMAX,2),
     .          X(NNMMAX)      , EX(NPEMAX)     , EF(EDFMAX)     ,
     .          DSF(NPEMAX)    , GDSF(NPEMAX)   ,
     .          VSGD(NBCMAX)   , GF(NEQMAX)     ,
     .          GAUSPT(NGPMAX,NGPMAX)    , GAUSWT(NGPMAX,NGPMAX),
     .          ESTIF(EDFMAX,EDFMAX)     ,
     .          GSTIF(NEQMAX,BNDMAX)     , NODE(NEMMAX,NPEMAX),
     .          Z(NNMMAX)      , EZ(NPEMAX), NLAYER(NEMMAX),
     .          KX(nemmax,NLAMAX), KZ(nemmax,NLAMAX),
     .          Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .          AK(nemmax,NLAMAX), FI_D(nemmax), D_FI_N(nemmax,NLAMAX),
     .          D_FI(nemmax,NLAMAX), SF2(2,2),
     .          LAMBD1(nemmax,NLAMAX), LAMBD2(nemmax,NLAMAX),
     .          GU(NEQMAX), GUPLS(NEQMAX), T_LAY(nemmax,NLAMAX),
     .            KX0(nemmax,NLAMAX), KZ0(nemmax,NLAMAX),
     .          IVSGD(NBCMAX), DELTA_D(NEQMAX)


C ----INITIALIZE LOAD SCALE

      LAMBDA = 0


C ----INITIALIZE SOLUTION VECTOR

      DO 5  I = 1, NEQMAX
            GU(I) = 0.000
         GUPLS(I) = 0.000
    5 CONTINUE

      DO 88 I = 1, NEQMAX
          GF(I) = 0.000
   88  CONTINUE


C ----EBC'S COPIED IN VECTOR IVSGD

        DO 43 I = 1, NBCMAX
          IVSGD(I) = VSGD(I)
   43  CONTINUE


C ----LOOP ON N. OF LOAD STEP

       DO 500 ILSTEP = 1, NLSTEP


C ---- INITIALIZE COUNTERS AND FLAG

       ICNVRG = 0
        ITER   = 0
        ERROR  = 1.0
        FLAGNR = 0

C ---- ITERATION LOOP

    1  ITER = ITER + 1
```

```
C ---- CHECK FOR ITERMAX

        IF (ITER .GT. ITRMAX) STOP


C ---  INITIALIZE THE GLOBAL STIFFNESS ARRAY AND FORCE VECTOR.

       DO 8 I = 1, NDFMSH
          GF(I) = 0.000
    8  CONTINUE

        IF (FLAGNR .EQ. 0) THEN
       DO 21 I = 1, NDFMSH
         DO 11 J = 1, BNDWTH
            GSTIF(I,J) = 0.000
   11    CONTINUE
   21  CONTINUE
       END IF


C ---  VERIFY IF WE ARE AT THE FIRST ITERATION OF THE FIRST LOAD STEP.

       F_K0 = 0
       IF (ILSTEP .EQ. 1 .AND.  ITER .EQ. 1) THEN
       F_K0 = 1
       END IF


      CALL CALC_T (NPEMAX, NLAMAX, NELMSH, NEMMAX,
     .              NLAYER, NEQMAX, KX, KZ, Z1LAY,  Z2LAY, AK,
     .              FI_D, NODE, GU, T_LAY, KX0, KZ0, F_K0,
     .              LAMBD1, LAMBD2)


       call CR_FI(NLAMAX, NELMSH, NEMMAX,
     .            NLAYER, KZ, Z1LAY, Z2LAY,
     .            AK, FI_D, D_FI_N, D_FI)


C --- BEGIN DO-LOOP ON NUMBER OF ELEMENTS TO CALCULATE THE ELEMENT
C --- MATRICES AND TO ASSEMBLE THE GLOBAL MATRICES.

      DO 50 ELMNUM = 1, NELMSH


C --- DETERMINE THE LOCAL (ELEMENT) DATA.

      DO 30 ENOD = 1, NODELM
         GNOD = NODE(ELMNUM,ENOD)
         EX(ENOD) = X(GNOD)
          EZ(ENOD) = Z(GNOD)
   30 CONTINUE


      call stiff_T      (EDFMAX, NPEMAX, NGPMAX, ndfelm,
     .                   NGPstf, ELMNUM,
     .                   DSF   , GDSF  , EX    , ESTIF , EF,
     .                   GAUSPT, GAUSWT,
     .                   EZ, KX   , KZ,    Z1LAY,     Z2LAY, NEMMAX,
     .                   NLAMAX, NLAYER,AK, FI_D, D_FI, SF2)

C --- PRINT STIFFNESS MATRIX AND FORCE VECTOR FOR ELEMENT ONE.
```

```
      IF (PRTSTF .EQ. 1 .AND. ELMNUM .EQ. 1 .AND. F_K0 .EQ. 1) THEN
         WRITE(OUNIT,1000)
         DO 40 I = 1, NDFELM
            WRITE(OUNIT,1010) I
            WRITE(OUNIT,1020) (ESTIF(I,J), J = 1, NDFELM)
   40    CONTINUE
         WRITE(OUNIT,1030)
         WRITE(OUNIT,1020) (EF(I), I = 1, NDFELM)
      END IF


C --- ASSEMBLE THE GLOBAL MATRICES IN BANDED FORM.

      CALL ASMBLE    (NEQMAX, BNDMAX, EDFMAX, NEMMAX, NPEMAX, NODELM,
     .                NDFNOD, NODE  , ESTIF , EF    , GSTIF , GF    ,
     .                ELMNUM)


C --- END OF LOOP TO COMPUTE GLOBAL STIFFNESS AND FORCE MATRICES.

   50 CONTINUE


C --- PRINT GLOBAL STIFFNESS MATRIX AND FORCE VECTOR. -- VEDI PRECEDENTI
VERSIONI


C ----COMPUTE NEW DISPLACEMENTS VECTOR

       DO 13 I = 1, NBCMAX
          VSGD(I) = IVSGD(I)*(LAMBDA + DLAM)
   13  CONTINUE


C --- IMPOSE PRIMARY BOUNDARY CONDITIONS.

      CALL UBNDRY    (NEQMAX, BNDMAX, NBCMAX, NDFNOD, NDFMSH, BNDWTH,
     .                NSGD  , ISGD  , VSGD  , GSTIF , GF    )



C --- SOLVE THE SYSTEM OF EQUATIONS.

      IRES = 0
      CALL SOLVE (NEQMAX, BNDMAX, NDFMSH, BNDWTH, GSTIF, GF, IRES)


C --- COMPUTE THE INCREMENT.
       DO 113 I = 1, NEQMAX
           DELTA_D(I) = GF(I) - GU(I)
  113  CONTINUE


C-----CHECK FOR CONVERGENGE

       GF2 = 0.0000
       GU2 = 0.0000
       DO 63 I = 1, NEQMAX
          GF2 = GF2 + DELTA_D(I)*DELTA_D(I)
           GU2 = GU2 + GU(I)*GU(I)
   63  CONTINUE

      IF (ITER .GT. 1) THEN
```

```
      UINC = GF2
       UOLD = GU2
         ERROR = DSQRT(UINC/UOLD)
         IF (ERROR .LT. EPSILN) THEN
         ICNVRG = 1
         END IF
      END IF

C-----UPDATE SOLUTION

        DO 73 I = 1, NEQMAX
           GU(I) = GF(I)
   73  CONTINUE

       IF (ICNVRG .EQ. 1) THEN
       DO 83 I = 1, NEQMAX
          GUPLS(I) = GU(I)
   83  CONTINUE
        END IF

C-----PRINT VALUE OF T AT THE CENTER OF THE MESH OR NEAR

       IF  (Flag_P .EQ. 0) THEN
       WRITE(OUNIT,2000)
      WRITE(OUNIT,2030)
       end if
       Flag_P = 1

       IF  (PRTITR .EQ. 1 .and. ICNVRG .EQ. 1 ) THEN
        CEN  = (NELM_X+1)*(NELM_Z/2) + (NELM_X/2+1)
       WRITE(OUNIT,2040) GU(CEN)
 2000  FORMAT(' ',/3X,'SOLUTION AT THE CENTER OF THE MESH: ')
 2030  FORMAT(' ',8X,'T (CENTER)')
 2040  FORMAT(' ',5X,    E14.8,6X)

       END IF


C ----INCREMENT LOAD

       IF (ICNVRG .EQ. 1) THEN


C---- CONVERGENCE OBTAINED, NEXT LOAD STEP

       LAMBDA = LAMBDA + DLAM
         GOTO 500

       ELSE


C---- CONVERGENCE NOT OBTAINED, NEXT ITERATION

         GOTO 1

      END IF

  500 CONTINUE



 1000 FORMAT(' ',//3X,'STIFFNESS MATRIX FOR ELEMENT ONE: ')
```

```
 1010 FORMAT(' ',/5X,'ROW:  ',I2,/)
 1020 FORMAT(' ',5X,6E12.4)
 1030 FORMAT(' ',//3X,'FORCE VECTOR FOR ELEMENT ONE: ',/)
 1040 FORMAT(' ',//3X,'GLOBAL STIFFNESS MATRIX: ',/)
 1050 FORMAT(' ',//3X,'GLOBAL FORCE VECTOR: ',/)


      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     SUBROUTINE:  CALC_T                                             C
C     PURPOSE:     EVALUATE TEMPERATURE AT THE CENTER OF EACH LAYER   C
C                          IN EACH ELEMENT                            C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE CALC_T(NPEMAX, NLAMAX, NELMSH, NEMMAX,
     .                  NLAYER, NEQMAX, KX, KZ, Z1LAY,  Z2LAY, AK,
     .                  FI_D, NODE, GU, T_LAY, KX0, KZ0, F_K0,
     .                  LAMBD1, LAMBD2)


      IMPLICIT LOGICAL (A-Z)

      integer             NPEMAX, NLAMAX, NELMSH,NEMMAX,
     .                      NEQMAX, NLAYER,
     .                    I, L, K1, K2, NODE, F_K0

      double precision KX, KZ, Z1LAY, Z2LAY, AK, FI_D,
     .                 FI_N, FI_T, GU, T_LAY,
     .                 KX0, KZ0, XI1, XI2, X3, LAMBD1, LAMBD2


      DIMENSION  NLAYER(NEMMAX), NODE(NEMMAX,NPEMAX),
     .           KX(nemmax,NLAMAX), KZ(nemmax,NLAMAX),
     .           Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .             AK(nemmax,NLAMAX), FI_D(nemmax),
     .           GU(NEQMAX), T_LAY(nemmax,NLAMAX),
     .             KX0(nemmax,NLAMAX), KZ0(nemmax,NLAMAX),
     .           LAMBD1(nemmax,NLAMAX), LAMBD2(nemmax,NLAMAX)


       IF (F_K0 .EQ. 0) THEN

      DO 10 I = 1, NELMSH

C----- TEMPERATURE AT THE CENTER OF LAYER 1

         FI_T = ( (Z2LAY(I,1)- Z1LAY(I,1)) * 0.5 ) / FI_D(I)
         XI1 =       0.5
         XI2 =       0.5
         T_LAY(I,1) = GU(NODE(I,1))*(1 - FI_T)*XI1 +
     .                  GU(NODE(I,2))*(1 - FI_T)*XI2 +
     .                  GU(NODE(I,3))*(FI_T)*XI2 +
     .                  GU(NODE(I,4))*(FI_T)*XI1

           FI_T = 0.0000
            DO 30 K1 = 2, NLAYER(I)
```

```
      FI_N = (Z2LAY(I,K1) - Z1LAY(I,1)) - (Z2LAY(I,K1) - Z1LAY(I,K1))/2
           X3 = FI_N
           DO 40 K2 = 2, K1
       FI_N = FI_N + ( X3 - ( Z2LAY(I,K2-1)-Z1LAY(I,1) ) )*AK(I,K2-1)
   40    CONTINUE


C----- COMPUTE TEMPERATURE AT THE CENTER OF LAYER 2,3,...,N

        FI_T = FI_N / FI_D(I)
        XI1 =       0.5
        XI2 =       0.5
        T_LAY(I,K1) = GU(NODE(I,1))*(1 - FI_T)*XI1 +
     .                 GU(NODE(I,2))*(1 - FI_T)*XI2 +
     .                 GU(NODE(I,3))*(FI_T)*XI2 +
     .                 GU(NODE(I,4))*(FI_T)*XI1
   30 CONTINUE

   10 CONTINUE



C---- EVALUATE THE NEW KX AND KZ

       DO 101 I = 1, NELMSH
            DO 151 L = 1, NLAYER(I)
      KX(I,L)=KX0(I,L) + LAMBD1(I,L)*T_LAY(I,L)+
     .                   LAMBD2(I,L)*T_LAY(I,L)*T_LAY(I,L)
      KZ(I,L)=KZ0(I,L) + LAMBD1(I,L)*T_LAY(I,L)+
     .                     LAMBD2(I,L)*T_LAY(I,L)*T_LAY(I,L)
  151       CONTINUE
  101  CONTINUE

      ELSE

       DO 200 I = 1, NELMSH
            DO 250 L = 1, NLAYER(I)
            KX(I,L) = KX0(I,L)
            KZ(I,L) = KZ0(I,L)
  250       CONTINUE
  200  CONTINUE

      END IF


      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     SUBROUTINE:  ASMBLE                                             C
C     PURPOSE:     ASSEMBLE THE GLOBAL MATRICES IN BANDED FORM.       C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE ASMBLE (NEQMAX, BNDMAX, EDFMAX, NEMMAX, NPEMAX, NODELM,
     .                   NDFNOD, NODE  , ESTIF , EF    , GSTIF , GF    ,
     .                   ELMNUM)

      IMPLICIT LOGICAL (A-Z)
```

```
      INTEGER            NEQMAX, BNDMAX, EDFMAX, NEMMAX, NPEMAX, NODELM,
     .                   NDFNOD, NODE  , ELMNUM, INOD  , JNOD  , IDOF  ,
     .                   JDOF  , ROW   , COL   , COL1  , EROW  , ECOL

      DOUBLE PRECISION   ESTIF , EF    , GSTIF , GF

      DIMENSION NODE(NEMMAX,NPEMAX) , ESTIF(EDFMAX,EDFMAX), EF(EDFMAX),
     .          GSTIF(NEQMAX,BNDMAX), GF(NEQMAX)


      DO 40 INOD = 1, NODELM
         ROW = NDFNOD * (NODE(ELMNUM,INOD) - 1)

         DO 30 IDOF = 1, NDFNOD
            ROW  = ROW + 1
            EROW = (INOD - 1) * NDFNOD + IDOF
            GF(ROW) = GF(ROW) + EF(EROW)

            DO 20 JNOD = 1, NODELM
               COL1 = NDFNOD * (NODE(ELMNUM,JNOD) - 1)

               DO 10 JDOF = 1, NDFNOD
                  COL  = COL1 - ROW + JDOF + 1
                  ECOL = (JNOD - 1) * NDFNOD + JDOF
                  IF (COL .GT. 0) THEN
                     GSTIF(ROW,COL) = GSTIF(ROW,COL) + ESTIF(EROW,ECOL)
                  END IF
   10          CONTINUE
   20       CONTINUE
   30    CONTINUE
   40 CONTINUE
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                       C
C     SUBROUTINE:  CR_FI                                                C
C     PURPOSE:     EVALUATE FUNCTIONS FI AND ITS DERIVATIVES            C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE CR_FI(NLAMAX, NELMSH, NEMMAX,
     .                 NLAYER, KZ, Z1LAY, Z2LAY,
     .                 AK, FI_D, D_FI_N, D_FI)


      IMPLICIT LOGICAL (A-Z)

      integer            NLAMAX, NELMSH,NEMMAX,
     .                   NLAYER, I , K, L , N

      double precision   KZ, Z1LAY, Z2LAY, AK, FI_D,
     .                   H, R, D_FI_N, D_FI, SUM_AK


      DIMENSION  NLAYER(NEMMAX), KZ(nemmax,NLAMAX),
     .           Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .           AK(nemmax,NLAMAX), FI_D(nemmax), D_FI_N(nemmax,NLAMAX),
     .           D_FI(nemmax,NLAMAX)
```

```
      DO 100 I = 1, NELMSH

      if (NLAYER(I) .gt. 1) then
        SUM_AK = 0
        AK(I,1) = ( KZ(I,1)/KZ(I,2) - 1 )
        SUM_AK = AK(I,1)

        DO 30 K = 2,  (NLAYER(I)-1)
        AK(I,K) = ( KZ(I,K)/KZ(I,K+1) - 1 )*( 1 + SUM_AK)
        SUM_AK  = SUM_AK + AK(I,K)

 30     CONTINUE

       END IF

       H = ( Z2LAY(I,NLAYER(I)) - Z1LAY(I,1) )

      FI_D(I) = H

       DO 40 R = 1, (NLAYER(I)-1)
       FI_D(I) = FI_D(I) + ( H - (Z2LAY(I,R)- Z1LAY(I,1)) )*AK(I,R)
 40   CONTINUE

       D_FI_N(I,1) = 1

       DO 50 L = 2, NLAYER(I)
       D_FI_N(I,L) = D_FI_N(I,L-1) +   AK(I,L-1)
 50    CONTINUE

       DO 60  N = 1, NLAYER(I)
       D_FI(I,N) = D_FI_N(I,N) /  FI_D(I)
 60    CONTINUE


100    CONTINUE



        RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  FBNDRY                                              C
C     PURPOSE:     IMPOSE THE PRESCRIBED FORCE B.C.'S ON THE           C
C                  BANDED SYMMETRIC STIFFNESS MATRIX AND MODIFY        C
C                  THE FORCE VECTOR.                                   C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE FBNDRY (NEQMAX, NBCMAX, NDFNOD, NSGF  , ISGF  , VSGF  ,
     .                   GF    )

      IMPLICIT LOGICAL (A-Z)

      INTEGER            NEQMAX, NBCMAX, NDFNOD, NSGF  , ISGF  , IBC   ,
     .                   IDOF

      DOUBLE PRECISION   VSGF  , GF

      DIMENSION          ISGF(NBCMAX,2), VSGF(NBCMAX), GF(NEQMAX)
```

```
      DO 10 IBC = 1, NSGF
         IDOF = NDFNOD * (ISGF(IBC,1) - 1) + ISGF(IBC,2)
         GF(IDOF) = GF(IDOF) + VSGF(IBC)
   10 CONTINUE
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  GQUAD                                               C
C     PURPOSE:     ASSIGN VALUES TO THE ARRAYS "GAUSPT" AND "GAUSWT"   C
C                  USED IN GAUSSIAN QUADRATURE.                        C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE GQUAD  (NGPMAX, GAUSPT, GAUSWT )

      IMPLICIT LOGICAL (A-Z)

       INTEGER            NGPMAX, I     , J
        DOUBLE PRECISION   GAUSPT, GAUSWT

       DIMENSION          GAUSPT(NGPMAX,NGPMAX), GAUSWT(NGPMAX,NGPMAX)


C ---  INITIALIZE ARRAYS.

       DO 20 I = 1,NGPMAX
        DO 10 J = 1,NGPMAX
         GAUSPT(I,J) = 0.000
         GAUSWT(I,J) = 0.000
   10 CONTINUE
   20 CONTINUE

C --- GAUSSIAN QUADRATURE OF ORDER 1.

      GAUSPT(1,1) = 0.000
      GAUSWT(1,1) = 2.000

C --- GAUSSIAN QUADRATURE OF ORDER 2.

      GAUSPT(1,2) = -1.000/SQRT(3.000)
      GAUSPT(2,2) = -GAUSPT(1,2)
      GAUSWT(1,2) =  1.000
      GAUSWT(2,2) =  GAUSWT(1,2)

C --- GAUSSIAN QUADRATURE OF ORDER 3.

      GAUSPT(1,3) = -SQRT(3.000/5.000)
      GAUSPT(2,3) =  0.000
      GAUSPT(3,3) = -GAUSPT(1,3)
      GAUSWT(1,3) =  5.000/9.000
      GAUSWT(2,3) =  8.000/9.000
      GAUSWT(3,3) =  GAUSWT(1,3)

C --- GAUSSIAN QUADRATURE OF ORDER 4.

      GAUSPT(1,4) = -0.8611363116
      GAUSPT(2,4) = -0.3399810436
      GAUSPT(3,4) = -GAUSPT(2,4)
```

```
      GAUSPT(4,4) = -GAUSPT(1,4)
      GAUSWT(1,4) =  0.3478548451
      GAUSWT(2,4) =  0.6521451549
      GAUSWT(3,4) =  GAUSWT(2,4)
      GAUSWT(4,4) =  GAUSWT(1,4)

      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  INPELM                                              C
C     PURPOSE:     READ ELEMENT DATA.                                  C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INPELM (OUNIT , nodelm, ndfelm, ngpstf, ngpgrd,
     .                   ndfnod)

      IMPLICIT LOGICAL (A-Z)

      integer           OUNIT , nodelm, ndfelm, ngpstf, ngpgrd, ndfnod

      NDFNOD = 1
      NDFELM = 4
       NGPSTF = 2
      NGPGRD = 2
       NODELM = 4

      WRITE(OUNIT,1000)
      WRITE(OUNIT,1010) NODELM, ndfnod, ndfelm, ngpstf, ngpgrd

 1000 FORMAT(' ',//3X,'ELEMENT DATA:')
 1010 FORMAT(' ',
     . //10X,'NUMBER OF NODES PER ELEMENT, ............. NODELM =',I4,
     .   /10X,'NUMBER OF D.O.F. IN EACH NODE, ........... NDFNOD =',I4,
     .   /10X,'NUMBER OF D.O.F. IN EACH ELEMENT, ......... NDFELM =',I4,
     . //10X,'INTEGRATION RULE , ....................... NGPSTF =',I4,
     .   /10X,'INTEGRATION POINTS , ...................... NGPGRD =',I4)
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  INP_NL                                              C
C     PURPOSE:     READ OPTION PARAMETERS FOR NON LINEAR ANALYSIS      C
C                  FROM THE INPUT FILE.                                C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INP_NL (IUNIT , OUNIT , NLSTEP, DLAM, ITRMAX, EPSILN )

      IMPLICIT LOGICAL (A-Z)

      INTEGER           IUNIT , OUNIT , NLSTEP, ITRMAX

      DOUBLE PRECISION  DLAM, EPSILN, NLS_TM
```

```
      READ(IUNIT,*)  NLSTEP, ITRMAX, EPSILN

       NLS_TM = NLSTEP

      DLAM = 1/NLS_TM

      WRITE(OUNIT,1000)
      WRITE(OUNIT,1010) NLSTEP, DLAM, ITRMAX, EPSILN


 1000 FORMAT(' ',//3X,'NON LINEAR ANALYSIS DATA:')
 1010 FORMAT(' ',
     .//10X,'NUMBER OF LOAD STEP, .................... NLSTEP =',I6,
     . /10X,'LOAD INCREMENT SCALE, ................... DLAM   =',E10.4,
     . /10X,'MAX. NUMBER OF ITERATIONS, .............. ITRMAX =',I6,
     . /10X,'TOLLERANCE, ............................. EPSILN =',E10.4)


      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     SUBROUTINE:  INLAYR                                             C
C     PURPOSE:     INPUT ELEMENT LAYERS                               C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INLAYR(IUNIT , OUNIT , NELMSH, NLAYER, NLAMAX,NEMMAX,
       .                   KX0, KZ0, Z1LAY, Z2LAY, LAMBD1, LAMBD2, PRTLAY,
     .                   NELM_X, NELM_Z, INPLAY)

      IMPLICIT LOGICAL (A-Z)

      INTEGER           I, J, E, K, IUNIT, OUNIT, NELMSH, NLAYER, NLAMAX,
       .                  NEMMAX, PRTLAY, INPLAY, NELM_X, NELM_Z, ASSLAY,
     .                  NLATMP

      DOUBLE PRECISION   KX0, KZ0, Z1LAY, Z2LAY, LAMBD1, LAMBD2,
       .                   KX0TMP, KZ0TMP, LB1TMP, LB2TMP,
     .                   Z1LTMP, Z2LTMP

      DIMENSION          NLAYER(NEMMAX),
       .                   KX0(nemmax,NLAMAX),     KZ0(nemmax,NLAMAX),
     .                   Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
       .                   LAMBD1(nemmax,NLAMAX), LAMBD2(nemmax,NLAMAX)


       IF (INPLAY .EQ. 1) THEN
      E = 0
       ASSLAY = 0
      DO 80 I=1, NELM_Z
          READ(IUNIT,*) NLATMP
          DO 70 J = 1, NLATMP
          E = 0
          READ(IUNIT,*) KX0TMP, KZ0TMP, LB1TMP, LB2TMP, Z1LTMP, Z2LTMP
            E = 1 + (I-1)*NELM_X
            DO 60 K = 1, NELM_X
            NLAYER(E) = NLATMP
               KX0(E,J)     = KX0TMP
            KZ0(E,J)     = KZ0TMP
```

```
          LAMBD1(E,J) = LB1TMP
          LAMBD2(E,J) = LB2TMP
         Z1LAY(E,J)  = Z1LTMP
         Z2LAY(E,J)  = Z2LTMP
            E = E + 1
   60       CONTINUE
   70     CONTINUE
   80 CONTINUE

       ELSE

      DO 180 I=1, NELMSH
          READ(IUNIT,*) NLAYER(I)
          DO 170 J=1, NLAYER(I)
          READ(IUNIT,*) KX0(I,J), KZ0(I,J), LAMBD1(I,J), LAMBD2(I,J),
     .     Z1LAY(I,J), Z2LAY(I,J)
  170     CONTINUE
  180 CONTINUE

      END IF


       IF (PRTLAY .EQ. 1) THEN
      WRITE(OUNIT,1000)

       DO 100        I=1, NELMSH

      WRITE(OUNIT,1010)
       WRITE(OUNIT,1020) I, NLAYER(I)
      WRITE(OUNIT,1030)
                DO 90 J=1, NLAYER(I)
             WRITE(OUNIT,1040) J,KX0(I,J),KZ0(I,J),Z1LAY(I,J),Z2LAY(I,J)

90        CONTINUE
100     CONTINUE
       END IF


C --- FORMAT STATEMENTS.

 1000 FORMAT(' ',///3X,'NUMBERS AND CHARACTERISTICS OF LAYERS: ')
 1010 FORMAT(' ',//6X,'ELEMENT',3X,'N LAYER',/)
 1020 FORMAT(' ',9X,I3,7X,I3)
 1030 FORMAT(' ',/6X,'N LAYER',5X,'KX0',12X,'KZ0',12X,'Z1LAY',9X,
     .                'Z2LAY',5X,/)
 1040 FORMAT(' ',9X,I3,4X,E10.4,4X,E10.4,4X,E10.4,4X,E10.4)

      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  INPMSH                                              C
C     PURPOSE:     INPUT MESH PARAMETERS AND FORM THE MESH.             C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INPMSH (IUNIT , OUNIT , NEMMAX, NPEMAX, NNMMAX, nodelm,
     .                   ndfnod,DOM_X, DOM_Z, NELM_X, NELM_Z,
     .                   NELMSH, NDFMSH, nodmsh, node, BNDWTH, X, Z)
```

```
      IMPLICIT LOGICAL (A-Z)

      integer           IUNIT , OUNIT , NEMMAX, NPEMAX, NNMMAX, nodelm,
     .                  ndfnod, NELM_X, NELM_Z, nodmsh, BNDWTH,
     .                  NDFMSH,   node, IMESH, EORDER, NXINCR, NZINCR,
     .                  I     , J     ,      N, NELMSH, NW, NPE

       DOUBLE PRECISION   X , Z, DOM_X, DOM_Z, DX, DZ

      DIMENSION     NODE(NEMMAX,NPEMAX), X(NNMMAX), Z(NNMMAX),
     .               DX(NNMMAX)     , DZ(NNMMAX)

      READ(IUNIT,*) IMESH, DOM_X, DOM_Z, NELM_X, NELM_Z


      NODMSH = (NELM_X + 1)*(NELM_Z + 1)
      NELMSH =  NELM_X * NELM_Z

      IF (IMESH .EQ. 0) THEN
         DO 10 I = 1, NELMSH
            READ(IUNIT,*) (NODE(I,J), J=1,NODELM)
10       CONTINUE
         DO 20 I = 1, NODMSH
         READ(IUNIT,*) X(I), Z(I)
20       CONTINUE

      ELSE

         EORDER = 1
         NXINCR = NELM_X * EORDER
         NZINCR = NELM_Z * EORDER
         READ(IUNIT,*) (DX(I), I = 1, NXINCR)
         READ(IUNIT,*) (DZ(I), I = 1, NZINCR)
         NPE = NODELM
         CALL MESH   (NPEMAX, NEMMAX, NNMMAX, EORDER, NODE   , NPE    ,
     .                NODMSH, NELMSH, NELM_X, NELM_Z, X      , Z      ,
     .                DX     , DZ)
      END IF

      NDFMSH = NDFNOD * NODMSH

      BNDWTH = 0
       DO 1       N = 1,NELMSH
         DO 2     I = 1,NODELM
           DO 3   J = 1,NODELM
             NW = ( ABS( NODE(N,I) - NODE(N,J) ) + 1 )
            IF (BNDWTH .LT. NW) THEN
            BNDWTH = NW
             END IF
 3         CONTINUE
 2       CONTINUE
 1     CONTINUE


      WRITE(OUNIT,1000)
      WRITE(OUNIT,1010) NELM_X, NELM_Z, NODMSH, NDFMSH, BNDWTH
      WRITE(OUNIT,1020)
      WRITE(OUNIT,1030)
      DO 90 N = 1,NELMSH
         WRITE(OUNIT,1040) N, (NODE(N,I), I=1,NODELM)
90    CONTINUE
```

```
 1000 FORMAT(' ',//3X,'MESH DATA:')
 1010 FORMAT(' ',
     . //10X,'N. OF ELEMENTS IN THE MESH IN X DIRECTION, .NELMSH =',I4,
     .  /10X,'N. OF ELEMENTS IN THE MESH IN Z DIRECTION, .NELMSH =',I4,
     .  /10X,'NUMBER OF NODES IN THE MESH, ............. NODMSH =',I4,
     .  /10X,'NUMBER OF D.O.F. IN THE MESH, ............ NDFMSH =',I4,
     .  /10X,'HALF BAND WIDTH OF GLOBAL STIFFNESS MATRIX, BNDWTH =',I4)
 1020 FORMAT(' ',//10X,'BOOLEAN (CONNECTIVITY) MATRIX: ')
 1030 FORMAT(' ',/10X,'ELEMENT NO.',6X,'NODE(1,2,3,4)',/)
 1040 FORMAT(' ',11X,I3,10X,9(I3,1X))
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  MESH                                                C
C     PURPOSE:     GENERATE THE CONNECTIVITY ARRAY NODE(I,J), THE      C
C                  COORDINATES X(I), Z(I), AND MESH INFORMATION        C
C                  (NODMSH, NELMSH, NODELM) FOR RECTANGULAR DOMAINS.   C
C                  THE DOMAIN IS DIVIDED INTO RECTANGULAR ELEMENTS     C
C                  WITH FOUR, EIGHT, OR NINE NODES (NUMELX BY NUMELY   C
C                  NONUNIFORM MESH IN GENERAL).                        C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC


       SUBROUTINE MESH    (NPEMAX, NEMMAX, NNMMAX, IEL    , NOD    , NPE,
     .                     NNM    , NEM    , NX     , NZ     , X      , Z  ,
     .                     DX     , DZ     )


      IMPLICIT DOUBLE PRECISION (A-H,O-Z)


      INTEGER  NPEMAX, NEMMAX, NNMMAX, IEL    , NOD    , NPE    ,
     .         NNM    , NEM    , NX     , NZ


      DOUBLE PRECISION X      , Z      , DX     , DZ


       DIMENSION X(NNMMAX) , Z(NNMMAX) , NOD(NEMMAX,NPEMAX),
     .          DX(NNMMAX), DZ(NNMMAX)

      NEX1=NX+1
      NEZ1=NZ+1
      NXX=IEL*NX
      NZZ=IEL*NZ
      NXX1=NXX+1
      NZZ1=NZZ+1
      NEM=NX*NZ
      NNM=NXX1*NZZ1-(IEL-1)*NX*NZ
      IF(NPE.EQ.9)NNM=NXX1*NZZ1

C --- GENERATE THE CONNECTIVITY ARRAY NOD(I,J).

      K0=0
      IF(NPE.EQ.9)K0=1
      NOD(1,1) = 1
      NOD(1,2) = IEL+1
      NOD(1,3) = NXX1+(IEL-1)*NEX1+IEL+1
```

```
      IF (NPE .EQ. 9)NOD(1,3)=4*NX+5
      NOD(1,4) = NOD(1,3) - IEL
      IF(NPE .EQ. 4)GO TO 200
      NOD(1,5) = 2
      NOD(1,6) = NXX1 + (NPE-6)
      NOD(1,7) = NOD(1,3) - 1
      NOD(1,8) = NXX1+1
      IF (NPE .EQ. 9)NOD(1,9)=NXX1+2
  200 IF(NZ .EQ. 1)GOTO 230
      M = 1
      DO 220 N = 2,NZ
      L = (N-1)*NX + 1
       DO 210 I = 1,NPE
  210  NOD(L,I) = NOD(M,I)+NXX1+(IEL-1)*NEX1+K0*NX
  220  M=L

  230 IF(NX .EQ .1)GO TO 270



         DO 260 NI = 2,NX
      DO 240 I = 1,NPE
      K1 = IEL
      IF(I .EQ. 6 .OR. I .EQ. 8)K1=1+K0
  240 NOD(NI,I) = NOD(NI-1,I)+K1
      M = NI
      DO 260 NJ = 2,NZ
      L = (NJ-1)*NX+NI
      DO 250 J = 1,NPE
  250 NOD(L,J) = NOD(M,J)+NXX1+(IEL-1)*NEX1+K0*NX
  260 M = L

C --- GENERATE THE COORDINATES X(I) AND Z(I)

  270 DX(NXX1)=0.0
      DZ(NZZ1)=0.0
      ZC=0.0
      IF (NPE .EQ. 9) GOTO 310
      DO 300 NI = 1, NEZ1
      I = (NXX1+(IEL-1)*NEX1)*(NI-1)+1
      J = (NI-1)*IEL+1
      X(I) = 0.0
      Z(I) = ZC
      DO 280 NJ = 1,NXX
      I=I+1
      X(I) = X(I-1)+DX(NJ)
  280 Z(I) = ZC
      IF(NI.GT.NZ .OR. IEL.EQ.1)GO TO 300
      J = J+1
      ZC = ZC+DZ(J-1)
      I = I+1
      X(I) = 0.0
      Z(I) = ZC
      DO 290 II = 1, NX
      K = 2*II-1
      I = I+1
      X(I) = X(I-1)+DX(K)+DX(K+1)
  290 Z(I) = ZC
  300 ZC = ZC+DZ(J)
      RETURN

  310 DO 330 NI=1,NZZ1
      I=NXX1*(NI-1)
```

```
      XC=0.0
      DO 320 NJ=1,NXX1
      I = I+1
      X(I) = XC
      Z(I) = ZC
  320 XC = XC + DX(NJ)
  330 ZC = ZC + DZ(NI)
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  INPNBC                                              C
C     PURPOSE:     INPUT NODAL BOUNDARY CONDITIONS.                    C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INPNBC (IUNIT , OUNIT , NBCMAX, NNMMAX, NODMSH, ndfnod,
     .                   NSGD  , NSGF  , ISGD  , ISGF  , X,  Z , vsgd  ,
     .                   VSGF, FLAG_D)

      IMPLICIT LOGICAL (A-Z)

      INTEGER            IUNIT , OUNIT , NBCMAX, NNMMAX, NODMSH, ndfnod,
     .                   NSGD  , NSGF  , ISGD  , ISGF  , FLAG_D,
     .                   I     , J     , K     , IBC   , NOD   , DOF

      DOUBLE PRECISION   X  ,   Z, VSGD  , VSGF

      DIMENSION          ISGD(NBCMAX,2), ISGF(NBCMAX,2), IBC(4),
     .                   VSGD(NBCMAX)  , VSGF(NBCMAX)  , X(NNMMAX),
     .                   Z(NNMMAX)

      READ(IUNIT,*) NSGD, NSGF

       FLAG_D = 1
       IF (NSGD .GT. 0) THEN
         READ(IUNIT,*)  ((ISGD(I,J),J=1,2),I=1,NSGD)
         READ(IUNIT,*)  (VSGD(I),I=1,NSGD)
      END IF
      IF (NSGF .GT. 0) THEN
          FLAG_D = 0
         READ(IUNIT,*)  ((ISGF(I,J),J=1,2),I=1,NSGF)
         READ(IUNIT,*)  (VSGF(I),I=1,NSGF)
      END IF

      WRITE(OUNIT,1000)
      WRITE(OUNIT,1010) NSGD
      WRITE(OUNIT,1020)
      DO 30 I = 1,NODMSH
         DO 10 K = 1,NDFNOD
            IBC(K) = 0
   10    CONTINUE
         DO 20 J = 1,NSGD
            NOD = ISGD(J,1)
            DOF = ISGD(J,2)
            IF (NOD .EQ. I) IBC(DOF) = DOF
   20    CONTINUE
         WRITE(OUNIT,1030) I, X(I), Z(I), (IBC(K), K = 1, NDFNOD)
   30 CONTINUE
```

```
      WRITE(OUNIT,1035)
      WRITE(OUNIT,1070) (VSGD(I), I = 1, NSGD)
       WRITE(OUNIT,2050)

      IF (NSGF .NE. 0) THEN
         WRITE(OUNIT,1040) NSGF
         WRITE(OUNIT,1050)
         DO 60 I = 1,NODMSH
         DO 40 K = 1,NDFNOD
            IBC(K) = 0
   40    CONTINUE
         DO 50 J = 1,NSGF
            NOD = ISGF(J,1)
            DOF = ISGF(J,2)
            IF (NOD .EQ. I) IBC(DOF) = DOF
   50    CONTINUE
         WRITE(OUNIT,1030) I, X(I), Z(I), (IBC(K), K = 1, NDFNOD)
   60    CONTINUE
         WRITE(OUNIT,1060)
         WRITE(OUNIT,1070) (VSGF(I), I = 1, NSGF)
          WRITE(OUNIT,2060)
      END IF

C --- FORMAT STATEMENTS.

 1000 FORMAT(' ',//3X,'NODAL BOUNDARY CONDITIONS: ',/)
 1010 FORMAT(' ',9X,'NUMBER OF SPECIFIED TEMPERATURE D.O.F =',I4)
 1020 FORMAT(' ',/10X,'NODE',4X,'X-COORD.',6X,'Z-COORD.'5X,
     .                 'SPECIFIED TEMPERATURE D.O.F.',/)
 1030 FORMAT(' ',9X,I3,4X,E10.4,4X,E10.4,3X,12(I2,1X))
 1035 FORMAT(' ',/10X,'VALUES OF THE SPECIFIED TEMPERATURE D.O.F.:')
 1040 FORMAT(' ',10X,'NUMBER OF SPECIFIED GENERALIZED FORCES =',I4)
 1050 FORMAT(' ',/10X,'NODE',4X,'X-COORD.',4X,'Z-COORD.',5X,
     .                 'SPECIFIED GENERALIZED FORCE',/)
 1060 FORMAT(' ',/10X,'VALUES OF THE SPECIFIED GENERALIZED FORCES:')
 1070 FORMAT(' ',/10X,6(E10.4,2X))
 2050  FORMAT(' ',/)
 2060  FORMAT(' ',/)
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                     C
C     SUBROUTINE:  INPOPT                                             C
C     PURPOSE:     READ OPTION PARAMETERS FROM THE INPUT FILE.        C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE INPOPT (IUNIT, OUNIT, PRTSTF, PRTGRD, PRTLAY, INPLAY,
     .                    PRTITR)

      IMPLICIT LOGICAL (A-Z)

      INTEGER    IUNIT , OUNIT , PRTSTF, PRTGRD, PRTLAY, INPLAY, PRTITR

      CHARACTER*72       TITLE

      READ(IUNIT,1000)   TITLE
       READ(IUNIT,*)      PRTSTF, PRTLAY, INPLAY, PRTITR

       PRTGRD = 0
```

```
      WRITE(OUNIT,1010)
      WRITE(OUNIT,1020)  TITLE

 1000 FORMAT(A72)
 1010 FORMAT(' ',//3X,
     .    'RESULTS OF "THERMNL VERS. 1.0" - WRITTEN BY ANTONIO PANTANO')
 1020 FORMAT(' ',//3X,A72)
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                       C
C     SUBROUTINE:  PRTGRD                                               C
C     PURPOSE:     WRITE THE SOLUTION GRADIENTS TO THE OUTPUT FILE.     C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine prtdu  (ounit , npemax, neqmax, nnmmax, edfmax, nemmax,
     .                   ngpmax, nodelm, ndfnod, ndfelm, nelmsh, ngpgrd,
     .                   node  ,
     .                   gauspt, gu    , eu    , ex    , sf    ,
     .                   gdsf  , a0    , a1    , a2    , x     )


      IMPLICIT LOGICAL (A-Z)

      integer            ounit , npemax, neqmax, nnmmax, edfmax, nemmax,
     .                   ngpmax, nodelm, ndfnod, ndfelm, nelmsh, ngpgrd,
     .                   node  ,
     .                   elmnum, idof  , igp   , enod  , gnod  , edof  ,
     .                   gdof

      double precision   gauspt, gu    , eu    , ex    , sf    ,
     .                   gdsf  , a0    , a1    , a2    , x     ,
     .                   xc    , dudx  , adudx , a     , xi    , elenth

      dimension          gu(neqmax)     , eu(edfmax)     , ex(npemax)     ,
     .                   sf(npemax)     , gdsf(npemax)   ,
     .                   a0(nemmax)     , a1(nemmax)     , a2(nemmax)     ,
     .                   x(nnmmax)      ,
     .                   gauspt(ngpmax,ngpmax), node(nemmax,npemax)

       WRITE(OUNIT,1000)
       WRITE(OUNIT,1010)

C --- BEGIN LOOP OVER NUMBER OF ELEMENTS IN THE MESH.

      DO 50 ELMNUM = 1, NELMSH

C --- DETERMINE THE LOCAL (ELEMENT) SOLUTION VECTOR AND NODAL
C --- COORDINATES.

      gdof = (node(elmnum,1) - 1) * ndfnod
      do 10 edof = 1, ndfelm
         eu(edof) = gu(gdof+edof)
   10 continue
      do 20 enod = 1, nodelm
         gnod = node(elmnum,enod)
         ex(enod) = x(gnod)
   20 continue
```

```
      elenth = ex(nodelm) - ex(1)

C --- BEGIN LOOP ON NUMBER OF GAUSS POINTS (NGPGRD).

      DO 40 IGP = 1, NGPGRD
      XI = GAUSPT(IGP,NGPGRD)

C --- COMPUTE THE SOLUTION GRADIENTS AT THE GAUSS POINTS.

CA        CALL SHPL1D (NPEMAX, NODELM, XI    , ELENTH, SF    , DSF   ,
CA       .                  GDSF  , JACOBN)

      DUDX = 0.000
      XC   = 0.000
      DO 30 IDOF = 1, NODELM
         DUDX = DUDX + EU(IDOF) * GDSF(IDOF)
         XC   = XC   + EX(IDOF) * SF(IDOF)
   30 CONTINUE

      A     = A0(elmnum) + A1(elmnum)*XC + A2(elmnum)*XC*XC
      ADUDX = A*DUDX

      WRITE(OUNIT,1030) ELMNUM, XC, DUDX, ADUDX

   40 CONTINUE
   50 CONTINUE

C --- FORMAT STATEMENTS.

 1000 FORMAT(' ',/3X,'SOLUTION GRADIENTS:  ',/)
 1010 FORMAT(' ',/10X,'ELEMENT NO.  X-COORD.     DU/DX       A*DU/DX',/)
 1030 FORMAT(' ', 11X,I3,6X,4(E10.4,2X))
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  PRTGU                                               C
C     PURPOSE:     WRITE THE SOLUTION VECTOR TO THE OUTPUT FILE.       C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      subroutine prtgu  (OUNIT , NEQMAX, NNMMAX, NDFNOD, NODMSH, GU    ,
     .                   X, Z )

      IMPLICIT LOGICAL (A-Z)

      INTEGER            OUNIT , NEQMAX, NNMMAX, NDFNOD, NODMSH,
     .                   U1    , U2    , I     , J

      DOUBLE PRECISION   GU    , X,    Z

      DIMENSION          GU(NEQMAX)    , X(NNMMAX),   Z(NNMMAX)

      WRITE(OUNIT,1000)

C --- OUTPUT OF SOLUTION VECTOR.

      WRITE(OUNIT,1010)
      U2 = 0
```

```
      DO 20 I = 1, NODMSH
         U1 = U2 + 1
         U2 = U2 + ndfNOD
         WRITE(OUNIT,1020) I, X(I), Z(I), (GU(J), J = U1, U2)
   20 CONTINUE

C --- FORMAT STATEMENTS.

 1000 FORMAT(' ',///3X,'FINAL SOLUTION OF THE NON LINEAR ANALYSIS:  ',/)
 1010 FORMAT(' ',10X,'NODE',7X,'X-COORD.',5X,'Z-COORD. '6X, 'T',/)
 1020 FORMAT(' ', 10X,I3,6X,E10.4,2X,E10.4,2X,E11.6)
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                       C
C     SUBROUTINE:  SHPL1D_T                                             C
C     PURPOSE:     EVALUATE THE 1-D LAGRANGIAN INTERPOLATION FUNCTIONS C
C                  AND THEIR GLOBAL DERIVATIVES AT THE GAUSS POINTS.   C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE SHPL1D_T (NPEMAX, XI , ELENTH_X, DSF,
     .                       GDSF  , JACOBN, SF2, IGP)


      IMPLICIT LOGICAL (A-Z)

      INTEGER            NPEMAX, I,  IGP

      DOUBLE PRECISION   XI   , ELENTH_X, DSF   , GDSF   , JACOBN,
     .                    SF2

      DIMENSION          DSF(NPEMAX), GDSF(NPEMAX),
     .                    SF2(2,2)

C --- LINEAR LAGRANGE INTERPOLATION FUNCTIONS FOR 2-NODED ELEMENTS.

         SF2(1,IGP)  =  0.5000 * (1.000 - XI)
         SF2(2,IGP)  =  0.5000 * (1.000 + XI)
         DSF(1) = -0.5000
         DSF(2) =  0.5000

C --- COMPUTE THE GLOBAL DERIVATIVES OF SF(I).

      JACOBN = ELENTH_X * 0.5000

      DO 10 I = 1, 2
         GDSF(I) = DSF(I) / JACOBN
   10 CONTINUE
      RETURN
      END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                       C
C     SOLVE:                                                            C
```

```
C     THIS PROGRAM SOLVES A BANDED SYMMETRIC SYSTEM OF EQUATIONS.     C
C     THE BANDED MATRIX IS INPUT THROUGH BAND(NEQNS,NBW), AND         C
C     RHS IS THE RIGHT HAND SIDE (FORCE VECTOR) OF THE EQUATION.      C
C     NEQNS IS THE NO. OF EQUATIONS AND NBW IS THE HALF BAND WIDTH.   C
C     IN RESOLVING, IRES .GT. 0, LHS ELIMINATION IS SKIPPED.          C
C                                                                     C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE SOLVE  (NRM ,  NCM  , NEQNS , NBW   , BAND, RHS, IRES)

      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

      INTEGER            NRM   , NCM   , NEQNS , NBW   , IRES

      DOUBLE PRECISION   BAND  , RHS

      DIMENSION          BAND(NRM,NCM), RHS(NRM)


      MEQNS=NEQNS-1
      IF (IRES.GT.0) GO TO 40
      DO 30 NPIV=1,MEQNS
      NPIVOT=NPIV+1
      LSTSUB=NPIV+NBW-1
      IF (LSTSUB.GT.NEQNS) LSTSUB=NEQNS
      DO 20 NROW=NPIVOT,LSTSUB


C --- INVERT ROWS AND COLUMNS FOR ROW FACTOR.

      NCOL=NROW-NPIV+1

      FACTOR=BAND(NPIV,NCOL)/BAND(NPIV,1)
      DO 10 NCOL=NROW,LSTSUB
      ICOL=NCOL-NROW+1
      JCOL=NCOL-NPIV+1
   10 BAND(NROW,ICOL)=BAND(NROW,ICOL)-FACTOR*BAND(NPIV,JCOL)
   20 RHS(NROW)=RHS(NROW)-FACTOR*RHS(NPIV)
   30 CONTINUE
      GO TO 70
   40 DO 60 NPIV=1,MEQNS
      NPIVOT=NPIV+1
      LSTSUB=NPIV+NBW-1
      IF (LSTSUB.GT.NEQNS) LSTSUB=NEQNS
      DO 50 NROW=NPIVOT,LSTSUB
      NCOL=NROW-NPIV+1
      FACTOR=BAND(NPIV,NCOL)/BAND(NPIV,1)
   50 RHS(NROW)=RHS(NROW)-FACTOR*RHS(NPIV)
   60 CONTINUE

C --- BACK SUBSTITUTION.

   70 DO 90 IJK=2,NEQNS
      NPIV=NEQNS-IJK+2
      RHS(NPIV)=RHS(NPIV)/BAND(NPIV,1)
      LSTSUB=NPIV-NBW+1
      IF (LSTSUB.LT.1) LSTSUB=1
      NPIVOT=NPIV-1
      DO 80 JKI=LSTSUB,NPIVOT
      NROW=NPIVOT-JKI+LSTSUB
      NCOL=NPIV-NROW+1
      FACTOR=BAND(NROW,NCOL)
   80 RHS(NROW)=RHS(NROW)-FACTOR*RHS(NPIV)
```

```
   90 CONTINUE
      RHS(1)=RHS(1)/BAND(1,1)
      RETURN
      END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                       C
C     SUBROUTINE:  STIFF_T                                              C
C     PURPOSE:     COMPUTE THE ELEMENTAL STIFFNESS AND FORCE ARRAYS     C
C                                                                       C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE STIFF_T(EDFMAX, NPEMAX, NGPMAX, ndfelm,
       .                   NGPstf, ELMNUM,
       .                   DSF, GDSF, EX, ESTIF, EF, GAUSPT, GAUSWT,
     .                    EZ, KX   , KZ,   Z1LAY,  Z2LAY, NEMMAX,
     .                    NLAMAX, NLAYER,AK, FI_D, D_FI,  SF2)


      IMPLICIT LOGICAL (A-Z)

      integer             EDFMAX, NPEMAX, NGPMAX,  ndfelm,
     .                    NGPstf, I     , J     , igp,
     .                    NLAMAX, NLAYER, NEMMAX, ELMNUM, K

      double precision    DSF   , GDSF, EX, ESTIF, EF,
     .                    GAUSPT, GAUSWT,
     .                    xi    , const , jacobn,
     .                    ELENTH_X, ELENTH_Z,
     .                    EZ, KX   , KZ,    Z1LAY,  Z2LAY,
       .                    AK,   FI_D,  D_FI, SF2, K_LAY,
     .                    FI, D_FI_F, FI_N, H_LAYER, FXI, DXI


      dimension  dsf(npemax), gdsf(npemax),
     .           EX(NPEMAX), ef(edfmax),
     .           GAUSPT(NGPMAX,NGPMAX), GAUSWT(NGPMAX,NGPMAX) ,
     .           ESTIF(EDFMAX,EDFMAX), EZ(NPEMAX), NLAYER(NEMMAX),
     .           KX(nemmax,NLAMAX), KZ(nemmax,NLAMAX),
     .           Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .           AK(nemmax,NLAMAX), FI_D(nemmax),
     .           D_FI(nemmax,NLAMAX), SF2(2,2), K_LAY(4,4,NLAMAX),
     .           FI(2,3), D_FI_F(2), FXI(2,3), DXI(2)



      ELENTH_X = EX(2) - EX(1)
      ELENTH_Z = EZ(4) - EZ(1)

C --- INITIALIZE THE ARRAYS.

      DO 2 I= 1, 4
         DO 3 J= 1, 4
           DO 4 K= 1, NLAYER(ELMNUM)
              K_LAY(I,J,K) = 0.000
    4      CONTINUE
    3    CONTINUE
    2 CONTINUE


      DO 20 I = 1, NDFELM
```

```
         EF(I) = 0.000
         DO 10 J = 1, NDFELM
            ESTIF(I,J) = 0.000
   10    CONTINUE
   20 CONTINUE


C --- BEGIN GAUSS QUADRATURE LOOP FOR FULL INTEGRATION.

      DO 21 IGP = 1, NGPstf

      XI = GAUSPT(IGP,NGPstf)

      CALL SHPL1D_T (NPEMAX, XI, ELENTH_X, DSF,
     .              GDSF  , JACOBN, SF2, IGP)

        CONST = JACOBN*GAUSWT(IGP,NGPstf)
   21 CONTINUE


      DO 22 I= 1, 4
         DO 23 J= 1, 4
           DO 24 K= 1, NLAYER(ELMNUM)

          CALL  FI_AT_N(I, J, K, ELMNUM, Z1LAY, Z2LAY,AK,
     .                   NEMMAX, NLAMAX, npemax, FI_D, D_FI,
     .                   D_FI_F, FI, FI_N, FXI, DXI, SF2, GDSF)

             H_LAYER  = ( Z2LAY(ELMNUM,K) - Z1LAY(ELMNUM,K) ) / 2

      K_LAY(I,J,K)= H_LAYER/3 *                (
     .        JACOBN * KX(ELMNUM,K) * ( 2*DXI(1)*DXI(2) ) *
     .        ( FI(1,1)*FI(2,1) + 4*FI(1,2)*FI(2,2) + FI(1,3)*FI(2,3) )+
     .       6 * JACOBN * KZ(ELMNUM,K) *
     .       ( FXI(1,1)*FXI(2,1) + FXI(1,2)*FXI(2,2) )*
     .       ( D_FI_F(1) * D_FI_F(2) )    )


          ESTIF(I,J) = ESTIF(I,J) + K_LAY(I,J,K)
   24      CONTINUE
   23    CONTINUE

   22 CONTINUE


   50 CONTINUE


      RETURN
      END

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C     SUBROUTINE:  UBNDRY                                              C
C     PURPOSE:     IMPOSE THE PRESCRIBED DISPLACEMENT B.C.'S ON THE    C
C                  BANDED SYMMETRIC STIFFNESS MATRIX AND MODIFY        C
C                  THE FORCE VECTOR.                                   C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

      SUBROUTINE UBNDRY (NEQMAX, BNDMAX, NBCMAX, NDFNOD, NDFMSH, BNDWTH,
     .                   NSGD   , ISGD   , VSGD   , GSTIF , GF    )
```

```
      IMPLICIT LOGICAL (A-Z)

      INTEGER            NEQMAX, BNDMAX, NBCMAX, NDFNOD, NDFMSH, BNDWTH,
     .                   NSGD  , ISGD  , IBC   , IDOF  , BWM1  , ROW   ,
     .                   COL   , K

      DOUBLE PRECISION   VSGD  , GSTIF , GF    , UVALUE

      DIMENSION ISGD(NBCMAX,2), VSGD(NBCMAX), GSTIF(NEQMAX,BNDMAX),
     .          GF(NEQMAX)


      DO 40 IBC = 1, NSGD
         IDOF   = NDFNOD * (ISGD(IBC,1) - 1) + ISGD(IBC,2)
         UVALUE = VSGD(IBC)
         BWM1   = BNDWTH - 1
         ROW    = IDOF - BNDWTH

         DO 20 K = 1, BWM1
            ROW = ROW + 1
            IF (ROW .GT. 0) THEN
               COL = IDOF - ROW + 1
               GF(ROW) = GF(ROW) - GSTIF(ROW,COL) * UVALUE
               GSTIF(ROW,COL) = 0.000
            END IF
   20    CONTINUE

         GSTIF(IDOF,1) = 1.000
         GF(IDOF) = UVALUE
         ROW = IDOF

         DO 30 K = 2, BNDWTH
            ROW = ROW + 1
            IF (ROW .LE. NDFMSH) THEN
               GF(ROW) = GF(ROW) - GSTIF(IDOF,K) * UVALUE
              GSTIF(IDOF,K) = 0.000
          END IF
   30    CONTINUE
   40 CONTINUE
      RETURN
       END


CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                       C
C     SUBROUTINE:  FI_AT_N                                              C
C     PURPOSE:     CHOOSE THE FUNCTION FI AND XI TO BE USED IN THE      C
C                  EVALUATION OF THE ACTUAL K_LAY                       C
C                                                                       C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

       SUBROUTINE  FI_AT_N (I,J,K, ELMNUM,Z1LAY,Z2LAY,AK,
       .                   NEMMAX, NLAMAX, npemax, FI_D, D_FI,
       .                   D_FI_F, FI, FI_N, FXI, DXI, SF2, gdsf)


      IMPLICIT LOGICAL (A-Z)

      integer            I, J, K, R, Q,
     .                   NLAMAX, NEMMAX, ELMNUM, npemax
```

```
      double precision    Z1LAY,  Z2LAY,
     .                     AK,    FI_D, D_FI,
     .                    FI, D_FI_F, FI_N, ZZ, FXI, DXI,
     .                     gdsf,SF2


      dimension  Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
     .           AK(nemmax,NLAMAX), FI_D(nemmax),
     .           D_FI(nemmax,NLAMAX), gdsf(npemax), SF2(2,2),
     .            FI(2,3), D_FI_F(2), FXI(2,3), DXI(2)



      DO 90 R = 1,2

            IF (I .EQ. 1 .OR. I .EQ. 4) THEN
         DXI(1)    = gdsf(1)
            FXI(1,R) = SF2(1,R)
            ELSE
         DXI(1)    = gdsf(2)
            FXI(1,R) = SF2(2,R)
            END IF

         IF (J .EQ. 1 .OR. J .EQ. 4) THEN
         DXI(2)    = gdsf(1)
            FXI(2,R) = SF2(1,R)
         ELSE
         DXI(2)    = gdsf(2)
            FXI(2,R) = SF2(2,R)
         END IF

 90   CONTINUE

        DO 110 R = 1,3

         FI_N =  0.000
            IF       (R .EQ. 1) THEN
            ZZ= Z1LAY(ELMNUM,K)-Z1LAY(ELMNUM,1)
            ELSE IF (R .EQ. 2) THEN
            ZZ=( (Z2LAY(ELMNUM,K)-Z1LAY(ELMNUM,1)) +
     .            (Z1LAY(ELMNUM,K)-Z1LAY(ELMNUM,1))   )/2
            ELSE IF (R .EQ. 3) THEN
            ZZ= Z2LAY(ELMNUM,K)-Z1LAY(ELMNUM,1)
         END IF

         FI_N =  ZZ
           DO 100 Q = 2, K
           FI_N = FI_N +(ZZ - (Z2LAY(ELMNUM,Q-1)-Z1LAY(ELMNUM,1)))
     .                 *AK(ELMNUM,Q-1)

100        CONTINUE

            IF (I .LT. 3) THEN
         D_FI_F(1) = -D_FI(ELMNUM,K)
            FI(1,R)= (1 - FI_N/FI_D(ELMNUM) )
            ELSE
         D_FI_F(1) = D_FI(ELMNUM,K)
            FI(1,R) = FI_N/FI_D(ELMNUM)
            END IF

         IF (J .LT. 3) THEN
         D_FI_F(2) = -D_FI(ELMNUM,K)
            FI(2,R) = (1 - FI_N/FI_D(ELMNUM) )
```

```
          ELSE
          D_FI_F(2) = D_FI(ELMNUM,K)
            FI(2,R) = FI_N/FI_D(ELMNUM)
          END IF

  110  CONTINUE

       RETURN
       END



CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C                                                                      C
C      SUBROUTINE:  PR_L_T                                             C
C      PURPOSE:     PRINT LAYER SOLUTION                               C
C                                                                      C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC

       SUBROUTINE PR_L_T(OUNIT, NPEMAX, NLAMAX, NELMSH, NEMMAX,
      .                  NLAYER, NEQMAX, Z1LAY, Z2LAY,
      .                  AK, FI_D, NODE, GUL, GF, T_LAY)


       IMPLICIT LOGICAL (A-Z)

       integer            OUNIT ,NPEMAX, NLAMAX, NELMSH,NEMMAX,
        .                   NEQMAX, NLAYER, I , J, K1, K2,NODE

       double precision   Z1LAY, Z2LAY, AK, FI_D,
      .                   GUL, GF, T_B, T_T, FI_N, FI_T, Z_CENT, T_LAY


       DIMENSION  NLAYER(NEMMAX),NODE(NEMMAX,NPEMAX),
      .           Z1LAY(nemmax,NLAMAX), Z2LAY(nemmax,NLAMAX),
       .            AK(nemmax,NLAMAX), FI_D(nemmax), GUL(nemmax,2,NLAMAX),
      .           GF(NEQMAX), T_LAY(nemmax,NLAMAX)


       DO 10 I = 1, NELMSH

         DO 20 J = 1, 2

          IF (J .EQ. 1) THEN
          T_B = GF(NODE(I,1))
          T_T = GF(NODE(I,4))
          FI_T = (Z2LAY(I,1)- Z1LAY(I,1)) / FI_D(I)
          GUL(I,1,1) = T_B * (1 - FI_T) + T_T * FI_T
          FI_T = 0.0000
            DO 30 K1 = 2, NLAYER(I)
              FI_N = Z2LAY(I,K1) - Z1LAY(I,1)
               DO 40 K2 = 2, K1
                 FI_N = FI_N +(Z2LAY(I,K1) - Z2LAY(I,K2-1))*AK(I,K2-1)
   40     CONTINUE
              FI_T = FI_N / FI_D(I)
               GUL(I,1,K1) = T_B * (1 - FI_T) + T_T * FI_T
   30  CONTINUE

          ELSE

          T_B = GF(NODE(I,2))
          T_T = GF(NODE(I,3))
          FI_T = (Z2LAY(I,1)- Z1LAY(I,1))/FI_D(I)
```

```
          GUL(I,2,1) = T_B * (1 - FI_T) + T_T * FI_T
          FI_T = 0.0000
             DO 50 K1 = 2, NLAYER(I)
               FI_N = Z2LAY(I,K1) - Z1LAY(I,1)
                DO 60 K2 = 2, K1
                   FI_N = FI_N +(Z2LAY(I,K1) - Z2LAY(I,K2-1))*AK(I,K2-1)
   60     CONTINUE
               FI_T = FI_N / FI_D(I)
                GUL(I,2,K1) = T_B * (1 - FI_T) + T_T * FI_T
   50  CONTINUE

          END IF
   20    CONTINUE
   10 CONTINUE


       WRITE(OUNIT,1000)

       DO 100        I=1, NELMSH

      WRITE(OUNIT,1010)
       WRITE(OUNIT,1020) I, NLAYER(I)
      WRITE(OUNIT,1030)
                DO 90 J=1, NLAYER(I)
            Z_CENT = Z1LAY(I,J) + (Z2LAY(I,J) - Z1LAY(I,J))/2
           WRITE(OUNIT,1040)  J, Z2LAY(I,J), GUL(I,1,J), GUL(I,2,J),
       .                        Z_CENT, T_LAY(I,J)


90         CONTINUE
100     CONTINUE

C --- FORMAT STATEMENTS.

 1000 FORMAT(' ',///3X,'SOLUTION AT EACH LAYER: ')
 1010 FORMAT(' ',//6X,'ELEMENT',3X,'LAYER',/)
 1020 FORMAT(' ',9X,I3,7X,I3)
 1030 FORMAT(' ',/6X,'LAYER N.',5X,'Z LAYER',6X,'SIDE 1-4',6X,
     .                'SIDE 2-3',7X,'Z(CENTER)',7X,'T(CENTER)'/)
 1040 FORMAT(' ',9X,I3,4X, E10.4,4X, E11.6,4X, E11.6,3X,E10.4,4X, E14.6)


      RETURN
      END
```

REFERENCES

REFERENCES

[1] P. C. Yang, C. H. Norris and Y. 1966. Stavsky. "Elastic Wave Propagation in Heterogeneous Plates", *Int. J. Sol. Structures*, Vol. 2:665-684.

[2] M. Ganapathi, M. Touratier. 1997. "A Study on Thermal Postbuckling Behavior of Laminated Composite Plates Using a Shear-Flexible Finite Element", *Finite Element in Analysis and Design,* Vol. 28:115-135.

[3] J.N. Reddy. 1990. "A Review of Refined Theories of Laminated Composite Plate", *Shock Vibr. Dig.,* Vol. 22:3-17.

[4] T. Kant, R. K. Khare. 1994. "Finite Element Thermal Stress Analysis of Composite Laminates Using a Higher-Order Theory", *Journal of Thermal Stresses,* 17:229-255.

[5] K. N. Cho, A. G. Striz, C. W. Bert. 1989. "Thermal Stress Analysis of Laminate Using High-Order Theory in Each Layer", *Journal of Thermal Stresses,* Vol. 12:321-332.

[6] R. Rolfes. 1991. "Higher Order Theory and Finite Element for Heat Conduction in Composites", *Numerical Methods in Thermal Problems, Proceedings of the Seventh International Conference,* Vol. VII, Part 2:880-889. R.W. Lewis, J.H. Chin, d G.M. Homsy.

[7] A.K. Noor and L.H. Tenek. 1992. "Steady-State Nonlinear Heat Transfer in Multi-layered Composite Panels", *ASCE Journal of Engineering Mechanics,* 118:1661-1679.

[8] R. Rolfes. 1990. "Efficient Thermal Analysis of Anisotropic Composite Plates Using New Finite Elements", Developments in the Science and Technology of Composite Materials, Proceedings of the Fourth European Conference on

Composite Materials, 743-748. J. Fuller, G. Gruninger, K. Schulte, A.R. Bunsell, and A. Massiah.

[9] N. Mukherjee, K. Sinha. 1993. "A Finite Elements Analysis of Inplane Thermostructural Behavior of Composite Plates", *Journal of Reinforced Plastic and Composites,* Vol. 12:1026-1042.

[10] K. Matsumoto, M. Zako. 1995. "Study on Three Dimensional Thermal Stress Analysis for Laminated Composite Materials", *ASME 1995,* PVP-Vol. 302:75-80.

[11] J. Argyris, L. Tenek, F.Öberg. 1995. "A Multilayer Composite Triangular Element for Steady-State Conduction/Convection/Radiation Heat Transfer in Complex Shells", *ASCE Computer Method in Applied Mechanics and Engineering,* 120:271-301.

[12] K. Tamma, A. Yurko. 1989. "Finite-Element Thermal Modeling/Analysis Formulations for Layered Composite Materials", *Numerical Heat Transfer,* Part B vol. 15:73-97.

[13] R. Rolfes, A. K. Noor, H. Sparr. 1997. "Evaluation of Transverse Thermal Stresses in Composite Plates Based on First-Order Shear Deformation Theory", *ASME 1997,* PVP-Vol. 369:167-184.

[14] A. K. Noor, Y. H. Kim, J. M. Peterson. 1994. "Transverse Shear Stresses and Their Sensitivity in Multilayered Composite Plates", *AIAA Journal,* Vol. 32 No. 6:1259-1269.

[15] J.N. Reddy. 1987. "Generalization of Two-Dimensional Theories of Laminated Composite Plates", *Comm. Appl. Num. Meths.,* Vol. 3:173-180.

[16] A. Toledano, H. Murakami. 1987. "A Composite Plates Theory for Arbitrary Laminate Configuration", *J. App. Mech.,* Vol. 54:181-189.

[17] C.Y. Lee, D. Liu. 1992. "An Interlaminar Shear Stress Continuity Theory for Laminated Composite Analysis", *Computers & Structures,* Vol. 42 No. 1:69-78.

[18] X. Lu, D. Liu. 1992. "An Interlaminar Shear Stress Continuity Theory for Both Thin and Thick Composite Laminate", *J. App. Mech.,* Vol. 59:502-509.

[19] N. Watanabe, K. Teranishi. 1995. "Thermal Stresses Analysis for Al Honeycomb Sandwich Plates With Very Thin CFRP Faces", *AIAA*, 1334-CP:1955-1963.

[20] A. K. Pandey, P. Dechaumphait, Allan R. Wieting. 1992. "Thermal-Structural Finite Element Analysis Using Linear Flux Formulation", *Journal of Thermophysic and Heat Transfer*, Vol. 6 No. 2:341348.

[21] M. DiSciuva. 1987. "An Improved Shear-Deformation Theory for Moderately Thick Multilayered Anisotropic Shells and Plates", *Journal of Applied Mechanics*, 54:589-596.

[22] M. DiSciuva. 1985. "Development of an Anisotropic Multilayered Shear Deformable Rectangular Plate Element", *Computers & Structures*, Vol. 21:789-796.

[23] R.C. Averill. 1994. "Static and Dynamic Response of Moderately Thick Laminated Beams with Damage", *Composites Engineering*, 4:3 81-3 95.

[24] R.C. Averill, Y.C, Yip. 1996. "Development of Simple, Robust Finite Elements Based on Refined Theories for Thick Laminated Beams", *Computers & Structures*, 59:529-546.

[25] Y.C. Yip, R.C. Averill. 1996. "Thick Beam Theory and Finite Element Model with Zig-Zag Sublaminate Approximations", *AIAA Journal*, 34:1627-1632.

[26] Y.C. Yip, R.C. Averill. 1997. "A 3D Laminated Plate Finite Element Model with Zig-Zag Sublaminate Approximations", *submitted for publication*.

[27] Y.B, Cho, R.C. Averill. 1997. "An Improved Theory and Finite Element Model for Laminated Beams Using First Order Zig-Zag Sublaminate Approximations", *Composite Structures*, 37:281-298.

[28] M. Cho and R.R. Parmerter. 1993. "Efficient Higher Order Composite Plate Theory for General Lamination Configurations", *AIAA Journal*, 31:1299-1306.

[29] M. DiSciuva. 1993. "A General Quadrilateral Multilayered Plate Element with Continuous Interlaminar Stresses", *Computers & Structures*, Vol. 47:91-105.

[30] Murakami. 1986. "Laminated Composite Theory with improved In-plane Response", *J. App. Mech.*, Vol. 53:661-666.

[31] Y.B, Cho, R.C. Averill. 1999. "First Order Zig-Zag Sublaminate Plate Theory and Finite Element Model for Laminated Composite and Sandwich Panels", *accepted for publication*.

[32] M. Savoia and J.N. Reddy. 1995. "Three-dimensional Thermal Analysis of Laminated Composite Plates", *Int. J. Solid Structures,* Vol. 32 No. 5:593-608.

[33] A. Pantano, R.C. Averill. 1999. "An Improved Thermal Lamination Theory and Finite Element Model for Analysis of Heat Transfer in Composite Structures", Submitted to the *Journal of Composite Materials* in May 1999.

[34] G. Prathap, b. P. Naganarayana. 1995. "Consistent Thermal Stress Evaluation in Finite Elements", *Computers & Structures,* Vol. 54 No. 3:415-426.

[35] G. Prathap, P. Naganarayana. 1997. "Accurate Thermal Stress Predictions Using C^0-continuous High-Order Shear Deformable Elements", *Computers Meth. Appl. Mech. Eng.,* Vol. 144:61-75.

[36] Maenghyo Cho, Heung Suk Park. 1997. "Comments on Three-dimensional Thermal Analysis of Laminated Composite Plates", *Int. J. Solid Structures,* Vol. 34 No. 35-36:4633-4651.

[37] A. Pantano, R.C. Averill. 1999. "A First Order Zig-Zag Sublaminate Model for Analysis of Thermal Stresses in Laminated Composite and Sandwich Panels", Submitted to the *Journal of Sandwich Structures and Materials,* in October 1999.

[38] ANSYS Inc. . 1996. User's Manual, edition 5.3.

[39] R. M. Jones. 1975. "Mechanics of Composite Materials", Taylor & Francis.

[40] E. Reissner. 1986. "On a Mixed Variational Theorem and Shear Deformation Rectangular Plate Theory", *Int. J. Num. Meth. Eng.,* Vol. 23:93-98.

[41] A. Tessler, T. J. R. Huges. 1983. "An Improved Treatment of Transverse Shear in Mindlin-Type Four-Node Quadrilateral Element", *Computer Method In Applied Mechanics and Eng.*, Vol. 39:311-335.

[42] G. Prathap, B. R. Somashekar. 1988. "Field and Edge Consistency Synthesis of a 4-Noded Quadrilateral Plate Bending Element", *Int. J. Num. Meth. Eng.*, Vol. 26:1693-1708.

[43] J. Barlow. 1976. "Optimal Stress Locations in Finite Elements Models", *Int. J. Num. Meth. Eng.*, Vol. 10:243-251.

[44] Dan Zenkert. 1997. "Introduction to Sandwich Construction", *EMAS Publishing.*

[45] L. C. Davis, B. E. Atrz. 1995. "Thermal Conductivity of Metal-Matrix Composites", *J. App. Phys.*, Vol. 77:4954-4960.

[46] M. M. Yovanovich. 1970. "On the Temperature Distribution and Constriction Resistance in Layered Media", *Composite Material,* Vol. 4:567-570.

[47] P. S. Ramensh, M. H. Lean. 1991. "An indirect Boundary Element Method for Heat Transfer in Composites Structures", *Proc. Of the Seventh Int. Conference held in Stanford "Numerical Method in Thermal Problem",* Vol. 7 part. 2:890-897.

[48] L. Xingde. 1995. "Temperature Calculation in the Thickness Direction for Composite of Arbitrary Layers", *Computers and Structures,* Vol. 62 N. 4:763-770.

[49] L. Wang, C. Wang. 1993. "Analysis of Thermal Stresses in Composite Laminates by Assumed Stress Hybrid Multilayer Element", *Journal of Thermal Stresses,* Vol. 13:103-112.

[50] K. D. Cole. 1998. "Thermal Properties of Honeycomb Panels – Analysis of Transient Experiments with One-Dimensional Models", *Procedings of the ASME Heat Transfer Division,* Vol. 5:83-93.

[51] V. Vihak, M. Yuzvyak, A. Yasinskij. 1998. "The Solution of the Plane Thermoelasticity Problem for a Rectangular Domain", *Journal of Thermal Stresses,* Vol. 21:545-561.

[52] N. H. Huang, T. R. M. Tauchert. 1991. "Thermal Stresses in Doubly-Curved Cross-ply Laminates", Journal of Solid Structures, Vol. 29 No. 8:991-1000.

[53] N. L. Breivik, M. W. Heyer. 1998. "Buckling and Postbuckling Behavior of Curved Composite Panels Due and Mechanical Loading", *J. of Reinforced Plastics and Composites*, Vol. 17 No. 14:1292-1306.

[54] Z. Guiping, T. Limin. 1993. "A Semi-Analytical Solution for Thermal Stress Analysis of Laminated Composite Plate in the Hamiltonian System", *Computers and Structures,* Vol. 55 N. 1:113-118.

[55] T. Hause, L. Librescu, T. F. Johnson. 1998. "Thermomechanical Load-Carring Capacity of Sandwich Flat Panels", *Journal of Thermal Stresses,* Vol. 21:627-653.

[56] J. A. Eberhardt, J. E. Locke. 1994. "Thermal-Stress Analysis Using of Anisotropic Plates with Temperature Dependent Material Properties and Free Edge Boundary Conditions", *AIAA,* 1376-CP:552-558.

[57] T. E. Tay. 1992. "Finite Element Analysis of Thermoelastic Coupling in Composite", *Computers and Structures,* Vol. 43 No. 1:107-112.

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

[59] J. G. Crose, R. L.Holman, N. J. Pagano. 1987. "Validation of Advanced Composite Thermal Stress Analysis Method", *Transaction of the ASME,* Vol. 109:40-46.