INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

Université d'Ottawa · University of Ottawa

THE COMPUTER ORIENTED KANTOROVICH-FINITE DIFFERENCE METHOD AND ITS APPLICATION ON BRIDGE ENGINEERING

by

Junqing Zhao

Thesis submitted to the School of Graduate Studies
in partial fulfillment of the requirements
for the degree of doctor of philosophy in Civil Engineering
under the auspices of the Ottawa-Carleton Institute for
Civil Engineering

@Junqing Zhao, Ottawa, Ontario, Canada

May, 1997

National Library of Canada

Bibliothèque nationale du Canada

Acquisitions and Bibliographic Services

Acquisitions et services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-21980-1

Canada

ACKNOWLEDGEMENTS

My special thanks are extended to the thesis supervisors, Dr. M.S. Cheung and Dr. S.F.Ng, for their support, advice and patience. This thesis could not have been done without their direction. I would like to express my gratitude to members of my thesis advisory committee, Dr. D. Lau, Dr. A.G. Razaqpur, and Dr.M. Saatcioglu, for their excellent teaching and pedagogy.

My deep thanks go to my family members, especially my wife Jianxiu, for their encouragement throughout my life and during the long years of study and research.

ABSTRACT

The Kantorovich - finite difference method has been developed and applied for analysis and design of bridge structures in this thesis. Firstly, the bridge deck is mapped into an unit square in the ξ-η plane. Secondly, the governing partial differential equation of the deck is reduced to the ordinary differential equation in the longitudinal direction of the bridge by the routine Kantorovich method. Finally, the finite difference method is employed to solve the above-derived ordinary differential equation to get the displacements and internal forces. The present method has been examined on right, skewed or general slab bridge decks and box-girder bridges. Mindlin plate theory and Kirchhoff plate theory are incorporated into the differential equation and, as a result, the designer has the option of either including or excluding the effect of shear deformation of the plate in the analysis. Possible shear locking is avoided by the reduced integration technique. Numerical examples show that the proposed new numerical model is versatile, efficient and reliable.

TABLE OF CONTENTS

List of Figures

List of Tables

NOMENCLATURE

a	length in x direction of the skewed slab
b	length in y direction of the skewed slab
{d}	displacement function along the longitudinal direction
D	$= Et^3 / 12(1-\nu^2)$
[D]	elastic matrix
E	Young's modulus
E_{bi} ,E_{si}	strain-displacement relationship matrix
{F}	force vector
G	shear modulus, $=E/[2(1+\nu)]$
h_j	length of jth strip
I	unit matrix
[J]	Jacobian matrix
|J|	Jacobian determinate
k	coordinate transformation coefficient, $=\cot\theta$
K	stiffness matrix
L	span of the plate or the beam
L_{bi} ,L_{si}	operator matrix
m	number of longitudinal sections in a strip
{M}	moment vector

{N}	membrane force vector
N_i	shape function
q	loading per unit area
{Q}	shear force vector
R	curvature radius of circular plate
t	thickness
[T]	coordinate transformation matrix
u,v,w	displacement in x, y and z direction
α	the angle between local coordinate axis x' and global coordinate axis
$\{\delta\}$	vector of displacement vector
$\{\epsilon\}$	strain vector
$\{\kappa\}$	flexural strain vector
$\{\gamma\}$	shear strain vector
ν	Poisson's ratio
ϕ_i	B_3 spline function centered at η_i
θ	skew angle
ψ_x, ψ_y	rotations
λ	distance between the two adjacent nodes in the longitudinal direction
ξ, η	natural co-ordinate
Π	potential energy functional

Chapter 1

INTRODUCTION

§1-1 Bridge Structure And Its Numerical Methods of Analysis

In modern highway bridge construction, the use of irregular bridge decks along with varying cross section and arbitrary supporting conditions is inevitable. Some traditional and popular methods of analysis, such as 'load distribution technique' [1-2] , 'approximate stiffness method' [3] and 'grillage analysis approach' [4-5] are only suitable in assessing the overall behavior of some specified bridges. Therefore, their applicability is limited. In view of this, the development of an efficient and reliable method of analysis for general bridge structures remains important.

The finite difference method, in which the governing differential equation and boundary conditions are replaced with difference equations, is another classical powerful mathematical model approximation. This method can be employed for a variety of structural problems. However, difficulty arises when complex geometrical configuration and boundary conditions are involved . This drawback has been limiting it from being widely used for the analysis of complex structures.

Modern numerical methods have been playing an increasingly important role with the development of the digital computer since 1950s. Among them, the finite element method [6-15] is perhaps the most powerful approach and is now widely used for the analysis of almost all types of structural problems. In implementation, the structure may be modelled using one, two, or three dimensional elements or combinations of these elements. However, for bridges with length much greater than their width, the use of two or three dimensional plate finite elements is often cumbersome and inefficient. Alternatively, the beam element approach treats bridges as one dimensional structures and such a model is generally not accurate enough for most bridge structures as it cannot give any information on the distribution of displacements and stresses in the transverse direction.

§1-2 Semi-analytical Approaches and the Kantorovich Method

An efficient alternative is to use semi-analytical methods, where the bridge is treated as a two dimensional structure with a bias. In other words, the longitudinal direction of the bridge is treated as the dominant direction and, as a result, high accuracy can be obtained in that direction. This approach is logical because for most types of bridges, the distribution of displacements and stresses along the longitudinal direction is more important than that in the transverse direction.

The semi-analytical finite strip method introduced by Y.K.Cheung in the 1960s [16-17] has been successfully used in the analysis of regular shaped structures. Fundamentally, the finite strip method is a hybrid of the orthogonal plate method and the finite element concept. The finite strip method has now been well accepted as a powerful tool for analyzing bridge structures, since it can significantly reduce the requirements on computer data input, computation time and storage requirements. The first application of the finite strip method on bridge structure was given by Cheung himself [18]. Powell and Ogden [19] extended its application to slab-beam bridges in 1969. A curved strip was adopted to the solution of certain curved bridge decks by Cheung [20]. In Ref.[21-25], the method was applied to continuous, folded and box-girder bridges. In spite of a number of advantages, the semi-analytical finite strip method has some drawbacks especially in dealing with multiple spans, discrete supports or general curved bridges. The textbooks published by Cheung [26], Loo and Cuseas [27] give excellent reviews on the finite strip method.

More recently, the spline finite strip method [28] was developed to make the conventional finite strip method more versatile by introducing the mathematical tool of B_3 spline function. Ensuring continuity up to the second derivative, the B_3 spline function is the smoothest interpolating function compared with other third order piecewise polynomial interpolating functions. In Chapter six of this study, a sister Kantorovich method called the spline Kantorovich method is developed by the help of B_3 spline function. When using the B_3 spline function, the penalty function approach [29] can be easily adopted to impose all different types of boundary conditions. In implementation, the spline finite strip method assembles the structural stiffness element by element, similar to that of the finite element method. Most recent success of the spline finite strip method included its application to box-girder bridges in 1983 [30], to skewed bridges in 1988 [31], to curved slab bridges in 1986 [32-33] and to nonlinear analysis of curved concrete bridges in 1993 [34].

The method of line is a classical semi-analytical method [35-40]. Ref.[35] summaries the theory and application of this method as a result of the earlier research work. The main point behind this method is to reduce the structural governing partial differential equations into ordinary differential equations on the chosen lines by finite difference augmentation. Little or no progress was noted in any research on the semi-analytical method of line in recent past. The method has the same drawback as most other semi-analytical methods, that is, the method is suitable only for regular shaped structures. In addition, the accuracy of method relies on the number of chosen lines and can be costly if a large number of lines is required.

The finite element method of lines [41-43] should be mentioned as another good example of a semi-analytical model. The successful application of this method, however, depends on the availability of computer software to solve the ordinary differential equations.

A simple and economical semi-analytical method, known as the Kantorovich method was developed in the 1950s [44]. It has been proven by Kantorovich and Krylov mathematically that the convergence speed of the Kantorovich method is log(n) times higher than that of the Ritz method, where n is the number of terms of the power adopted as the displacement trial function. The Ritz method forms the basis of the widely used finite element method, in which the problem of finding the minimum of a functional of a double integral is reduced to the problem of minimizing a functional of several variables. This is accomplished by choosing the form of the solution a priori and obtaining the best values of the constants utilizing minimum energy principles. In the Kantorovich method, the problem of finding the minimum of a double integral is reduced to the problem of minimizing a simple integral. As a result, only part of the expression giving the solution needs to be chosen a priori and other parts of the functions can be determined in accordance with the character of the problem. Nevertheless, the Kantorovich method also experienced difficulties for structures with irregular geometries, with concentrated or patch loadings, and with discrete supports.

§ 1-3 General Philosophy of the Kantorovich - Finite Difference Method

To overcome the difficulties and to retain the advantages of the Kantorovich method, the Kantorovich - finite difference method [45-46] is developed in this thesis. Firstly, the deck of the bridge is mapped from the x-y Cartesian coordinate system into an unit square in ξ-η coordinate system by means of the transformation function. Usually the cubic shape transformation function is sufficiently accurate for bridge decks.

Secondly, the ordinary differential equation can be derived by means of the routine Kantorovich method and expressed in terms of the total displacement vector $\{d\}$.

Finally the finite difference method is employed to solve the derived ordinary differential equation. In the finite difference method, a mesh size must first be chosen. Then the numerical solution of the differential equation is determined at m chosen points referred to as nodal points.

From the features of the Kantorovich - finite difference method, we may predict the following advantages when used in the analysis of bridge structures: 1) The method is very versatile since irregular and curved boundaries can be fitted easily by introducing coordinate transformation. 2) The input data are minimal compared with the conventional finite element method since nodal line displacements instead of nodal point displacements are chosen as unknowns. 3) The method can easily deal with various supporting and loading conditions. 4) Since no displacement function along the longitudinal direction of the structure is assumed, the accuracy is higher than most other numerical methods. 5) Convergence is easily assessed by reducing the mesh size of the finite difference.

§1-4 Organization of the Thesis

Chapter 1 gives a brief review of the existing analysis approaches for bridge structures. A general introduction of the proposed Kantorovich finite difference method is then presented. Chapter 2 presents with the analysis of a general slab and goes on to deal with shear deformation based on Mindlin plate theory. Numerical results have demonstrated clearly that the present method yields much more accurate results when compared with some other numerical methods of analysis. In Chapter 3, the application of the Kantorovich - finite difference method on skewed bridge decks is described. In this case, the ordinary differential equation can be expressed in explicit form. To extend the present method to the analysis of box-girder bridges, a model for plane stress/strain problems is first formulated in Chapter 4. Consequently, the present method can be employed to analyze 'shell' structures by combining the membrane and bending parts together.

A special chapter is devoted to a related numerical approach with the present method, that is, the spline Kantorovich method [47]. Firstly, the governing partial differential equation of the plate is reduced to the ordinary differential equation in the longitudinal direction of the bridge by the routine Kantorovich method as previously done. Then the B_3 spline function is adopted as the displacement trial function. And finally the point collocation method is utilized to solve the derived differential equation. The penalty function approach is readily adopted to impose any types of boundary conditions.

The conclusions and recommendations for further study are discussed in Chapter six.

Chapter 2

ANALYSIS OF GENERAL BRIDGE DECK BY THE KANTOROVICH - FINITE DIFFERENCE METHOD BASED ON THE MINDLIN PLATE THEORY

§2-1 Introduction

In the domain of plate analysis, there are two main plate theories that are widely used. One is the Kirchhoff thin plate theory and the other is the Mindlin moderately thick plate theory. Since only the first order differential is involved in the energy functional of the Mindlin plate theory, C^0 continuity is required when assuming the displacement function. This characteristic leads to more and more applications of this theory to the numerical analysis of the plates.

As a semi-analytical numerical method, the Kantorovich method developed in the 1950s can be an efficient tool for plate analysis. The main theme behind it is that the problem of finding the minimum of a double integral is reduced to the problem of minimizing a simple integral. As a result, only part of the expression giving the solution is chosen a priori, with other parts of the functions being determined in accordance with the character of the problem. In the finite element method, however, the whole solution has to be chosen a priori and the best values of the constants are then obtained by utilizing the principle of minimum potential energy. That is the reason why the Kantorovich method can achieve much higher accuracy than the finite element method. Nevertheless, the conventional Kantorovich method experienced difficulties for structures with irregular geometries, with concentrated or patching loadings, and with discrete supports.

To overcome these difficulties and to retain the advantages of the Kantorovich method, the Kantorovich - finite difference method is formulated in this chapter to analyze general bridge decks. Firstly, the bridge deck is mapped into a unit square in ξ-η coordinates. Subsequently, the ordinary differential equation is derived by means of the routine Kantorovich method. Finally the finite difference method is employed to solve the derived ordinary differential equation. Mindlin plate theory is adopted and deflection w, nodal line rotations ψ_x and ψ_y are taken as independent unknown displacement functions, so that the shear deformation effect is taken into account. Possible shear locking is avoided by the technique of selected reduced integration. Even with less degrees of freedom used for the same problem, numerical results have shown that the present method yields more accurate results when compared with some other numerical methods of analysis.

§2-2 Coordinate Transformation

In general, bridge decks are bounded by two parallel curved sides along the free edges while the abutments are straight but skewed. Thus, when using the Kantorovich - finite difference method, it is easy to formulate the strips along the longitudinal direction.

The bridge deck can be mapped from the x-y Cartesian coordinate system into the natural coordinate ξ-η plane by means of the transformation function. It is believed that the cubic shape function as expressed below is sufficiently accurate for most purposes.

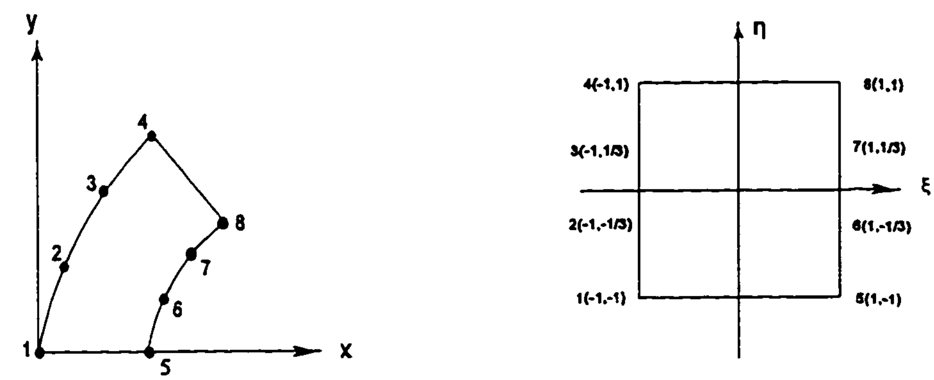


Fig.2-1 The mapping of the curved bridge deck

$$x=\sum_{i=1}^{8} N_i(\xi,\eta)x_i$$

$$y=\sum_{i=1}^{8} N_i(\xi,\eta)y_i \qquad (2\text{-}1)$$

where (x_i, y_i) represent coordinates of the ith control point on the curved edges, N_i (ξ,η) is the corresponding shape transformation function as given in Table 2-1.

Table 2-1. Shape function for coordinate transformation

Control Points	Ni (ξ, η)
1,4,5,8	$1/32\ (1+\xi_i\xi)(9\eta^2-1)(1+\eta_i\eta)$
2,3,6,7	$9/32\ (1+\xi_i\xi)(1-\eta^2)(1+9\eta_i\eta)$

We have:

$$\begin{Bmatrix} \dfrac{\partial}{\partial \xi} \\ \dfrac{\partial}{\partial \eta} \end{Bmatrix} = [J] \begin{Bmatrix} \dfrac{\partial}{\partial x} \\ \dfrac{\partial}{\partial y} \end{Bmatrix} \tag{2-2}$$

and

$$\begin{Bmatrix} \dfrac{\partial}{\partial x} \\ \dfrac{\partial}{\partial y} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \dfrac{\partial}{\partial \xi} \\ \dfrac{\partial}{\partial \eta} \end{Bmatrix} \tag{2-3}$$

$$[J] = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} \tag{2-4}$$

where [J] and $[J]^{-1}$ are the Jacobian matrix and its inverse respectively. And,

$$|J| = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi} \tag{2-6}$$

$$[J]^{-1} = \frac{1}{|J|} \begin{bmatrix} \dfrac{\partial y}{\partial \eta} & -\dfrac{\partial y}{\partial \xi} \\ -\dfrac{\partial x}{\partial \eta} & \dfrac{\partial x}{\partial \xi} \end{bmatrix} \tag{2-5}$$

If the geometrical shape of the bridge deck is very complicated, the spline function can be adopted to map it from the Cartesian coordinate x-y plane into a unit square in the natural coordinate ξ-η plane.

§2-3 The Displacement, Strain And Stress of the Element Strip

The deflection w and the rotations ψ_x and ψ_y of an element strip may be expressed as:

$$\{u\}=\begin{Bmatrix} w \\ \psi_x \\ \psi_y \end{Bmatrix}=[N_1(\xi)I_{3x3},\ N_2(\xi)I_{3x3}]\begin{Bmatrix} \{d_1\} \\ \{d_2\} \end{Bmatrix}=[N]\{d\}^e \tag{2-7}$$

where

$$N_1=1-\xi$$
$$N_2=\xi \tag{2-8}$$

in which $\xi = (\xi - \xi_j)/h_j$, $h_j = \xi_{j+1} - \xi_j$ and $\{d_i\} = [\omega_i(\eta), \psi_{xi}(\eta), \psi_{yi}(\eta)]^T$, i=1,2.

The flexural strain $\{\kappa\}$ and shear strain $\{\gamma\}$ can be written in terms of the operator matrices $[L_b]$ and $[L_s]$:

$$\{\kappa\}=[L_b]\{u\}$$
$$\{\gamma\}=[L_s]\{u\} \tag{2-9}$$

where

$$\{\kappa\}=\begin{Bmatrix}\kappa_x\\ \kappa_y\\ 2\kappa_{xy}\end{Bmatrix}=-\begin{Bmatrix}\dfrac{\partial\psi_x}{\partial x}\\ \dfrac{\partial\psi_y}{\partial y}\\ \dfrac{\partial\psi_x}{\partial y}+\dfrac{\partial\psi_y}{\partial x}\end{Bmatrix} \tag{2-10}$$

$$\{\gamma\}=\begin{Bmatrix}\gamma_x\\ \gamma_y\end{Bmatrix}=\begin{Bmatrix}\dfrac{\partial w}{\partial x}-\psi_x\\ \dfrac{\partial w}{\partial y}-\psi_y\end{Bmatrix} \tag{2-11}$$

and

$$[L_b]=-\begin{bmatrix}0 & \dfrac{\partial}{\partial x} & 0\\ 0 & 0 & \dfrac{\partial}{\partial y}\\ 0 & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial x}\end{bmatrix} \tag{2-12}$$

$$[L_s]=\begin{bmatrix}\dfrac{\partial}{\partial x} & -1 & 0\\ \dfrac{\partial}{\partial y} & 0 & -1\end{bmatrix} \tag{2-13}$$

From the coordinate transformation, $[L_b]$ and $[L_s]$ may be rewritten as follows:

$$\{L_b\}=[L_{b1}]\frac{\partial}{\partial\xi}+[L_{b2}]\frac{\partial}{\partial\eta}$$

$$\{L_s\}=[L_{s0}]+[L_{s1}]\frac{\partial}{\partial\xi}+[L_{s2}]\frac{\partial}{\partial\eta} \tag{2-14}$$

in which

$$[L_{b1}]=\frac{1}{|J|}\begin{bmatrix} 0 & -\frac{\partial y}{\partial\eta} & 0 \\ 0 & 0 & \frac{\partial x}{\partial\eta} \\ 0 & \frac{\partial x}{\partial\eta} & -\frac{\partial y}{\partial\eta} \end{bmatrix} \tag{2-15}$$

$$[L_{b2}]=\frac{1}{|J|}\begin{bmatrix} 0 & \frac{\partial y}{\partial\xi} & 0 \\ 0 & 0 & -\frac{\partial x}{\partial\xi} \\ 0 & -\frac{\partial x}{\partial\xi} & \frac{\partial y}{\partial\xi} \end{bmatrix} \tag{2-16}$$

$$[L_{s0}]=\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{2-17}$$

$$[L_{s1}]=\frac{1}{|J|}\begin{bmatrix}\frac{\partial y}{\partial \eta} & 0 & 0\\ -\frac{\partial x}{\partial \eta} & 0 & 0\end{bmatrix} \tag{2-18}$$

$$[L_{s2}]=\frac{1}{|J|}\begin{bmatrix}-\frac{\partial y}{\partial \xi} & 0 & 0\\ \frac{\partial x}{\partial \xi} & 0 & 0\end{bmatrix} \tag{2-19}$$

The first derivatives of $[L_{bi}]$ and $[L_{si}]$ given below will be utilized later.

$$[L'_{b1}]=\frac{1}{|J|}\begin{bmatrix}0 & \frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial y}{\partial \eta}-\frac{\partial^2 y}{\partial \eta^2} & 0\\ 0 & 0 & -\frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial x}{\partial \eta}+\frac{\partial^2 x}{\partial \eta^2}\\ 0 & -\frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial x}{\partial \eta}+\frac{\partial^2 x}{\partial \eta^2} & \frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial y}{\partial \eta}-\frac{\partial^2 y}{\partial \eta^2}\end{bmatrix} \tag{2-20}$$

$$[L'_{b2}]=\frac{1}{|J|}\begin{bmatrix}0 & \frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial y}{\partial \xi}-\frac{\partial^2 y}{\partial \eta \partial \xi} & 0\\ 0 & 0 & -\frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial x}{\partial \xi}+\frac{\partial^2 x}{\partial \eta \partial \xi}\\ 0 & -\frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial x}{\partial \xi}+\frac{\partial^2 x}{\partial \eta \partial \xi} & \frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial y}{\partial \xi}-\frac{\partial^2 y}{\partial \eta \partial \xi}\end{bmatrix} \tag{2-21}$$

$$[L_{s1}'] = \frac{1}{|J|}\begin{bmatrix} \frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial y}{\partial \eta} - \frac{\partial^2 y}{\partial^2 \eta} & 0 & 0 \\ -\frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial x}{\partial \eta} + \frac{\partial^2 x}{\partial^2 \eta} & 0 & 0 \end{bmatrix} \tag{2-22}$$

$$[L_{s2}'] = \frac{1}{|J|}\begin{bmatrix} \frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial y}{\partial \xi} - \frac{\partial^2 y}{\partial \eta \partial \xi} & 0 & 0 \\ -\frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial x}{\partial \xi} + \frac{\partial^2 x}{\partial \eta \partial \xi} & 0 & 0 \end{bmatrix} \tag{2-23}$$

Substituting Eq.(2-7) and Eq.(2-14) to (2-19) into Eq.(2-9), we obtain :

$$\begin{aligned} \{\kappa\} &= [E_{b1}]\{d\}^e + [E_{b2}]\{d'\}^e \\ \{\gamma\} &= [E_{s1}]\{d\}^e + [E_{s2}]\{d'\}^e \end{aligned} \tag{2-24}$$

where

$$\begin{aligned} [E_{b1}] &= [L_{b1}][N'] \\ [E_{b2}] &= [L_{b2}][N] \end{aligned} \tag{2-25}$$

and

$$[E_{s1}] = [L_{s0}][N] + [L_{s1}][N'] \tag{2-26}$$

$$[E_{s2}] = [L_{s2}][N]$$

The moment vector {M} and the shear force vector {Q} can be obtained by means of Hooke's law.

$$\{M\}=[M_x, M_y, M_{xy}]^T$$
$$\{Q\}=[Q_x, Q_y]^T \tag{2-27}$$

and

$$[D_b]=\frac{Et^3}{12(1-\upsilon^2)}\begin{bmatrix}1 & \upsilon & 0\\ \upsilon & 1 & 0\\ 0 & 0 & \frac{1-\upsilon}{2}\end{bmatrix} \tag{2-28}$$

$$[D_s]=\kappa Gt\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} \tag{2-29}$$

where k=1/1.2 and G=E/[2(1+υ)].

§2-4 The Element Strip's Potential Energy Functional

The element potential energy functional of the Mindlin plate can be expressed as:

$$\Pi^e=\frac{1}{2}\iint_{A_e}(\{\kappa\}^T[D_b]\{\kappa\}+\{\gamma\}^T[D_s]\{\gamma\})dA-\iint_{A_e}\{u\}^T\{q\}dA \tag{2-30}$$

where the first two terms represent the bending and shear potential energy respectively. The last term denotes the potential energy of the external load {q}.

Substituting Eq.(2-24) into the above expression yields:

$$
\begin{aligned}
\Pi^e = & \frac{1}{2}\iint_{A_e}\left(\{d\}^{eT}[E_{b1}]^T[D_b][E_{b1}]\{d\}^e + 2\{d\}^{eT}[E_{b1}]^T[D_b][E_{b2}]\{d'\}^e\right. \\
& \left.+\{d'\}^{eT}[E_{b2}]^T[D_b][E_{b2}]\{d'\}^e\right)dA + \frac{1}{2}\iint_{A_e}\left(\{d\}^{eT}[E_{s1}]^T[D_s][E_{s1}]\{d\}^e\right. \\
& \left.+2\{d\}^{eT}[E_{s1}]^T[D_s][E_{s2}]\{d'\}^e + \{d'\}^{eT}[E_{s2}]^T[D_s][E_{s2}]\{d'\}^e\right)dA - \iint_{A_e}\{d\}^{eT}[N]^T\{q\}dA
\end{aligned}
\tag{2-31}
$$

Performing the integration with respect to ξ, the problem is reduced to that of the minimum of a simple integral as follows:

$$
\Pi^e = \frac{1}{2}\int_{-1}^{1}\left(\{d'\}^{eT}[A]^e\{d'\}^e + 2\{d\}^{eT}[B]^e\{d'\}^e + \{d\}^{eT}[C]^e\{d\}^e\right)d\eta - \int_{-1}^{1}\{d\}^{eT}\{F\}^e d\eta \tag{2-32}
$$

where

$$
\begin{aligned}
[A]^e &= h_j\int_0^1\left([E_{b2}]^T[D_b][E_{b2}] + [E_{s2}]^T[D_s][E_{s2}]\right)|J|\,d\xi \\
[B]^e &= h_j\int_0^1\left([E_{b1}]^T[D_b][E_{b2}] + [E_{s1}]^T[D_s][E_{s2}]\right)|J|\,d\xi \\
[C]^e &= h_j\int_0^1\left([E_{b1}]^T[D_b][E_{b1}] + [E_{s1}]^T[D_s][E_{s1}]\right)|J|\,d\xi \\
\{F\}^e &= h_j\int_0^1[N]^T\{q\}\,|J|\,d\xi
\end{aligned}
\tag{2-33}
$$

It should be mentioned that $[A]^e$, $[B]^e$ and $[C]^e$ are functions of coordinate η only. Their values will be determined once the specific value of η is given.

§2-5 The Derivation of the Governing Ordinary Differential Equation

Since the displacements between the element strips are compatible, the total potential energy of the whole structure can be obtained by summing the potential energy of all element strips.

$$\Pi=\sum_e \Pi^e=\frac{1}{2}\int_{-1}^{1}(\{d'\}^T[A]\{d'\}+2\{d\}^T[B]\{d'\}+\{d\}^T[C]\{d\})d\eta-\int_{-1}^{1}\{d\}^T\{F\}d\eta \qquad (2\text{-}34)$$

in which all matrices and vectors shown can be obtained by assembling the corresponding matrices and vectors of element strips.

Taking variation about nodal line displacement of the whole bridge deck {d}, we obtain:

$$\delta\Pi(\{d\})=\int_{-1}^{1}(-\delta\{d'\}^T[A]\{d'\}+\delta\{d\}^T[B]\{d'\}+\delta\{d'\}^T[B]^T\{d\}+\delta\{d\}^T[C]\{d\})d\eta-\int_{-1}^{1}\delta\{d\}^T\{F\}d\eta \qquad (2\text{-}35)$$

Integrating by parts, the above expression becomes:

$$\begin{aligned}\delta\Pi(\{d\})=\int_{-1}^{1}\Big(&-\delta\{d\}^T([A]\{d''\}+[A']\{d'\})+\delta\{d\}^T[B]\{d'\}-\delta\{d\}^T([B]^T\{d'\}+[B']^T\{d\})\\&+\delta\{d\}^T[C]\{d\}\Big)d\eta-\int_{-1}^{1}\delta\{d\}^T\{F\}d\eta\end{aligned} \qquad (2\text{-}36)$$

Or:

$$\delta\Pi(\{d\})=-\int_{-1}^{1}\delta\{d\}^T([A]\{d''\}+([A']+[B]^T-[B])\{d'\}+([B']^T-[C])\{d\}+\{F\})d\eta \qquad (2\text{-}37)$$

Enforcing $\delta\Pi(\{d\})=0$, the corresponding ordinary differential equation will be:

$$[\bar{A}]\{d''\}+[\bar{B}]\{d'\}+[\bar{C}]\{d\}+\{F\}=0 \qquad (2\text{-}38)$$

in which

$$\begin{aligned}
&[\bar{A}]=[A]\\
&[\bar{B}]=[A']+[B]^T-[B]\\
&[\bar{C}]=[B']^T-[C]
\end{aligned} \qquad (2\text{-}39)$$

§2-6 Solving the Ordinary Differential Equation by the Finite Difference Method

In the finite difference method, a mesh size must first be chosen. Then the numerical solution of the differential equation is determined at the chosen points referred to as nodal points. The number of simultaneous equations is therefore equal to the number of chosen points. The derivatives of the displacement $\{d\,(\eta)\}$ at node k can be approximated by:

$$\begin{aligned}
&\{d'\}(\eta_\kappa)=\frac{1}{2\lambda}(\{d\}_{\kappa+1}-\{d\}_{\kappa-1})\\
&\{d''\}(\eta_\kappa)=\frac{1}{\lambda^2}(\{d\}_{\kappa-1}+\{d\}_{\kappa+1}-2\{d\}_\kappa)
\end{aligned} \qquad (2\text{-}40)$$

where λ is the constant spacing between the nodal points.

Substituting the above expressions into Eq.(2-38) yields:

$$[\bar{A}]_\kappa \frac{1}{\lambda^2}(\{d\}_{\kappa-1}+\{d\}_{\kappa+1}-2\{d\}_\kappa)+[\bar{B}]_\kappa \frac{1}{2\lambda}(\{d\}_{\kappa+1}-\{d\}_{\kappa-1})+[\bar{C}]_\kappa\{d\}_\kappa+\{F\}_\kappa=0 \tag{2-41}$$

Or:

$$(2[\bar{A}]_\kappa-\lambda[\bar{B}]_\kappa)\{d\}_{\kappa-1}+(-4[\bar{A}]_\kappa+2\lambda^2[\bar{C}]_\kappa)\{d\}_\kappa+(2[\bar{A}]_\kappa+\lambda[\bar{B}]_\kappa)\{d\}_{\kappa+1}+2\lambda^2\{F\}_\kappa=0 \quad \kappa=1,2,3,\ldots\ldots,m+1 \tag{2-42}$$

Eq.(2-42) is the standard finite difference equations used to solve the ordinary differential equation (2-38).

By noting:

$$\begin{aligned} [\hat{A}]_\kappa &= 2[\bar{A}]_\kappa-\lambda[\bar{B}]_\kappa \\ [\hat{B}]_\kappa &= -4[\bar{A}]_\kappa+2\lambda^2[\bar{C}]_\kappa \\ [\hat{C}]_\kappa &= 2[\bar{A}]_\kappa+\lambda[\bar{B}]_\kappa \\ \{\hat{F}\}_\kappa &= 2\lambda^2\{F\}_\kappa \end{aligned} \tag{2-43}$$

Equation (2-42) can be rewritten as:

$$[\hat{A}]_\kappa\{d\}_{\kappa-1}+[\hat{B}]_\kappa\{d\}_\kappa+[\hat{C}]_\kappa\{d\}_{\kappa+1}+\{\hat{F}\}_\kappa=0 \quad \kappa=1,2,3,\ldots,m+1 \tag{2-44}$$

Once we get the displacement vector {d}, the bending moments and shear forces in Eq.(2-27) can be obtained as:

$$\{M\}_k=[D_b](-\frac{1}{2\lambda}[E_{b2}]\{d\}^e_{k-1}+[E_{b1}]\{d\}^e_k+\frac{1}{2\lambda}[E_{b2}]\{d\}^e_{k+1})$$
$$\{Q\}_k=[D_s](-\frac{1}{2\lambda}[E_{s2}]\{d\}^e_{k-1}+[E_{s1}]\{d\}^e_k+\frac{1}{2\lambda}[E_{s2}]\{d\}^e_{k+1}) \qquad (2\text{-}45)$$

§2-7 Numerical Examples

Several numerical examples have been included to demonstrate the accuracy and versatility of the present method.

1. Simply supported square plate is subjected to uniform distributed load q with thickness/span ratio of 0.3 (Fig.2-2). In Table 2-2, the convergence of the method is examined by calculating central deflection and central bending moments with different numbers of strips and finite difference sections. It can be seen that all cases converge to the analytical solution. In addition, the accuracy of longitudinal moment M_{yc} is higher than that of the transverse moment M_{xc}, which conforms with our previous prediction.

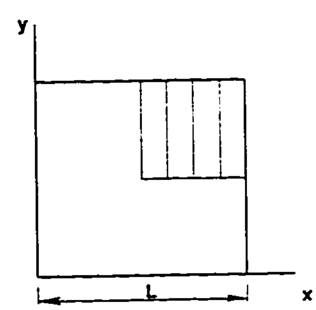


Fig.2-2 Simply supported square plate

Table 2-2 Simply Supported Square Plate

Strips*Sections	w_c	M_{xc}	M_{yc}
2*2	0.5306	3.839	4.062
2*4	0.5564	4.450	4.422
4*4	0.5766	4.568	4.622
4*6	0.5870	4.680	4.699
6*6	0.5899	4.694	4.717
Analytical Solution [48]	0.5957	4.79	4.79
Multiplier	$0.01qL^4/D$	$0.01qL^2$	

2. A clamped circular plate with radius of R is subjected to uniformly distributed load q. Two different plates with thickness/span ratio of 0.2 and 0.01 are analyzed. The Poission ratio is taken as 0.3. The results on deflection and bending moments at central point C and boundary point A are shown in Table 2-3. It can be seen that the present method can yield very accurate results for structures with curved boundary.

Table 2-3 Clamped Circular Plate

Mesh (Strips*Sections)	t/R=0.2			t/R=0.01		
	w_c	M_{yc}	M_{xa}	w_c	M_{yc}	M_{xa}
4*4	0.0178	0.0800	-0.1233	0.0155	0.0892	-0.1413
6*6	0.0182	0.0807	-0.1247	0.0159	0.0834	-0.1334
Thin Plate Theory [49]	0.0165	0.0813	-0.125	0.0165	0.0813	-0.125
Thick Plate Theory [48]	0.0184	0.0813	-0.125	0.0165	0.0813	-0.125
Multiplier	qR^4/D	qR^2		qR^4/D	qR^2	

3. A uniformly loaded trapezoidal plate with all boundaries clamped (Fig.2-3). Six strip and six section mesh (105 degrees of freedom) is employed. The deflection results are tabulated in Table 2-4 and compared with those of Ref. [50].

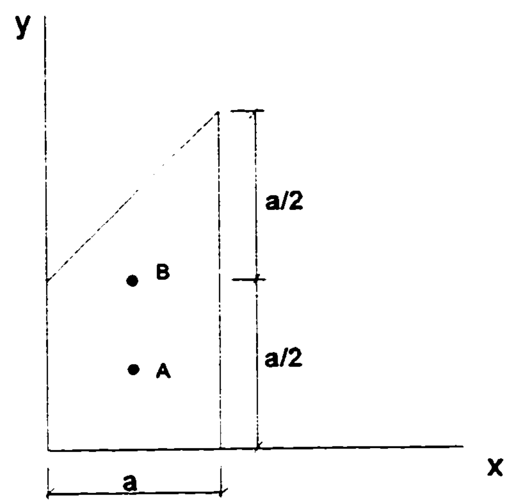


Fig.2-3 The clamped trapezoidal plate

Table 2-4. Deflection Result of the Trapezoidal Plate

Methods	Point A	Point B
Spline sub-area method [50]	2.0658	-
FEM (12 elements) [50]	1.9643	-
FEM (96 elements)*	2.332	-
Present method	2.363	1.914
Multiplier	$10^{-3}qa^4/D$	

* By using Program of SAP 84

4. A 60° fan-shaped bridge deck (Fig. 2-4) The internal and external radii are 7 inches and 13 inches respectively. The bridge is simply supported along its two ends. Concentrated loads are applied at three different locations (Points A, B and C). 12x8 mesh is used and the mid-span deflections resulting are tabulated in Table 2-5 and compared with results of the finite element method [51-52]. All results are in good agreement.

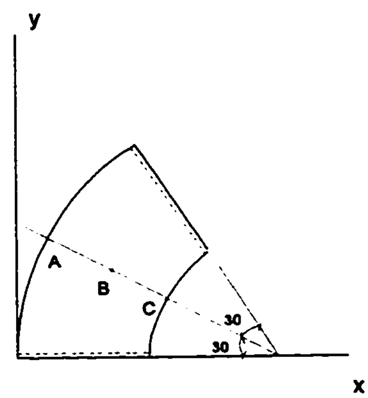


Fig 2-4 Fan-shaped Bridge deck

§2-8 Summary

Compared with other numerical methods, the present method has the following advantages: (1) Since curved boundaries can be fitted easily by introducing coordinate transformation, the method is very versatile. (2) The input data are minimal compared with the conventional finite element method. (3) The method can deal with various supporting and load conditions. (4) Convergence is easily investigated by reducing the mesh size of the finite difference.

The accuracy and efficiency of this method were demonstrated by a variety of examples. Excellent agreements are found throughout. It can be concluded that the Kantorovich - finite difference method provides an efficient numerical tool for the analysis of slab brdige decks.

Table 2-5 Deflection across mid-span of fan-shaped plate

Loading Position	Radius (in.)	Present Method	Coull [51]	Sawko [52]
At Point A	7	0.01944	0.0194	0.0198
	9	0.03540	0.0353	0.0355
	11	0.05787	0.0578	0.0582
	13	0.08801	0.0876	0.0888
At Point B	7	0.01555	0.0155	0.0154
	9	0.02411	0.0241	0.0241
	11	0.03434	0.0342	0.0344
	13	0.04588	0.0457	0.0459
At Point C	7	0.01700	0.0169	0.0168
	9	0.01525	0.0157	0.0150
	11	0.01637	0.0163	0.0163
	13	0.01944	0.0195	0.0194

$E=4.6\times10^5$ lb/in^2, $\nu=0.35$, $t=0.168$in, $p=1$lb

Chapter 3

THE ANALYSIS OF SKEWED BRIDGE DECK BY THE KANTOROVICH - FINITE DIFFERENCE METHOD BASED ON THE KIRCHHOFF PLATE THEORY

§3-1 Introduction

The Kantorovich method was initially developed by taking advantage of the regular shape of certain classes of structures. However, this semi-analytical method, which reduces the two dimensional partial differential governing equation into a one dimensional ordinary differential equation by means of the structural energy functional, suffers from many drawbacks when dealing with concentrated forces, multispan or skewed structures.

Finite difference method is used extensively for solving complex differential equation. It can be considered as a means for obtaining the approximate solutions of differential equations, when exact solutions to such equations are not possible. According to the method, we first superimpose on the region of interest a network. The partial derivatives in the equation are then replaced by the difference quotients, converting the differential equation to a difference equation. We then use the difference equation and the given data to determine the function values at the intersections (nodes) of the grid.

By combining the conventional Kantorovich method with the finite difference method, the Kantorovich - finite difference method is developed. Unlike the conventional Kantorovich method, the ordinary differential equation is derived in the ξ-η coordinate system instead of the x-y coordinate system. The finite difference method is then applied to solve the ordinary differential equation derived from the governing partial differential equation after the application of the Kantorovich method. The present method is free from the drawbacks of the conventional Kantorovich method. The accuracy requirement can be easily met by reducing the mesh size of finite difference. The main theme of this chapter is to extend the method to skewed bridge decks which are widely used in practice. The Kichhoff plate theory is employed in the analysis. A number of examples with different loading and supporting conditions are given to demonstrate the versatility of this method.

§3-2 Statement of the Problem

A parallelogram slab with a skew angle θ is considered (Fig.3-1) with clamped, simply supported or free edges.

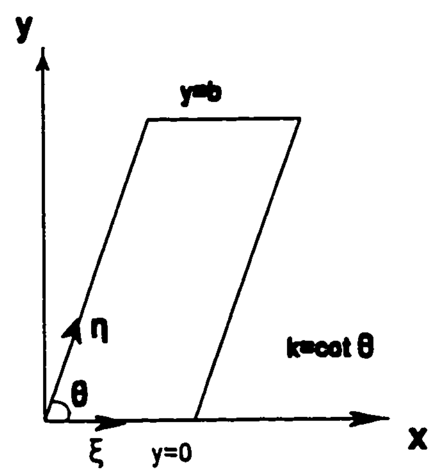


Fig.3-1 A parallelogram slab with a skew angle θ

$$\Pi^e=\frac{1}{2}\iint_{A_e}\{\kappa\}^T[D_b]\{\kappa\}dA-\iint_{A_e}\{\omega\}^T\{q\}dA \tag{3-1}$$

In equation (3-1), the first term represents the bending potential energy and the second term is the potential energy of external load {q}, {w} and {κ} are the deflection vector and the flexural strain vector respectively.

$$[D_b]=\frac{Et^3}{12(1-\upsilon^2)}\begin{bmatrix}1 & \upsilon & 0\\ \upsilon & 1 & 0\\ 0 & 0 & \frac{1-\upsilon}{2}\end{bmatrix} \tag{3-2}$$

The structure can be mapped from the Cartesian coordinate x-y plane into the natural coordinate ξ-η plane by means of the transformation function as follows :

$$\begin{cases} x=a\xi+\kappa b\eta \\ y=b\eta \end{cases} \tag{3-3}$$

In addition, the relationship between the second derivatives of the x-y plane and the ξ-η plane can be easily established as:

$$\begin{Bmatrix}\frac{\partial^2}{\partial x^2}\\ \frac{\partial^2}{\partial y^2}\\ \frac{\partial^2}{\partial x\partial y}\end{Bmatrix}=\begin{bmatrix}\frac{1}{a^2} & 0 & 0\\ \frac{k^2}{a^2} & \frac{1}{b^2} & \frac{-2k}{ab}\\ \frac{-k}{a^2} & 0 & \frac{1}{ab}\end{bmatrix}\begin{Bmatrix}\frac{\partial^2}{\partial\xi^2}\\ \frac{\partial^2}{\partial\eta^2}\\ \frac{\partial^2}{\partial\xi\partial\eta}\end{Bmatrix} \tag{3-4}$$

§3-3 The Displacement, Strain and Stress of the Element Strip

The square region in the ξ-η plane is partitioned into n strips in the ξ direction. Taking deflection $\omega_1(\eta)$, $\omega_2(\eta)$ and normal rotation $\psi_1(\eta)$, $\psi_2(\eta)$ as unknown functions of the element strip, the displacement can be expressed as the product of the unknown function in η direction and conventional beam function (Hermite cubic polynomial) in the other direction, i.e.:

$$\{\omega(\xi,\eta)\}=[N]\{d\}^e \tag{3-5}$$

where

$$[N]=[N_1(\xi),\ h_jN_2(\xi),\ N_3(\xi),\ h_jN_4(\xi)] \tag{3-6}$$

$$\{d\}^e=[\omega_1(\eta),\ \Psi_1(\eta),\ \omega_2(\eta),\ \Psi_2(\eta)]^T \tag{3-7}$$

$$\begin{aligned} N_1(\xi)&=1-3\xi^2+2\xi^3 \\ N_2(\xi)&=\xi(1-2\xi+\xi^2) \\ N_3(\xi)&=3\xi^2-2\xi^3 \\ N_4(\xi)&=\xi(\xi^2-\xi) \end{aligned} \tag{3-8}$$

in which

$$\begin{aligned} \xi&=\frac{\xi-\xi_j}{h_j} \\ h_j&=\xi_{j+1}-\xi_j \end{aligned} \tag{3-9}$$

The flexural strain vector of the skewed bridge is:

$$\{\kappa\} = \begin{Bmatrix} \kappa_x \\ \kappa_y \\ 2\kappa_{xy} \end{Bmatrix} = -\begin{Bmatrix} \dfrac{\partial^2 \omega}{\partial x^2} \\ \dfrac{\partial^2 \omega}{\partial y^2} \\ \dfrac{\partial^2 \omega}{\partial x \partial y} \end{Bmatrix} \tag{3-10}$$

Substituting Eq. (3-4) into Eq.(3-10) yields :

$$\{\kappa\} = -\begin{bmatrix} \dfrac{1}{a^2} & 0 & 0 \\ \dfrac{\kappa^2}{a^2} & \dfrac{1}{b^2} & -\dfrac{2\kappa}{ab} \\ -\dfrac{2\kappa}{a^2} & 0 & \dfrac{2}{ab} \end{bmatrix} \begin{Bmatrix} \dfrac{\partial^2 \omega}{\partial \xi^2} \\ \dfrac{\partial^2 \omega}{\partial \eta^2} \\ \dfrac{\partial^2 \omega}{\partial \xi \partial \eta} \end{Bmatrix} \tag{3-11}$$

Writing { κ } in terms of the nodal line displacement,

$$\{\kappa\} = -\begin{bmatrix} \dfrac{[N'']}{a^2} & 0 & 0 \\ \dfrac{\kappa^2[N'']}{a^2} & \dfrac{[N]}{b^2} & -\dfrac{2\kappa[N']}{ab} \\ -\dfrac{2\kappa[N'']}{a^2} & 0 & \dfrac{2[N']}{ab} \end{bmatrix} \begin{Bmatrix} \{d\}^e \\ \{d''\}^e \\ \{d'\}^e \end{Bmatrix} \tag{3-12}$$

And, expressing it in matrix form:

$$\{\kappa\}=[E_{b1}]\{d\}^e+[E_{b2}]\{d'\}^e+[E_{b3}]\{d''\}^e \tag{3-13}$$

where

$$[E_{b1}]=\begin{bmatrix}-\frac{[N'']}{a^2} & -\frac{\kappa^2[N'']}{a^2} & \frac{2\kappa[N'']}{a^2}\end{bmatrix}^T$$
$$[E_{b2}]=\begin{bmatrix}0 & \frac{2\kappa[N']}{ab} & \frac{-2[N']}{ab}\end{bmatrix}^T \tag{3-14}$$
$$[E_{b3}]=\begin{bmatrix}0 & \frac{-[N]}{b^2} & 0\end{bmatrix}^T$$

The moment vector {M} can be derived using Hooke's law :

$$\{M\}=[D_b]\{\kappa\} \tag{3-15}$$

in which

$$\{M\}=[M_x, M_y, M_{xy}]^T \tag{3-16}$$

§3-4 The Derivation of the Governing Ordinary Differential Equation

Substituting Eq. (3-13) into the potential energy expression Eq.(3-1), we get:

$$\Pi^e=\frac{1}{2}\int_0^1(\{d''\}^{eT}[A_1]^e\{d''\}^e+2\{d'\}^{eT}[B_1]^e\{d''\}^e+2\{d\}^{eT}[C_1]^e\{d''\}^e$$
$$+\{d'\}^{eT}[A_2]^e\{d'\}^e+2\{d\}^{eT}[B_2]^e\{d'\}^e+\{d\}^{eT}[C_2]^e\{d\}^e)d\eta-\int_0^1\{d\}^{eT}\{F\}^e d\eta \qquad (3\text{-}17)$$

in which

$$\begin{aligned}
[A_1]^e&=abh_j\int_0^1[E_{b3}]^T[D_b][E_{b3}]d\xi\\
[B_1]^e&=abh_j\int_0^1[E_{b2}]^T[D_b][E_{b3}]d\xi\\
[C_1]^e&=abh_j\int_0^1[E_{b1}]^T[D_b][E_{b3}]d\xi\\
[A_2]^e&=abh_j\int_0^1[E_{b2}]^T[D_b][E_{b2}]d\xi\\
[B_2]^e&=abh_j\int_0^1[E_{b1}]^T[D_b][E_{b2}]d\xi\\
[C_2]^e&=abh_j\int_0^1[E_{b1}]^T[D_b][E_{b1}]d\xi\\
\{F\}^e&=abh_j\int_0^1[N]^T\{q\}d\xi
\end{aligned} \qquad (3\text{-}18)$$

It should be noted that $[A_i]^e$, $[B_i]^e$ and $[C_i]^e$, i =1 and 2, are constant matrices.

Since the displacements between the element strips are compatible, the total potential energy of the structure can be obtained by summing the potential energy of all element strips.

$$\Pi=\sum_e\Pi^e=\frac{1}{2}\int_0^1(\{d''\}^T[A_1]\{d''\}+2\{d'\}^T[B_1]\{d''\}+2\{d\}^T[C_1]\{d''\}$$
$$+\{d'\}^T[A_2]\{d'\}+2\{d\}^T[B_2]\{d'\}+\{d\}^T[C_2]\{d\})d\eta-\int_0^1\{d\}^T\{F\}d\eta \qquad (3\text{-}19)$$

All matrices and vectors can be obtained by assembling the corresponding matrices and vectors of the element strips.

Taking variation about the nodal line displacement $\{d\}$ of the above expression, we have:

$$
\begin{aligned}
\delta\Pi(\{d\}) = \int_0^1 (&\delta\{d''\}^T[A_1]\{d''\} + \delta\{d'\}^T[B_1]\{d''\} + \{d'\}^T[B_1]\delta\{d''\} \\
&+ \delta\{d\}^T[C_1]\{d''\} + \{d\}^T[C_1]\delta\{d''\} + \delta\{d'\}^T[A_2]\{d'\} + \delta\{d\}^T[B_2]\{d'\} \\
&+ \{d\}^T[B_2]\delta\{d'\} + \delta\{d\}^T[C_2]\{d\})d\eta - \int_0^1 \delta\{d\}^{eT}\{F\}d\eta
\end{aligned}
\tag{3-20}
$$

Integrating by parts, the above expression becomes:

$$
\begin{aligned}
\delta\Pi(\{d\}) = \int_0^1 &\delta\{d\}^T([A_1]\{d''''\} + ([B_1]^T - [B_1])\{d'''\} + ([C_1] + [C_1]^T - [A_2])\{d''\} \\
&+ ([B_2] - [B_2]^T)\{d'\} + [C_2]\{d\})d\eta - \int_0^1 \delta\{d\}^T\{F\}d\eta
\end{aligned}
\tag{3-21}
$$

Enforcing $\delta\Pi(\{d\}) = 0$, the governing ordinary differential equation is obtained as follows:

$$
[\bar{A}]\{d''''\} + [\bar{B}]\{d'''\} + [\bar{C}]\{d''\} + [\bar{D}]\{d'\} + [\bar{E}]\{d\} - \{F\} = 0 \tag{3-22}
$$

where

$$
\begin{aligned}
&[\bar{A}]=[A_1]\\
&[\bar{B}]=[B_1]^T-[B_1]\\
&[\bar{C}]=[C_1]+[C_1]^T-[A_2] \qquad (3\text{-}23)\\
&[\bar{D}]=[B_2]-[B_2]^T\\
&[\bar{E}]=[C_2]
\end{aligned}
$$

§3-5 Solving the Ordinary Differential Equation by the Finite Difference Method

Denoting λ the distance between two nodes, a mesh is first chosen. Numerical solution of the differential equation for displacement is then determined at m chosen points referred to as nodal points. The derivatives of the displacement $\{d(\eta)\}$ at node k can be approximated by:

$$
\begin{aligned}
&d'_{\kappa}(\eta)=\frac{1}{2\lambda}(-\{d\}_{\kappa-1}+\{d\}_{\kappa+1})\\
&d''_{\kappa}(\eta)=\frac{1}{\lambda^2}(\{d\}_{\kappa-1}-2\{d\}_{\kappa}+\{d\}_{\kappa+1})\\
&d'''_{\kappa}(\eta)=\frac{1}{2\lambda^3}(-\{d\}_{\kappa-2}+2\{d\}_{\kappa-1}-2\{d\}_{\kappa+1}+\{d\}_{\kappa+2}) \qquad (3\text{-}24)\\
&d''''_{\kappa}(\eta)=\frac{1}{\lambda^4}(\{d\}_{\kappa-2}-4\{d\}_{\kappa-1}+6\{d\}_{\kappa}-4\{d\}_{\kappa+1}+\{d\}_{\kappa+2})
\end{aligned}
$$

Substituting these expressions into Eq.(3-22) yields:

$$[\hat{A}]\{d\}_{\kappa-2}+[\hat{B}]\{d\}_{\kappa-1}+[\hat{C}]\{d\}_{\kappa}+[\hat{D}]\{d\}_{\kappa+1}+[\hat{E}]\{d\}_{\kappa+2}-\{F\}=0 \qquad \kappa=1,2,3,\ldots,m+1 \tag{3-25}$$

Eq.(3-25) is the standard finite difference equation used to solve the ordinary differential equation (3-22).

And,

$$\begin{aligned}
[\hat{A}]&=\frac{[\bar{A}]}{\lambda^4}-\frac{1}{2}\frac{[\bar{B}]}{\lambda^3}\\
[\hat{B}]&=-4\frac{[\bar{A}]}{\lambda^4}+\frac{[\bar{B}]}{\lambda^3}+\frac{[\bar{C}]}{\lambda^2}-\frac{1}{2}\frac{[\bar{D}]}{\lambda}\\
[\hat{C}]&=6\frac{[\bar{A}]}{\lambda^4}-2\frac{[\bar{C}]}{\lambda^2}+[\bar{E}]\\
[\hat{D}]&=-4\frac{[\bar{A}]}{\lambda^4}-\frac{[\bar{B}]}{\lambda^3}+\frac{[\bar{C}]}{\lambda^2}+\frac{1}{2}\frac{[\bar{D}]}{\lambda}\\
[\hat{E}]&=\frac{[\bar{A}]}{\lambda^4}+\frac{1}{2}\frac{[\bar{B}]}{\lambda^3}
\end{aligned} \tag{3-26}$$

Once we get the displacement vector {d}, the bending moment expressed by Eq. (3-15) at node k can be rewritten as follows:

$$\{M(\xi,\eta_k)\}=[D_b]([E_{b1}]\{d\}^e_k+[E_{b2}]\{d'\}^e_k+[E_{b3}]\{d''\}^e_k) \tag{3-27}$$

Substituting Eq.(3-24) into Eq.(3-27), the bending moment is expressed as:

$$\{M(\xi,\eta_k)\}=[E_{m1}]\{d\}^e_{k-1}+[E_{m2}]\{d\}^e_k+[E_{m3}]\{d\}^e_{k+1} \tag{3-28}$$

$$\begin{aligned}
[E_{m1}] &= -\frac{[E_{b2}]}{2\lambda}+\frac{[E_{b3}]}{\lambda^2} \\
[E_{m2}] &= [E_{b1}]-2\frac{[E_{b3}]}{\lambda^2} \\
[E_{m3}] &= \frac{[E_{b2}]}{2\lambda}+\frac{[E_{b3}]}{\lambda^2}
\end{aligned} \tag{3-29}$$

§3-6 Numerical Examples

According to the formulation given in the above section, a computer program has been coded to solve the bending problems of skewed bridge decks.

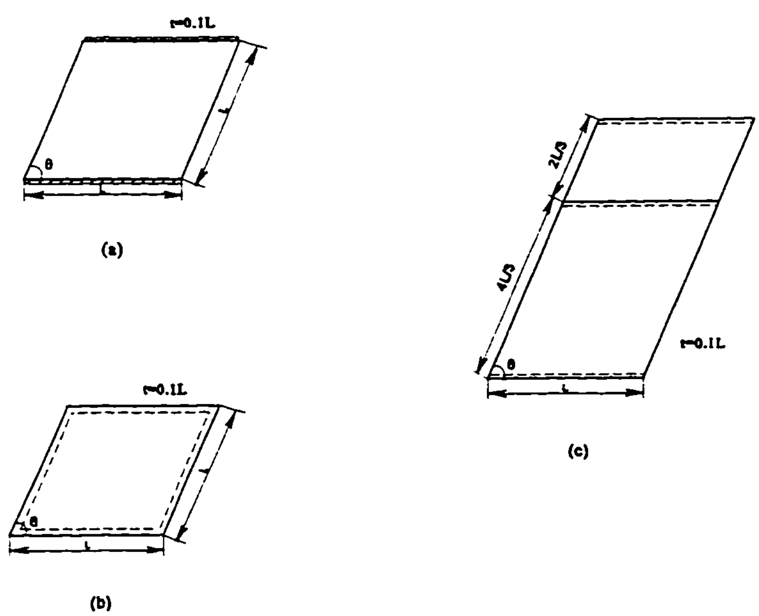


Figure 3-2 Skewed bridge decks

1. Clamped plate: A uniform thickness parallelogram plate with two parallel fixed edges and two free edges under a uniformly distributed load q is tested (Fig.3-2(a)). The skew angle θ is varied from 90^0 (right plate) to 30^0 and the Possion's ratio is taken as 0.15. The mesh used is 6 strips by 8 sections for the whole plate. The variation of central deflection and bending moment at the midpoint with the skew angle are compared with those from Ref.[53] in Fig.3-3. Close agreements for both deflections and moments are found.

2. Four side simply supported plate Fig3-2(b): Both uniformly distributed load and point load (at plate center) cases are considered and the mesh used is the same as that in example 1. The variations of central deflection and central bending moment at the midpoint of the plate with the angle of skew are compared in Table 3-1 with the finite element's results of Ref.[49]. Again, good agreements are found.

Table 3-1 Results of four side simply supported slab (ν=0.3)

$W_c = \alpha q L^4 \sin\theta / D$, $\quad W_c = \alpha P$

$M_c = \beta q L^2 \sin\theta$, $\quad M_c = \beta P$

θ		Uniformly Distributed		Concentrated	
		α	β	α	β
90^0	Ref.[49]	0.0040	0.0479	0.01	0.35
	Present	0.0040	0.0480	0.01	0.35
75^0	Ref.[49]	0.0037	0.0476	0.01	0.35
	Present	0.0037	0.0473	0.01	0.35
60^0	Ref.[49]	0.0029	0.0463	0.00	0.35
	Present	0.0028	0.0455	0.00	0.34
45^0	Ref.[49]	0.0017	0.0426	0.00	0.34
	Present	0.0017	0.0417	0.00	0.33
30^0	Ref.[49]	0.0007	0.0339	0.00	0.32
	Present	0.0006	0.0324	0.00	0.32

3. Continuous plates Fig.3-2(c): 8 strip and 12 section mesh is adopted for the analysis. The deflection and moment curves along the central line are plotted in Fig. 3-4 and Fig.3-5 respectively. Again, the results are quite satisfactory compared with those from the finite element method of Ref.[49].

§3-7 Summary

The Kantorovich-finite difference method can be successfully applied to the analysis of skewed bridge decks with various skew and boundary conditions. The requirement of input data is minimal. In addition, this method only requires two parameters per node for interpolation and thus is definitely an advantage over the standard finite element method, which requires three parameters per node for interpolation. It is anticipated that this method will offer a competitive alternative to the finite element method for the analysis of the skewed bridge decks.

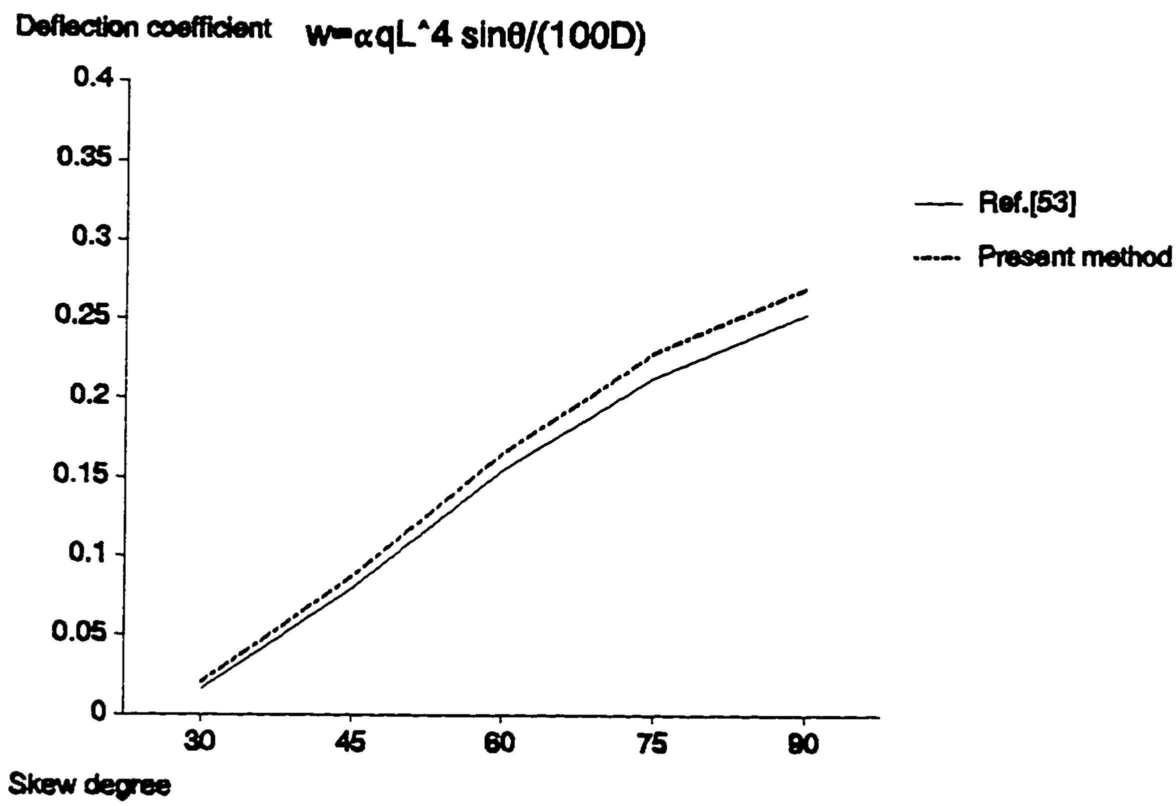


Fig.3-3(a) Deflection coefficient for clamped skew slab

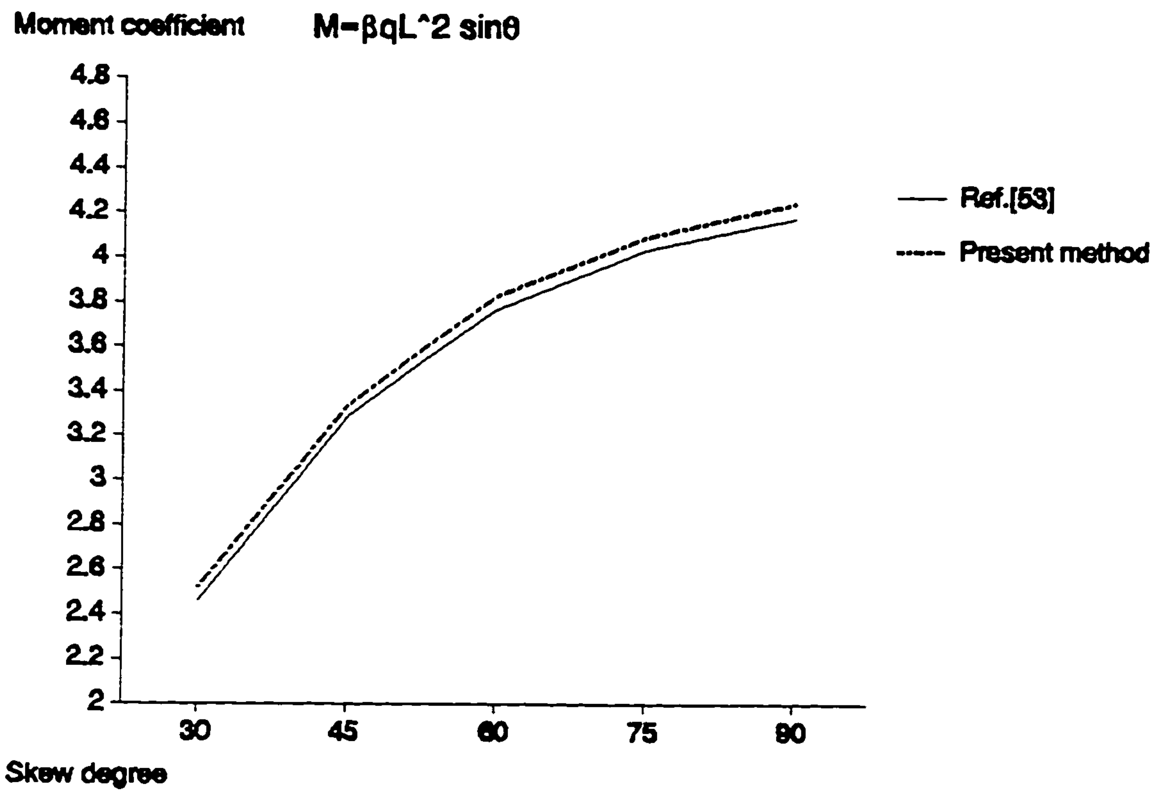


Fig.3-3(b) Moment coefficient for clamped skew slab

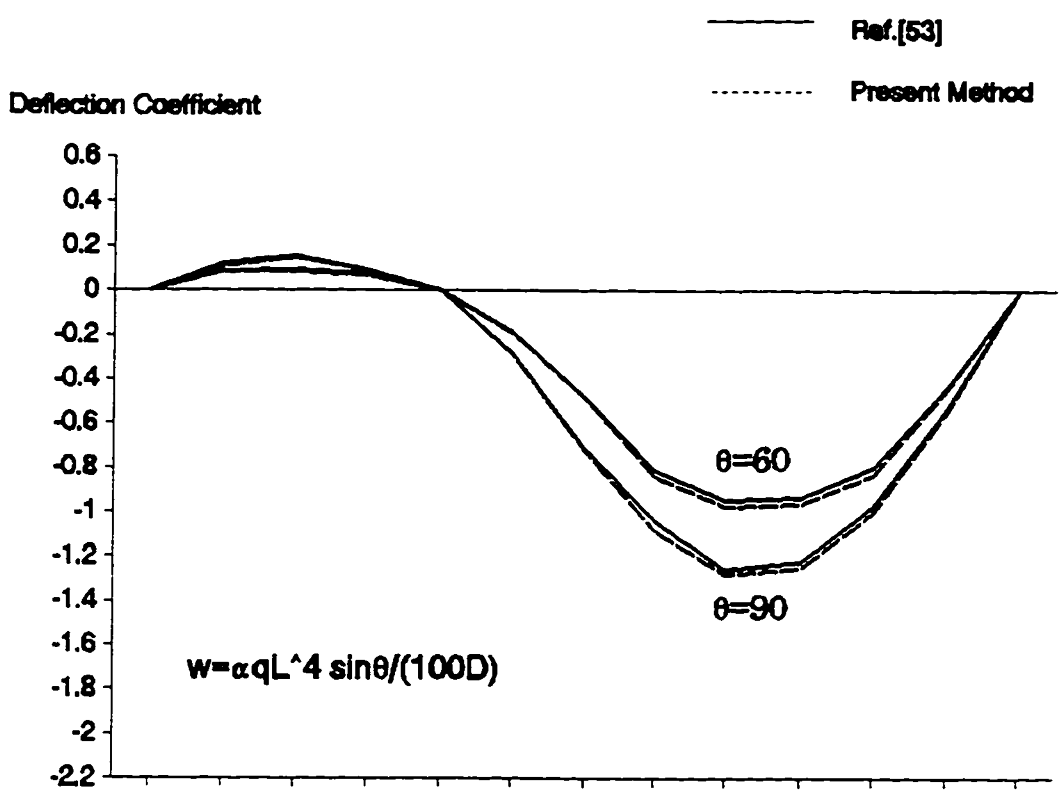


Fig.3-4 Deflection distribution along central line

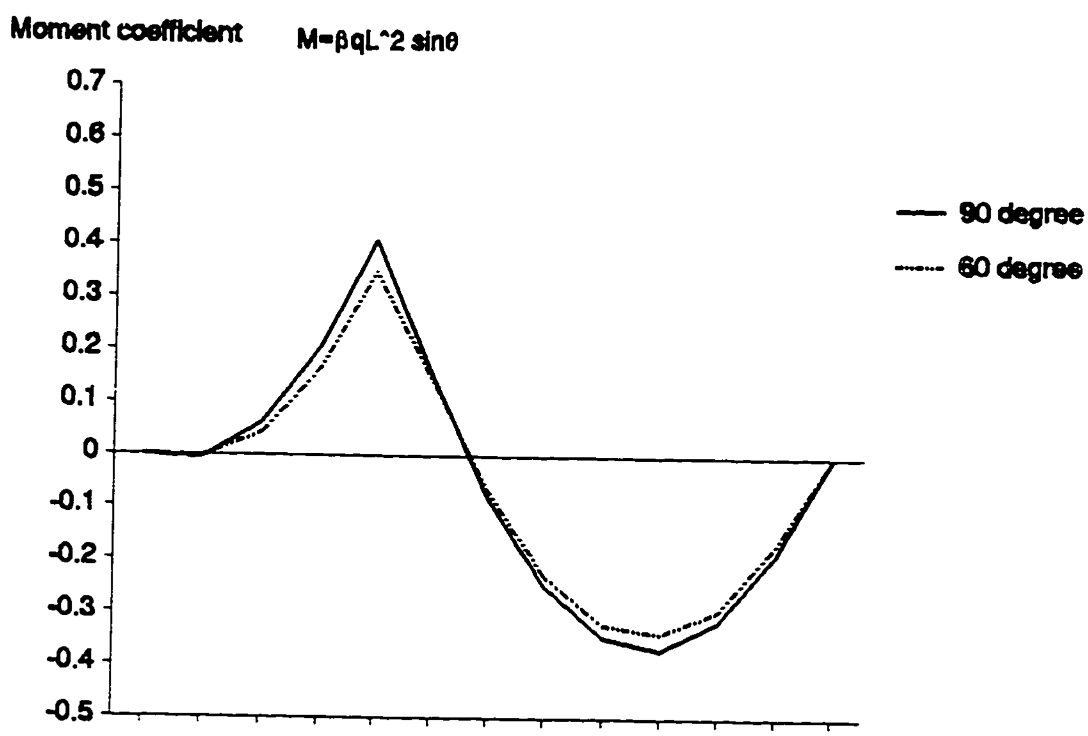


Fig.3-5 Moment distribution along central line

Chapter 4

THE KANTOROVICH - FINITE DIFFERENCE METHOD AND THE ANALYSIS OF BOX-GIRDER BRIDGE

§4-1 Introduction

Bridge decks may be classified into three main categories, slab bridge decks, slab-on-girder and box-girder bridge decks. The Kantorovich-finite difference method has been developed and applied for the analysis of slab bridge decks in the previous two chapters. Unlike most of the simi-analytical numerical approaches, this method can also be used for bridges with arbitrary geometries, with complicated loading and supporting conditions. Numerical examples showed that it is reliable and efficient when applied to general slab bridge decks. The main theme of this chapter is to extend the method to box girder bridges which are widely used in practice.

To obtain satisfactory results, the Kantorovich - finite difference method requires only minimum computer time and data preparation. In addition, the Kantorovich - finite difference method can overcome difficulties encounted by the conventional semi-analytical numerical methods such as the finite strip method. In this analysis, the Kirchhoff thin plate theory is employed as the plates forming box girder bridges are usually thin enough so that the effect of the shear deformation can be ignored. Numerical example shows that the present method can achieve reasonable accuracy while requiring much less input data.

§4-2 The Membrane Strain And Stress of the Element Strip

First of all, the coordinate transformation is adopted as in previous chapters. The box girder bridge is subjected to in-plane stresses and out-of-plane bending forces (Fig. 4-1). For convenience, we discuss the plane stress and plate bending stress separately.

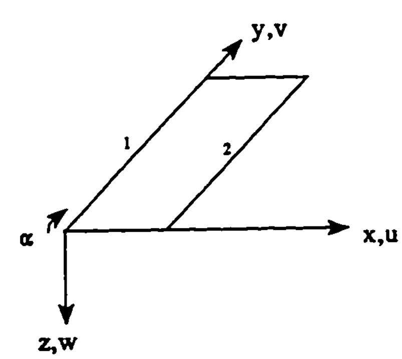


Fig.4-1 Elemental strip for box girder bridge

Taking the plane displacement $u_i(\eta)$ and $v_i(\eta)$ i=1,2 as unknown functions of the element strip, the membrane displacement u (ξ,η) can be expressed as the product of unknown function in η direction and polynomial in the other direction, i.e.:

$$\{u(\xi,\eta)\}=[N]\{d_m\}^e \tag{4-1}$$

where

$$
\begin{aligned}
&\{u(\xi,\eta)\}=[\{u\},\ \{v\}]^T\\
&[N]=[N_1(\xi)I_{2x2}\ \ N_2(\xi)I_{2x2}]\\
&\{d_m\}^e=[\{d_1\},\ \{d_2\}]^T\\
&\{d_i\}=[u_i,\ v_i]^T \qquad i=1,2
\end{aligned}
\tag{4-2}
$$

and

$$
\begin{aligned}
&N_1(\xi)=1-\xi\\
&N_2(\xi)=\xi
\end{aligned}
\tag{4-3}
$$

in which

$$
\xi=\frac{\xi-\xi_j}{\xi_{j+1}-\xi_j} \tag{4-4}
$$

The membrane strain vector is:

$$
\{\varepsilon_m\}=\begin{Bmatrix}\varepsilon_x\\ \varepsilon_y\\ \varepsilon_{xy}\end{Bmatrix}=\begin{Bmatrix}\dfrac{\partial u}{\partial x}\\ \dfrac{\partial v}{\partial y}\\ \dfrac{\partial u}{\partial y}+\dfrac{\partial v}{\partial x}\end{Bmatrix} \tag{4-5}
$$

In matrix form, $\{\varepsilon_m\}$ can be written as:

$$
\{\varepsilon_m\}=[L_m]\{u\} \tag{4-6}
$$

where

$$[L_m]=\begin{bmatrix} \frac{\partial}{\partial x} & 0 \\ 0 & \frac{\partial}{\partial y} \\ \frac{\partial}{\partial y} & \frac{\partial}{\partial x} \end{bmatrix} \tag{4-7}$$

From the coordinate transformation, [L_m] may be rewritten as follows:

$$[L_m]=[L_{m1}]\frac{\partial}{\partial \xi}+[L_{m2}]\frac{\partial}{\partial \eta} \tag{4-8}$$

in which

$$[L_{m1}]=\frac{1}{|J|}\begin{bmatrix} -\frac{\partial y}{\partial \eta} & 0 \\ 0 & \frac{\partial x}{\partial \eta} \\ \frac{\partial x}{\partial \eta} & -\frac{\partial y}{\partial \eta} \end{bmatrix} \tag{4-9}$$

$$[L_{m2}]=\frac{1}{|J|}\begin{bmatrix} \frac{\partial y}{\partial \xi} & 0 \\ 0 & -\frac{\partial x}{\partial \xi} \\ \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \end{bmatrix} \tag{4-10}$$

Also, the first derivatives of $[L_{m1}]$ and $[L_{m2}]$ given below are used when deriving the governing ordinary differential equation.

$$[L'_{m1}] = \frac{1}{|J|}\begin{bmatrix} \frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial y}{\partial \eta} - \frac{\partial^2 y}{\partial \eta^2} & 0 \\ 0 & -\frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial x}{\partial \eta} + \frac{\partial^2 x}{\partial \eta^2} \\ -\frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial x}{\partial \eta} + \frac{\partial^2 x}{\partial \eta^2} & \frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial y}{\partial \eta} - \frac{\partial^2 y}{\partial \eta^2} \end{bmatrix} \tag{4-11}$$

$$[L'_{m2}] = \frac{1}{|J|}\begin{bmatrix} \frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial y}{\partial \xi} - \frac{\partial^2 y}{\partial \eta \partial \xi} & 0 \\ 0 & -\frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial x}{\partial \xi} + \frac{\partial^2 x}{\partial \eta \partial \xi} \\ -\frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial x}{\partial \xi} + \frac{\partial^2 x}{\partial \eta \partial \xi} & \frac{1}{|J|}\frac{\partial |J|}{\partial \eta}\frac{\partial y}{\partial \xi} - \frac{\partial^2 y}{\partial \eta \partial \xi} \end{bmatrix} \tag{4-12}$$

The much simpler expressions of $[L_{mi}]$ can be given if the two bridge sides are parallel but skewed:

$$[L_{m1}] = \begin{bmatrix} \frac{1}{a} & 0 \\ 0 & \frac{\kappa}{a} \\ \frac{\kappa}{a} & \frac{1}{a} \end{bmatrix} \tag{4-13}$$

$$[L_{m2}]=\begin{bmatrix} 0 & 0 \\ 0 & \frac{1}{b} \\ \frac{1}{b} & 0 \end{bmatrix} \tag{4-14}$$

Substituting Eq.(4-1) and Eq.(4-8) into Eq.(4-6), we obtain:

$$\{\varepsilon_m\}=[E_{m1}]\{d_m\}^e+[E_{m2}]\{d_m'\}^e \tag{4-15}$$

where

$$\begin{aligned} [E_{m1}]&=[L_{m1}][N'] \\ [E_{m2}]&=[L_{m2}][N] \end{aligned} \tag{4-16}$$

The membrane force vector $\{N_m\}$ can be obtained by means of Hooke's law.

$$\{N_m\}=[D_m]\{\varepsilon_m\} \tag{4-17}$$

in which

$$\{N_m\}=[N_x,\ N_y,\ N_{xy}]^T \tag{4-18}$$

and

$$[D_m]=\frac{Et}{1-\mu^2}\begin{bmatrix}1 & \mu & 0\\ \mu & 1 & 0\\ 0 & 0 & \frac{1-\mu}{2}\end{bmatrix} \tag{4-19}$$

The element potential energy of plane stress is given by:

$$\Pi^e=\frac{1}{2}\iint_{Ae}\{\varepsilon_m\}^T[D_m]\{\varepsilon_m\}dA-\iint_{Ae}\{u\}^T\{q_m\}dA \tag{4-20}$$

where the first term represents the strain potential energy and the second term is the potential energy of the external load $\{q_m\}$.

Substituting Eq.(4-15) into the above potential energy expression yields:

$$\begin{aligned}\Pi^e=\frac{1}{2}\iint_{Ae}(\{d_m\}^{eT}[E_{m1}]^T[D_m][E_{m1}]\{d_m\}^e+2\{d_m\}^{eT}[E_{m1}]^T[D_m][E_{m2}]\{d_m'\}^e\\ +\{d_m'\}^{eT}[E_{m2}]^T[D_m][E_{m2}]\{d_m'\}^e)dA-\iint_{Ae}\{d_m\}^{eT}[N]^T\{q_m\}dA\end{aligned} \tag{4-21}$$

Performing integration about ξ, the problem is reduced to that of the minimum of the simple integral :

$$\Pi^e=\frac{1}{2}\int_{-1}^{1}(\{d_m'\}^{eT}[A]^e\{d_m'\}^e+2\{d_m\}^{eT}[B]^e\{d_m'\}^e+\{d_m\}^{eT}[C]^e\{d_m\}^e)d\eta-\int_{-1}^{1}\{d_m\}^{eT}\{F\}^e d\eta \tag{4-22}$$

where

$$
\begin{aligned}
[A]^e &= h_j \int_0^1 [E_{m2}]^T [D_m][E_{m2}]|J| d\xi \\
[B]^e &= h_j \int_0^1 [E_{m1}]^T [D_m][E_{m2}]|J| d\xi \\
[C]^e &= h_j \int_0^1 [E_{m1}]^T [D_m][E_{m1}]|J| d\xi \\
\{F\}^e &= h_j \int_0^1 [N]^T \{q_m\}|J| d\xi
\end{aligned}
\tag{4-23}
$$

The total potential energy can be obtained by summing the corresponding potential energy of all element strips since the displacements between the adjacent element strips are compatible.

$$
\Pi = \sum_e \Pi^e = \frac{1}{2}\int_{-1}^{1} (\{d_m'\}^T[A]\{d_m'\} + 2\{d_m\}^T[B]\{d_m'\} + \{d_m\}^T[C]\{d_m\}) d\eta - \int_{-1}^{1} \{d_m\}^T\{F\} d\eta \tag{4-24}
$$

in which all matrices and vectors are obtained by assembling the corresponding matrices and vectors of the element strips.

Taking variation about the displacement function $\{d_m\}$ of the above expression, we obtain:

$$
\begin{aligned}
\delta\Pi(\{d_m\}) = \int_{-1}^{1} (\delta\{d_m'\}^T[A]\{d_m'\} + \delta\{d_m\}^T[B]\{d_m'\} + \delta\{d_m'\}[B]^T\{d_m\} \\
+ \delta\{d_m\}^T[C]\{d_m\}) d\eta - \int_{-1}^{1} \delta\{d_m\}^T\{F\} d\eta
\end{aligned}
\tag{4-25}
$$

Integrating by parts, yields:

$$\delta\Pi(\{d_m\})=\int_{-1}^{1}[-\delta\{d_m\}^T([A]\{d_m''\}+[A']\{d_m'\})+\delta\{d_m\}^T[B]\{d_m'\}-\delta\{d_m\}^T([B]^T\{d_m'\}+[B']^T\{d_m\})+\delta\{d_m\}^T[C]\{d_m\}]d\eta-\int_{-1}^{1}\delta\{d_m\}^T\{F\}d\eta \tag{4-26}$$

Or:

$$\delta\Pi(\{d_m\})=-\int_{-1}^{1}\delta\{d_m\}^T([A]\{d_m''\}+([A']+[B]^T-[B])\{d_m'\}+([B']^T-[C])\{d_m\}+\{F\})d\eta \tag{4-27}$$

Enforcing $\delta(\{d_m\})=0$ produces a set of linear ordinary differential equations of the form:

$$[A]\{d_m''\}+([A']+[B]^T-[B])\{d_m'\}+([B']^T-[C])\{d_m\}+\{F_m\}=0 \tag{4-28}$$

The governing ordinary differential equation (4-28) can be further simplified as:

$$[C_m]\{d_m''\}+[B_m]\{d_m'\}+[A_m]\{d_m\}+\{F_m\}=0 \tag{4-29}$$

by utilizing the relationship:

$$\begin{aligned}&[C_m]=[A]\\&[B_m]=[A']+[B]^T-[B]\\&[A_m]=[B']-[C]\\&\{F_m\}=\{F\}\end{aligned} \tag{4-30}$$

§4-3 The Bending Strain And Stress of the Element Strip

Since we only deal with right and skewed box girder bridges, all the derivations in Chapter 3 are applicable. Therefore, the governing ordinary differential equation for plate bending may be expressed as:

$$[E_b]\{d_b''''\}+[D_b]\{d_b'''\}+[C_b]\{d_b''\}+[B_b]\{d_b'\}+[A_b]\{d_b\}-\{F_b\}=0 \tag{4-31}$$

where matrices $[E_b]$, $[D_b]$, $[C_b]$, $[B_b]$ and $[A_b]$ are equivalent to [A], [B], [C], [D] and [E] in Chapter 3 respectively. And,

$$\begin{aligned}&\{d_b\}=[\{d_{b1}\},\{d_{b2}\},\ldots,\{d_{b(n+1)}\}]^T\\&\{d_{bi}\}=[w_i,\psi_i]^T \qquad i=1,2,\ldots,m+1\end{aligned} \tag{4-32}$$

§4-4 Application of the Kantorovich Finite Difference Method to Box Girder Bridges

In order to analyze box girder bridges, the total displacement vector {d} is introduced which includes both the membrane displacement and the deflection of the bridge deck, i.e.

$$\{d\}=[\{d_m\}\ \{d_b\}]^T \tag{4-33}$$

The governing ordinary differential equation can be expressed in terms of the total displacement vector { d } :

$$[\hat{E}]\{d''''\}+[\hat{D}]\{d'''\}+[\hat{C}]\{d''\}+[\hat{B}]\{d'\}+[\hat{A}]\{d\}+\{\hat{F}\}=0 \tag{4-34}$$

where

$$[\hat{A}]=\begin{bmatrix}[A_m] & 0\\ 0 & [A_b]\end{bmatrix}$$

$$[\hat{B}]=\begin{bmatrix}[B_m] & 0\\ 0 & [B_b]\end{bmatrix}$$

$$[\hat{C}]=\begin{bmatrix}[C_m] & 0\\ 0 & [C_b]\end{bmatrix} \qquad (4\text{-}35)$$

$$[\hat{D}]=\begin{bmatrix}0 & 0\\ 0 & [D_b]\end{bmatrix}$$

$$[\hat{E}]=\begin{bmatrix}0 & 0\\ 0 & [E_b]\end{bmatrix}$$

$$\{\hat{F}\}=\begin{Bmatrix}\{F_m\}\\ \{F_b\}\end{Bmatrix}$$

It should be pointed out that all the above derivation has been carried out in a local coordinate system, wherein the x and y axes coincide with the mid-surface of a plate. In box-girder bridges, any two plates may in general meet at any angle and a global coordinate system is obviously required in order to obtain the governing ordinary differential equation for the whole structure.

In Fig.4-2 the local coordinate of a coplanar plate is labeled as x'y'z' and the global coordinate as xyz. y and y' are coincident with each other and also with the intersection line of two adjoining plates of the box-girder bridge. The transformation of displacements between the two sets of coordinate systems is given by :

$$\{d'\}=[T]\{d\} \qquad (4\text{-}36)$$

in which [T] is the transformation matrix.

$$[T]=\begin{bmatrix} \cos\alpha & 0 & \sin\alpha & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\alpha & 0 & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (4\text{-}37)$$

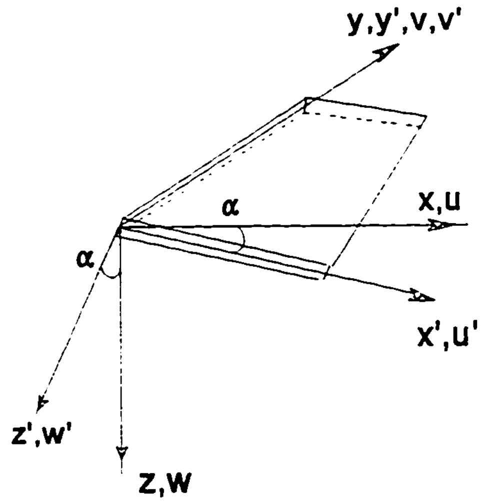


Fig. 4-2 Individual and global coordinate system

Finally, the finite difference method is employed to solve the ordinary differential equation for the whole box-girder bridge.

§4-5 Numerical Results

Since the bending part of the model presented here for box girder bridges has been tested in Chapter 3, a numerical example is now given to test the plane stress part before a real box girder bridge is analyzed.

1. The simply supported beam is under the uniformly distributed load q (Fig.4-3): The following data are used for the analysis: L=9, h=1.5, q=10, $E=2x10^6$, ν=0.167 and t=1. Due to the symmetry, only half beam is analyzed. The results of the displacement at point A and the flexural stress at point B are given in Table 5-1 along with the analytical solution given in Ref. [49] :

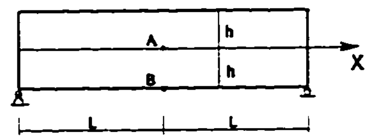


Fig. 4-3 Simply supported beam

$$v_A = \frac{5qL^4}{24EI}\left[1 + \frac{12h^2}{5L^2}\left(\frac{4}{5} + \frac{\nu}{2}\right)\right] \qquad (4\text{-}38)$$

Table 4-1 Displacement and Flexural Stress Results of the Simply Supported Beam

Strips*Sections	v_A	σ_B
3*3	3.1756	268.8
5*5	3.2082	269.8
Analytical Solution [49]	3.216	272.0

From the table, it can be seen that the method can have very high accuracy with only a few strips and sections used for structural discretization.

2. Two equal span continuous three cell box-girder bridge under eccentric concentrated loads is analyzed (Fig.4-3). By symmetry, only half the bridge (one span) is discretized by using a mesh of 26 strips and 10 sections. The results of transverse distributions of longitudinal stresses at loaded sections and intermediate supports are compared with those from Scordelis & Davis [54] in Fig.4-5 and Fig.4-6. Fairly good agreement is again found.

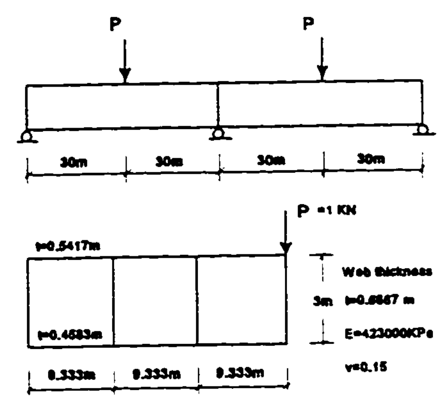


Fig. 4-4 Two equal span three cell box girder bridge

3. A simply supported six-cell box-girder bridge is under concentrated load at point A and point B as shown in Fig.4-7. Due to the symmetry, only half span is analyzed. Six section and 38 strip mesh is adopted for the analysis. The results of longitudinal membrane force and transverse moments at mid-span are depicted in Fig.4-8 and Fig.4-9 respectively together with those from the experiment [55]. Good agreement is observed.

§4-6 Summary

The proposed numerical approach shows considerable advantages over the finite element method in the analysis of box-girder bridges. The simplicity of the conventional Kantorovich method is preserved but its drawbacks in dealing with continuous structures or point supports are eliminated. It can reduce the analysis of box girder bridges to a simple task since needed input data for the analysis is minimized. The present method can provide a simple means for rapid and reliable analysis of box-girder bridges.

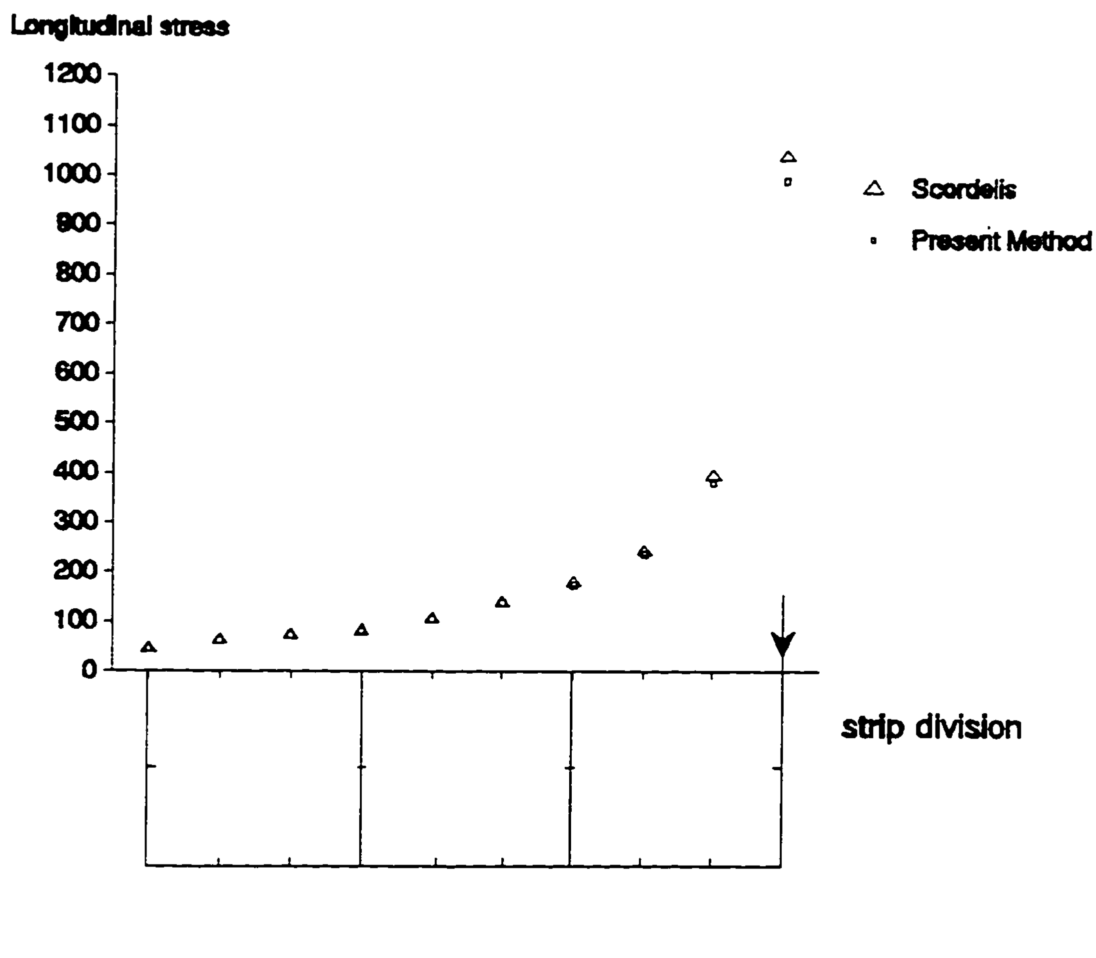


Fig.4-5 Distribution of longitudinal stress
at loaded section (top flange)

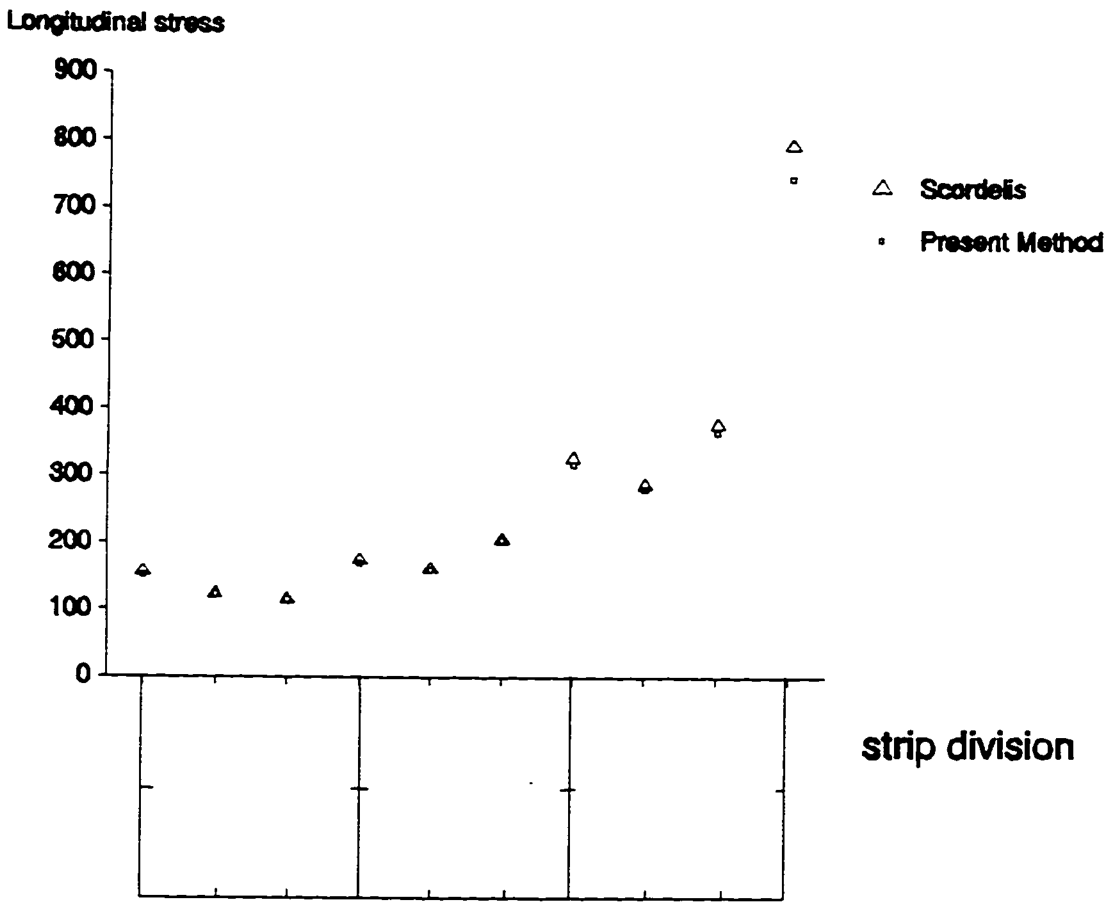
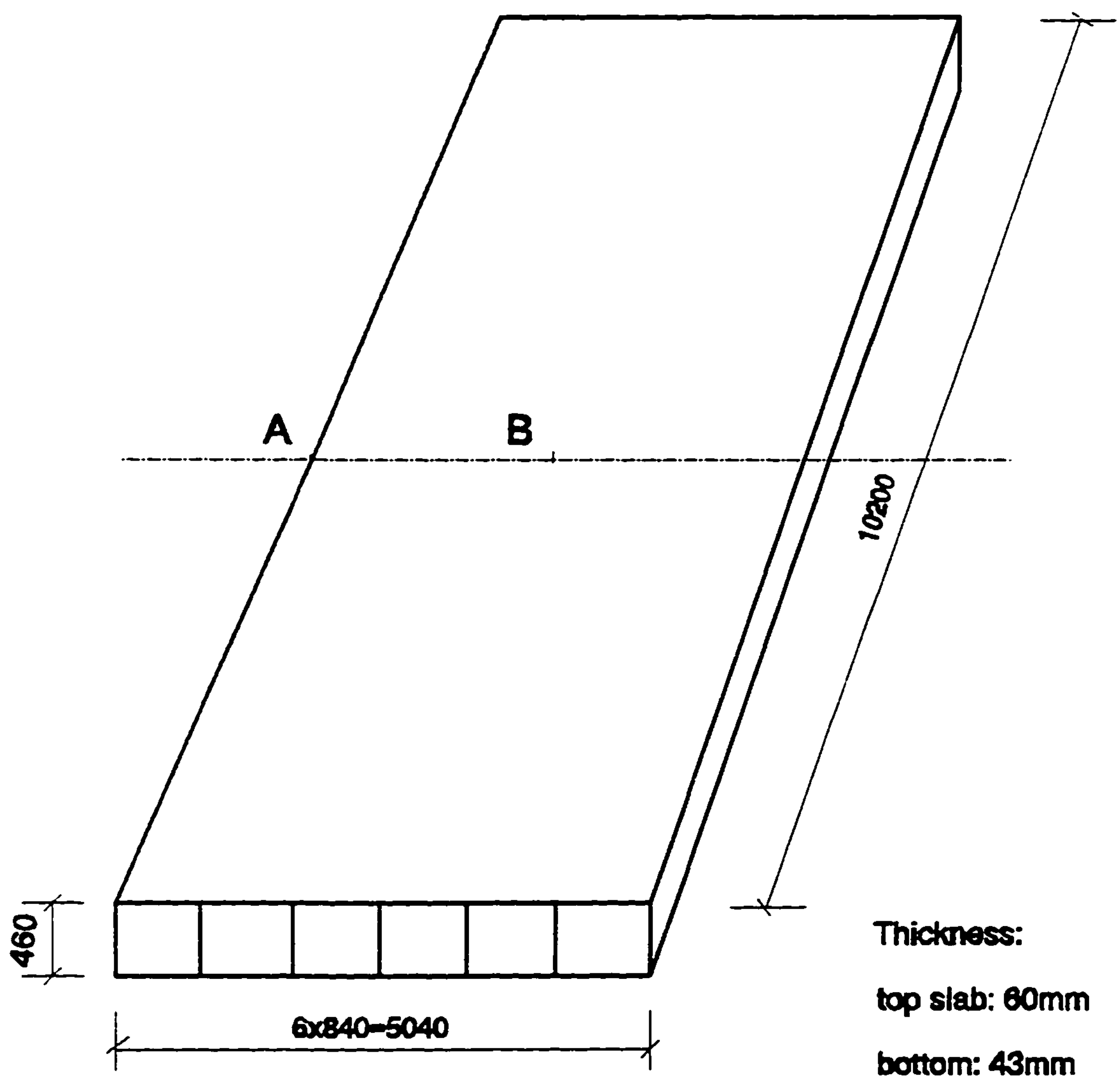


Fig.4-6 Distribution of longitudinal stress at intermediate support (top flange)



Thickness:

top slab: 60mm

bottom: 43mm

web: 77mm

Fig.4-7 Simply Supported six cell box girder bridge

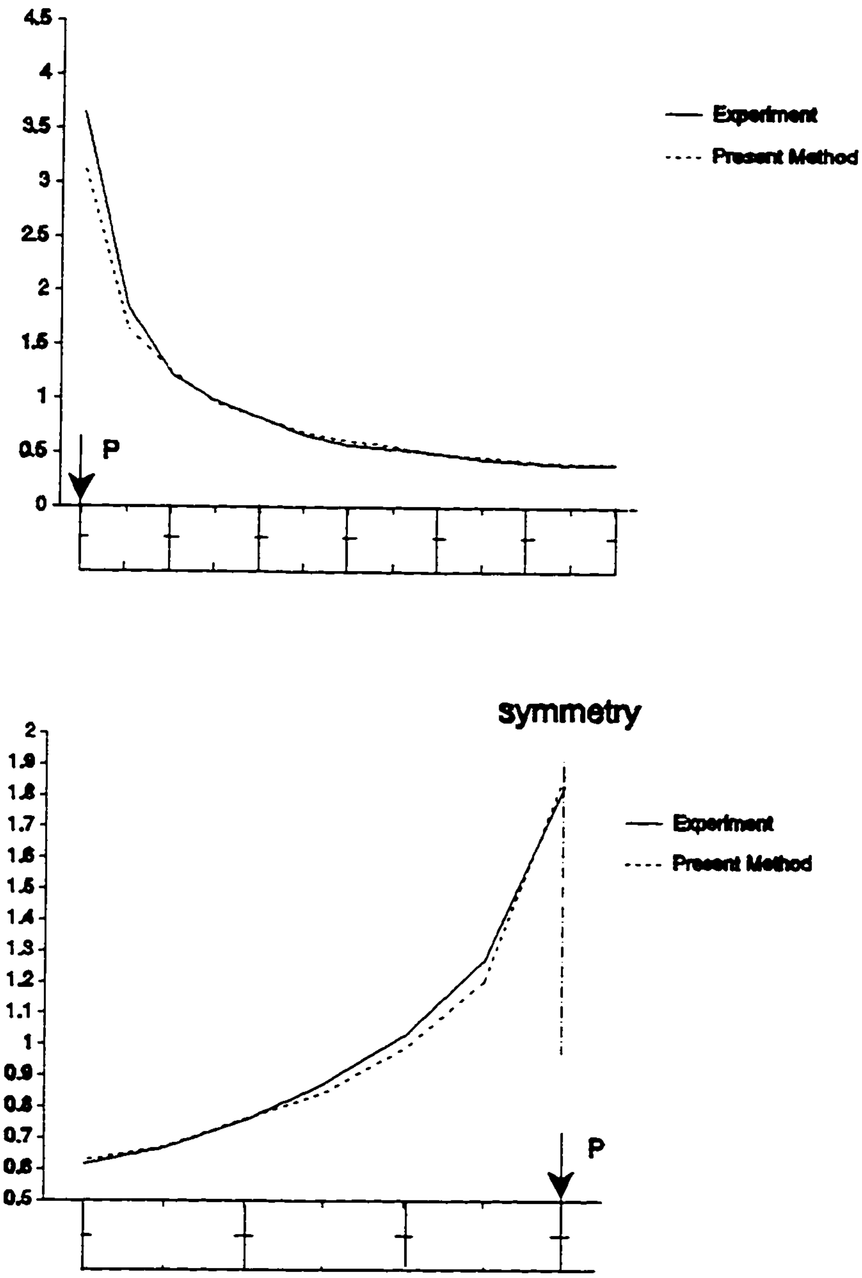


Fig.4-8 Distribution of longitudinal membrane force Ny at mid-span

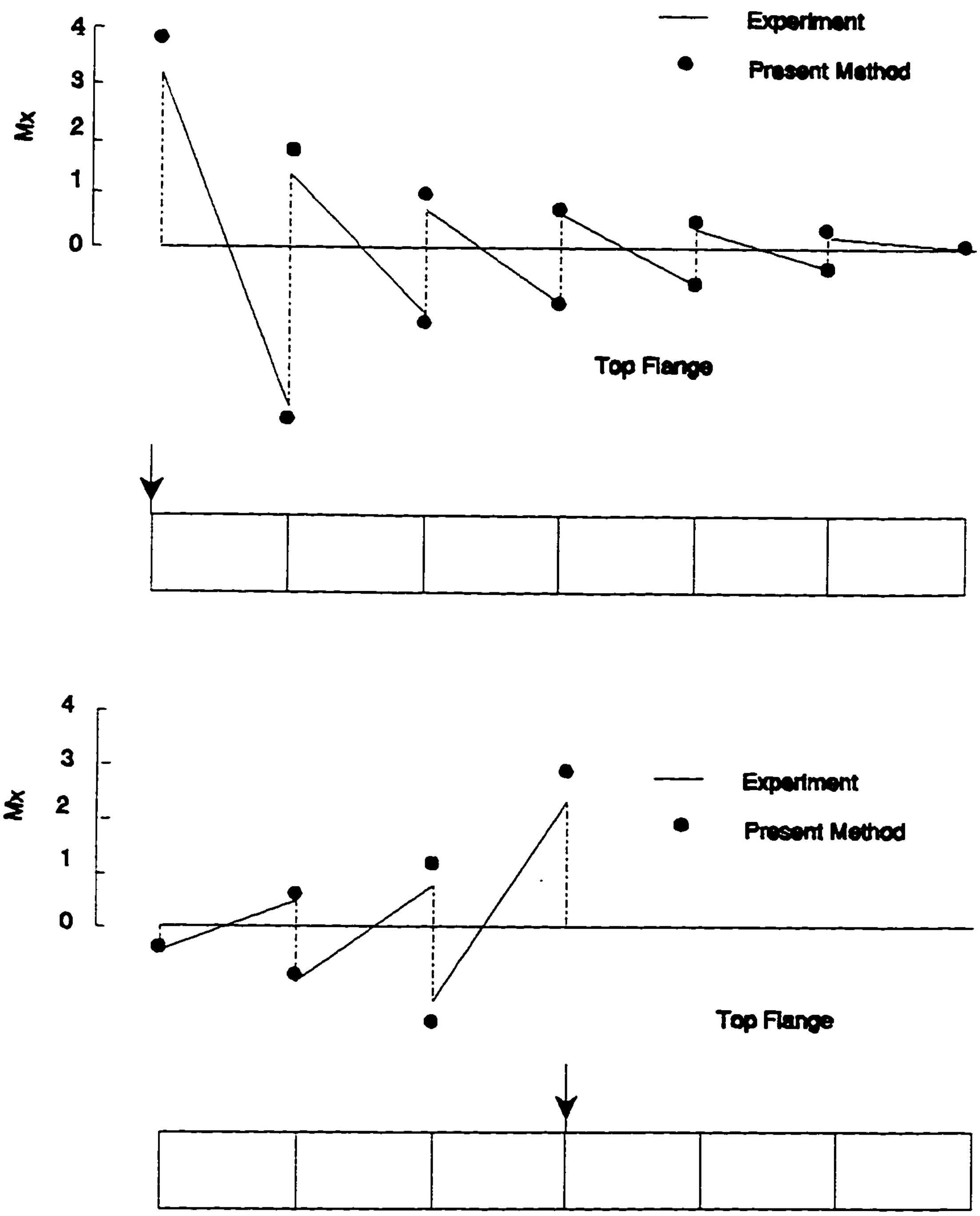


Fig.4-9 Distribution of transverse moment Mx at mid-span

Chapter 5

THE SPLINE KANTOROVICH METHOD AND ITS APPLICATION ON BRIDGE ENGINEERING

§5-1 General

To analyze simply supported right bridge decks of uniform sections (or a structure which may be realistically analyzed as such), the standard finite strip method or the Kantorovich method are the most efficient choices since the number of dimensions of the structure being analyzed is reduced by at least one. Further, as a result of this reduction in dimension, computer time, storage and input data requirements are reduced significantly. In spite of those advantages, the above mentioned semi-analytical numerical models have experienced difficulties when dealing with concentrated forces, multiple spans, discrete supports in bridge analysis. To overcome these difficulties and to retain the advantages of the finite strip method, a mathematical tool called 'spline function' was adopted to form the spline finite strip method by Y.K.Cheung et al in the 1980s. In the method, each nodal line is divided into a number of sections by evenly spaced knots, and each knot is taken as the center of a local spline. All the local splines on a nodal line form a series which are then utilized to simulate the longitudinal variation of displacements.

Obviously, a similar approach may be employed to improve the traditional Kantorovich method by introducing the spline function.

The spline Kantorovich method is developed and applied to the analysis of general bridge decks in this chapter. Firstly, the deck of the bridge is mapped into an unit square in the ξ-η plane. Secondly, the governing partial differential equation of the plate is reduced to the ordinary differential equation in the longitudinal direction of the bridge by the routine Kantorovich method. The spline function is then used as the displacement trial function. Finally the point allocation method, which requires the residuals to vanish at the chosen collection points, is utilized to solve the ordinary differential equation. The Mindlin plate theory is incorporated into the differential equation and, as a result, the effect of shear deformation of the plate is taken into account. Possible shear locking is avoided by the reduced integration technique. The application of the spline Kantorovich method to the bending problem of the slab bridges is presented and very similar accuracy is found when compared with the Kantorovich - finite difference method developed in the previous chapters.

§ 5-2 The Spline Function Interpolation

The semi-analytical finite strip method uses a function series, such as a beam eigen-function and orthogonal polynomial series, which can satisfy the boundary conditions a priori in the longitudinal direction. Such function series encounter some difficulties when dealing with plate bending problem involving abruptly changing properties, concentrated or patching loads and discrete supports since they are continuously differentiable. Such drawbacks can significantly limit its application in structural analysis. However, those problems can be completely overcome by introducing the mathematical tool of the spline function since the suitable spline function with the required continuity or discontinuity conditions can always be found. In addition, almost any prescribed external and internal boundary conditions could be met by a modified spline function. The success of the spline finite strip method has demonstrated that many difficulties encountered by the semi-analytical numerical methods in structural analysis may be solved by simply resorting to the use of the spline function as the displacement trial function.

As a tool for drawing a continuous smooth curves by a draughtsman, the spline became a mathematical tool attributing to the work of Schoenberg [56]. It has not attracted too much attention to engineering analysts until the advent of the digital computer. A variety of spline functions are available. The B_3 spline function, in which each local B_3 spline function has non-zero values over four consecutive subintervals, is chosen here to represent the displacement. It can ensure continuity up to the second derivative. Besides, the second derivative of B_3 spline function varies linearly and as a result, it can more easily simulate peak values of bending moment at the loaded section or at intermediate supports.

In order to use B_3 spline functions to interpolate an arbitrary function over the interval $-1\leq \eta \geq 1$, the interval is divided into a number of sections by evenly spaced knots η_i. With $\eta=\eta_i$, the B_3 spline function may be defined by:

$$\phi_i(\eta)=\frac{1}{6\lambda^3}\begin{cases} 0 & \eta\in[-1,\eta_{i-2}] \\ (\eta-\eta_{i-2})^3 & \eta\in[\eta_{i-2},\eta_{i-1}] \\ \lambda^3+3\lambda^2(\eta-\eta_{i-1})+3\lambda(\eta-\eta_{i-1})^2-3(\eta-\eta_{i-1})^3 & \eta\in[\eta_{i-1},\eta_i] \\ \lambda^3+3\lambda^2(\eta_{i+1}-\eta)+3\lambda(\eta_{i+1}-\eta)^2-3(\eta_{i+1}-\eta)^3 & \eta\in[\eta_i,\eta_{i+1}] \\ (\eta_{i+2}-\eta)^3 & \eta\in[\eta_{i+1},\eta_{i+2}] \\ 0 & \eta\in[\eta_{i+2},1] \end{cases} \tag{5-1}$$

where λ represents the equal section length between two knots. It should be pointed out that the length of different spline sections need not be equal in order to deal with some complicated problems of structural analysis.

The spline functions centered at all the knots can be adopted to interpolate the displacement function in the longitudinal direction.

$$d(\eta)=\sum_{i=-1}^{m+1} \delta_i \phi_i(\eta) \tag{5-2}$$

in which the δ_i are displacement coefficients to be determined. Within each section, the value of the interpolation spline function is related to four splines which are centered at the two end knots of the current section and the two knots adjacent to those knots. Therefore, two addition knots η_{-1} and η_{m+1} are needed in the first and the last sections and the m+1 interpolation requirements should be given in order to determine the m+1 displacement parameter δ_i :

$$\begin{aligned} &d|_{\eta=\eta_i}=d(\eta_i), \quad i=1,2,\ldots,m \\ &d'|_{\eta=\eta_0}=d'(\eta_0) \\ &d'|_{\eta=\eta_m}=d'(\eta_m) \end{aligned} \tag{5-3}$$

§5-3 The Spline Kantorovich Method

The whole domain in the ξ-η plane is partitioned into n strips along the ξ axis and m sections along the longitudinal direction of η. The B_3 spline function is adopted as the displacement trial function of the element strip, i.e.:

$$\{u\}=\begin{Bmatrix}\omega\\ \psi_x\\ \psi_y\end{Bmatrix}=[N_1(\xi)I_{3x3}\ ,\ N_2(\xi)I_{3x3}]\begin{Bmatrix}\{d_1(\eta)\}\\ \{d_2(\eta)\}\end{Bmatrix}=[N]\{d\}^e \tag{5-4}$$

where

$$\begin{aligned}N_1&=1-\xi\\ N_2&=\xi\end{aligned} \tag{5-5}$$

and

$$\{d\}^e=[\Phi(\eta)I_{6*6}]\begin{Bmatrix}w_j\\ \psi_{xj}\\ \psi_{yj}\\ w_{j+1}\\ \psi_{x(j+1)}\\ \psi_{y(j+1)}\end{Bmatrix} \tag{5-6}$$

$$[\Phi(\eta)]=[\phi_{-1},\phi_0,\phi_1,\ldots,\phi_m,\phi_{m+1}] \tag{5-7}$$

where w_j, ψ_{xj} , ψ_{yj} and w_{j+1}, ψ_{xj}, ψ_{yj+1} are the corresponding displacement parameters for nodal line j and j+1 respectively.

In dealing with general slab bridge decks, all the derivation in Chapter 2 up to the derivation of the following governing ordinary differential equation is applicable here.

$$[\bar{A}]\{d''\}+[\bar{B}]\{d'\}+[\bar{C}]\{d\}+\{F\}=0 \tag{5-8}$$

All the symbols in the above equation have the same meaning as those in Eq.(2-36) of Chapter 2.

It should be noted that the displacements between element strips are compatible. Therefore, the displacement vector $\{d(\eta)\}$ representing the whole structure as well as its derivatives can be easily obtained as follows:

$$\{d(\eta)\}=[\Phi(\eta)I_{3(n+1)x3(n+1)}]\{\delta\} \tag{5-9}$$

$$\{d'(\eta)\}=[\Phi'(\eta)I_{3(n+1)x3(n+1)}]\{\delta\} \tag{5-10}$$

$$\{d''(\eta)\}=[\Phi''(\eta)I_{3(n+1)*3(n+1)}]\{\delta\} \tag{5-11}$$

in which $\{\delta\}$ represents the displacement parameter vector for the whole structure.

Substituting the above three expressions into Eq.(5-8) yields:

$$([\bar{A}]\Phi''(\eta)+[\bar{B}]\Phi'(\eta)+[\bar{C}]\Phi(\eta))I_{3(n+1)x3(n+1)}\{\delta\}+\{F\}=0 \tag{5-12}$$

By noting :

$$[K]=([\bar{A}]\Phi''(\eta)+[\bar{B}]\Phi'(\eta)+[\bar{C}]\Phi(\eta))I_{3(n+1)x3(n+1)} \tag{5-13}$$

the above equation can be rewritten as :

$$[K]\{\delta\}+\{F\}=0 \tag{5-14}$$

The method of weighed residual has been recognized as an efficient numerical techniques in the last two decades due to its simplicity in data input and high accuracy. It provides analytical procedure for obtaining solutions in the form of functions which are close in some sense to the exact solution. Each choice of weighting function correspondings to a different criterion in approximate methods. A few well known weighted residual methods include the least square method, partition method, Galerkin method, moment method and collection method etc.. In this research, the simplest method of weighted residuals -- the point allocation method is employed to solve Eq. (5-11) though the other approaches are equally applicable. In implementation, all knots are chosen as allocation points on each nodal line.

The B_3 spline representations can easily be made to adopt to various prescribed boundary conditions. Due to the localization of the B_3 splines, only three boundary local splines have to be amended. The amended scheme is summarized in Table 5-1.

Table 5-1 Amendment scheme for boundary local splines

Boundary condition	Outer adjacent point	Point on the boundary	Inner adjacent point
Free end	ϕ_1	ϕ_2	ϕ_3
Simply supported	Eliminated	ϕ_0 - $4\phi_{-1}$	ϕ_1 - ϕ_{-1}
Clamped	Eliminated	Eliminated	ϕ_1 - $1/2\phi_0 + \phi_{-1}$

The penalty function method is adopted to impose various discrete boundary conditions. The main idea behind it is that a fictitious spring with a large stiffness coefficient

κ is introduced in the corresponding direction of imposed constraints. It should be noted that, in the method presented, the spring stiffness will affect the node in question and two adjacent nodes. For example, if the displacement w is prescribed to be zero at node i, then the stiffness matrix must be modified by the new stiffness:

$$[\bar{K}]=\kappa\begin{bmatrix} 2 & 4 & 1 \\ 4 & 16 & 4 \\ 1 & 4 & 1 \end{bmatrix} \tag{5-15}$$

Assembling the stiffness matrices of all point constraints with the structural stiffness matrix will accomplish the treatment of boundary conditions.

§ 5-4 Numerical Examples on General Bridge Decks

Two numerical examples which are given in Chapter 2 are tested here to examine the validity and accuracy of the spline Kantorovich method.

1. A four side simply supported square plate with span L is subjected to a uniform distributed load q (Fig.5-1). The thickness/span ratio is 0.3. The same number of strips and sections for discretion is adopted similar to example 1 in Chapter 2. In Table 5-2, the results for central deflection and central bending moments are given along with those calculated by the Kantorovich - finite difference method. It can be seen that in all cases the results converge to the analytical solution and it is well demonstrated here that both methods yield very high accuracy.

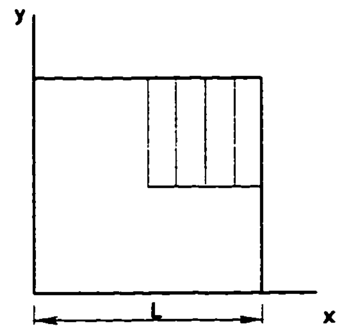


Fig.5-1 Simply supported square plate

Table 5-2 Simply Supported Square Plate

Strips*Sections	w_c		M_{xc}		M_{yc}	
	Method One #	Method Two *	Method One	Method Two	Method One	Method Two
2*2	0.5306	0.5298	3.839	3.829	4.062	4.0551
2*4	0.5564	0.5540	4.450	4.417	4.422	4.400
4*4	0.5766	0.5731	4.568	4.524	4.622	4.5868
4*6	0.5870	0.5825	4.680	4.611	4.699	4.6321
6*6	0.5899	0.5839	4.694	4.623	4.717	4.6565
Analytical Solution [48]	0.5957		4.79		4.79	
Multiplier	$0.01qL^4/D$		$0.01qL^2$			

The Kantorovich - Finite difference method

* The spline Kantorovich Method

2. A 60° fan-shaped bridge deck (Fig. 5-2) is subjected to a concentrated load. The internal and external radii are 7 inches and 13 inches respectively. The bridge is simply supported along its two ends. 12x6 mesh is used and the mid-span deflections results are tabulated in Table 5-3. Results obtained from the spline Kantorovich method and compare with those of the Kantorovich - finite difference method. As expected, close results are obtained.

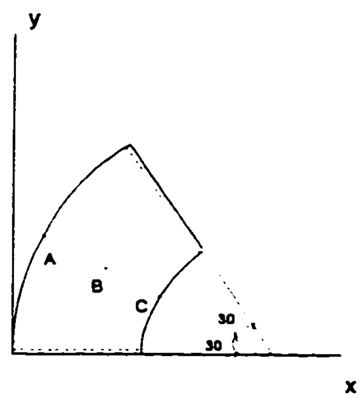


Fig. 5-2 Fan-shaped bridge deck

Table 5-3 Deflection across mid-span of fan-shaped plate

Loading Position	Radius (in.)	Present Method	Method One #	Sawko [52]
At Point A	7	0.01944	0.01934	0.0198
	9	0.03540	0.03523	0.0355
	11	0.05787	0.05748	0.0582
	13	0.08801	0.08754	0.0888
At Point B	7	0.01555	0.01551	0.0154
	9	0.02411	0.02401	0.0241
	11	0.03434	0.03416	0.0344
	13	0.04588	0.04572	0.0459
At Point C	7	0.01700	0.01684	0.0168
	9	0.01525	0.01503	0.0150
	11	0.01637	0.01610	0.0163
	13	0.01944	0.01908	0.0194

$E=4.6\times10^5$ lb/in^2, $\nu=0.35$, t=0.168in, p=1lb, # The Kantorovich - Finite difference method

§5-5 Summary

As a sister method of the Kantorovich - finite difference method, the spline Kantorovich method has been developed and applied to analyze general slab bridge decks. The method retains the advantages of the conventional Kantorovich method. In addition, as a result of the coordinate transformation technique, the method has been proven to be extremely versatile. The requirement on input data is minimal so that the method is very efficient in comparison with the conventional finite element method. When the method is augmented with the penalty function method, all types of bridge boundary conditions can be easily dealt with. Numerical results show that the present numerical model has similar accuracy with the Kantorovich - finite difference method developed in the previous chapters of this study. It can be concluded that the spline Kantorovich method provides a competitive alternative for engineering application of bridge deck analysis.

Chapter 6

CONCLUSIONS AND RECOMMENDATIONS

§6-1 Conclusions

In general, the governing equations of the bridge structures are a number of partial differential equations in which the unknowns are the internal forces and the deformations. The exact solution for those equations is very difficult even impossible to obtain and numerical computations must be resorted to. With the development of the high speed electronic digital computer, those numerical analyses have transformed from a somewhat esoteric science to a practical discipline in the last half century.

From the mathematical viewpoint, there are three general numerical techniques applicable directly to differential equations. They are the finite difference method, the weighted residual method and the variational method. The finite difference method can be considered as a direct discretion of differential equations. The weighted residual method is the approximate method which provides analytical procedure for obtaining solutions in the form of functions which are close to the exact solution in some sense. The variational method is based on the criterion of the calculus of variations in which the unknown functions are chosen to make the integral take a maximum or minimum value. It has been proved theoretically in reference [57] that the different finite element approaches could be unified by combining the variational method (or called energy method when used in structural analysis) and the weighted residual method. In terms of the mathematical foundation of the numerical analysis methods presented in this thesis, the Kantorovich - finite difference method can be considered as the outcome of the combined variational method and finite difference method, while the spline Kantorovich method is another example of applying the variational method in combination with the method of weighted residual.

If a structure has a dominant direction and a not too complicated plan configuration (In fact, a high proportion of bridges can be simplified to such a structure), the semi-analytical numerical methods would be the most efficient choices, in which bridges are treated as two dimensional structures with bias. Consequently, the computer time, storage as well as input data could be reduced significantly comparing with the finite element method. Kantorovich method is an ideal semi-analytical numerical approach in which the partial differential equation is reduced to the ordinary differential equation.

With the help of the conventional Kantorovich method, an efficient and reliable semi-analytical numerical approach is developed in the present study for the analysis of relatively regular shaped bridges with arbitrary loading and supporting conditions. The main achievements of this thesis are listed as follows:

(1) The Kantorovich - finite difference method has been successfully developed as a new semi-analytical numerical approach for bridge analysis. The main point is to reduce the governing partial differential equations into one dimensional ordinary differential equations by application of the Kantorovich method. With the help of the coordinate transformation, the presented method can be applied to irregular shaped bridge structures with various

boundary and loading conditions. The finite difference method is then adopted to solve the derived ordinary differential equation without difficulties.

(2) The presented method is examined by bending problem of general slab bridge decks. Mindlin thick plate theory is employed to take the shear deformation into account. In order to avoid shear locking, the reduced integration technique is used. To test the efficiency and effectiveness of the method, several numerical examples have been analyzed and compared with other available numerical methods. High accuracy is achieved even with fewer degrees of freedom for interpolation and less input data.

(3) The efficiency of analysis of skewed bridge is improved by adopting the Kantorovich - finite difference method. In this analysis, Kirchhoff thin plate theory is incorporated. The present method only require two parameters per node for interpolation instead of three for the finite element method. All comparisons of results indicate that the present finite difference modelling yields good agreement with other investigators. The validity of the Kantorovich - finite difference method is further proved.

(4) The Kantorovich - finite difference method has been extended to analyze the box-girder bridges. Firstly, the formula of the method for plane stress/strain problem is derived and tested. Secondly, by combining it with the bending part previously obtained, the Kantorovich - finite difference method for 'shell' structure is formulated. The numerical results obtained with minimum input data are in reasonable good agreement with those from the other analysis methods.

(5) By introducing the spline function, another achievement of this thesis is to develop another Kantorovich method based semi-analytical approach, that is, spline Kantorovich method. This new numerical model can also adapt to bridge structures with arbitrary

configurations, with discrete supports and concentrated or patch loads. Numerical results on general slab bridge decks show that this semi-analytical approach offers another very competitive alternative for bridge deck analysis.

It can be concluded that the Kantorovich-finite difference method proposed in this thesis can be a reliable and efficient numerical tool in modern bridge design with limit cost and analyst effort.

§6-2 Recommendations

Some recommendations for further study on this topic are suggested as follows:

(1) Extension of the Kantorovich finite difference method to include material and geometrical non-linear would be of interest. High order displacement functions may be employed in order to improve the accuracy for nonlinear analysis.

(2) The method needs to be developed for other kinds of bridges such as haunched bridges, cable-stayed bridges and composite material bridges etc..

(3) The spline Kantorovich method may be further studied and extended to analyze box-girder bridges and other kinds of bridges.

(4) Both Kantorovich - finite difference method and spline Kantorovich method may be extended to other types of structures, such as high-rise buildings and offshore structures, in which one dimension is dominant.

REFERENCES

1 Guyon, Y. Calcul Des Ponts Larges A Poutres Multiples Solidarisees Par Les Entretoises, Annales Des Ponts et Chaussees, pp683-718, 1946.

2 Cusens, A.R. and Pama R.P. Distribution Of Concentrated Load On Orthotropic Bridge Decks. The Structure Engineers, Vol.47, pp277-285, 1969.

3 Troitsky M.S. and Azad, A.K. Analysis Of Orthoropic Steel Bridge Decks By A Stiffness Method. Proc. Inst. Engrs., Part 2, Vol.55, pp447-462, 1973.

4 West, R., Recommendations On The Use of Grillage Analysis For Slab And Pseudo-slab Bridge Decks. Report 46.017, Cement And Concrete Association/CIRIA, London, 1973.

5 Cheung, M.S., Bakht, B. and Jaeger, L.G. Analysis Of Box-Girder Bridges By Grillage And Orthotropic Plate Method, Canadian Journal of Civil Engineering, December, 1982.

6 Zienkiewicz, O.C. The Finite Element Method, McGraw-Hill, 3rd edition, 1978.

7 Jategaonkar, R., Jaeger, L.G. and Cheung, M.S. Bridge Analysis Using Finite Elements, Canadian Society for Civil Engineering, 1985.

8 Strange, G.T. and Fix, G.J., An Analysis of the Finite Element Method, Prentice-Hall, 1973.

9 Cheung, M.S., Ng, S.F. and Zhao, J.Q. A Materially Nonlinear Finite Element Model for the Analysis of Curved Reinforced Concrete Box-Girder Bridges. Canadian Journal of Civil Engineering, Vol.20, No.5, pp754-759, 1993.

10 Long, Y.Q. and Zhao, J.Q., A Generalized Conforming Curved Element for Shallow Shells, Journal of Engineering Mechanics, Vol.9, No.1, pp3-10, 1992.

11 Long, Y.Q., and Zhao J.Q., The Combined Application of Energy Method and Weighted Residuals Method to Derive Finite Elements, China Aeronautical Journal, Vol.11, No.5, pp230-235, 1990.

12 Zhao, J.Q., The Convergence Study of Wachspress' Element QP6, Journal of Shanghai Mechanics, Vol.10, No.2, pp76-80, 1989.

13 Long Y.Q. and Zhao, J.Q., A New Triangular Generalized Element for Thin

Plates, International Journal of Communications in Applied Numerical Method, Vol.4, No.6, pp781-792, 1988.

14 Zhao, J.Q., Discussion on the Constructing Principle and Convergence of the Generalized Conforming Elements, China Civil Engineering Journal, Vol.21, No.4, pp35-37, 1988.

15 Long Y.Q. and Zhao J.Q., A Generalized Conforming Rectangular Element for Thick/Thin Plates, China Engineering Mechanics Journal, Vol.5, No.1, pp1-7, 1988.

16 Cheung, Y.K. The Finite Strip Method in the Analysis of Elastic Plates with Two Opposite Simply Supported Ends, Proc. Inst. Civil Engrs., Vol.40, pp1-7, 1968.

17 Cheung, Y.K. The Finite Strip Method Analysis of Elastic Plates, J. Engng Mech. Div., ASCE, Vol.94, pp1365-1378, 1968.

18 Cheung, Y.K. Analysis of Box Bridges by the Finite Strip Methods, SP 26 American Concrete Institute, Detroit, MI, pp357-378, 1969

19 Powell, G.H. and Ogden, D.W. Analysis of Orthotropic Bridge Decks, Am. Soc. Civ. Eng., Vol.95, ST5, pp903-923, 1969.

20 Cheung, Y.K. Analysis of Curvilinear Orthotropic Curved Bridge Decks, Publication International Association for Bridges and Structural Eng., 1969

21 Cheung, M.S., Cheung, Y.K. and Ghali, A. Analysis of Slab and Girder Bridges by the Finite Strip Method, Build. Sci, Vol.5, pp95-104, 1970

22 Loo, Y.C. and Cusens, A.R. Developments of the Finite Strip Method in the Analysis of Orthotropic Plates, Developments in bridge Design and Construction, 1971

23 Loo, Y.C. Analysis of Continuous Highway Box Bridges with Intermediate Stiffening, Eighth Aust. Road Res. Board Conf., pp13-20, 1976.

24 Delcourt, C. and Y.K. Finite Strip Method Analysis of Continuous Folded Plates, Proc. Int. Asso. of Brid. and Strut. Eng., pp1-16., May, 1978

25 Cheung, M.S. and Ghan, M.Y.T Analysis of Continuous Curved Box-Girder Bridges by the Finite Strip Method, ASCE Preprint 3515, Boston, 1979.

26 Cheung, Y.K. Finite Strip in Structural Analysis, Pergamon Press, 1976.

27 Loo, Y.C. and Cusens, A.R. The Finite Strip in Bridge Engineering, Viewpoint Publication, 1978.

28 Cheung, Y.K., Fan, S.C. and Wu, C.Q. Spline Finite Strip Method in Structural Analysis, Proc. International Conference on Finite Element Method, Shanghai, pp704-709, 1982.

29 Cook, R.D., Malkus, D.S. and Plesha, M.E., Concepts and Applications of

Finite Element Analysis, 3rd edition. John Wiley & Sons, New York, 1989.

30 Cheung, Y.K. and Fan S.C. Static Analysis of Right Box Girder Bridges by Spline Finite Strip Method, Proc. Instn. Civil Engrs, Part 2, Vol.75, pp311-323, 1983.

31 Tham, L.G., Li, W.Y., Cheung, Y.K. and Chen, M.J. Bending of Skewed Plates by Spline Finite Strip Method, Computers & Structures, Vol.22, pp31-38, 1988.

32 Cheung, Y.K., Tham, L.G. and Li, W.Y. Application of Spline Finite Strip Method in the Analysis of Curved Slab Bridge, Proc. Instn. Civil Engrs, Part2, Vol.81, pp111-124, 1986.

33 Li, W.Y., Cheung, Y.K. and Tham,L.G. Spline Finite Strip Analysis of General Plates, Journal of Engineering Mechanics, ASCE, Vol.112, pp43-54, 1986.

34 Cheung, M.S., Ng, S.F. and Zhao, J.Q. Analysis of Curved Reinforced Concrete Slab Bridges by the Spline Finite Strip Method, Canadian Journal of Civil Engineering, Vol.20, pp855-862, 1993

35 Liskovets, O.A. The Method of Lines (review), English translation appeared in Differential Equations, Vol.7, pp1308-1323, 1990.

36 Meyer, G.H. An Application of the Method of Lines to Multi Dimensional Free Boundary Problems, J. Inst. Maths Applics., No.20, pp317-329, 1977.

37 Meyer, G.H. The Method of Lines and Invariant Imbedding for Elliptic and Parabolic Free Boundary Problems, SIAM J. Numer. Anal, Vol.18., pp150-164, 1981.

38 Jones, D.J., South, J.C. and Klunker, E.B. On the Numerical Solution of Elliptic Partial Differential Equations by the Method of Lines, J. of Computational Physics, pp496-527, 1972.

39 Xanthis, L.S., A Psedo-ODE Modelling Trick for the Direct Method of Lines Computational of Important Fracture Mechanics Parameters, ACM SIGNUM Newsletter, Vol.21, 1986.

40 Yuan, S., MOL Analysis of Thin Plate Bending Problems, Computational Mechanics Communications, Vol.1, 1990.

41.Yuan, S. and Gao, J. A New Computational Tool in Structural Analysis: the Finite Element Methods of Lines, Proc. Int. Conf. on EPMESC, Macao, 1990.

42.Xanth, L.S. and Yuan, S. Finite Element Method of Lines for Elliptic Boundary Value Problems, To Appear in Comput. Meths. Appl. Mech. Engrg.

43.Yuan, S., Xanth, L.S. and Gao, J. Parametric Finite Element Method of Lines for Plane Elasticity Problems, To Appear in Comput. Meths. Appl. Mech. Engrg.

44 Kantorovich, N.B. The Approximate Methods of High Analysis, New York, 1958.

45. Zhao, J.Q., Cheung, M.S. and Ng, S.F. The Kantorovich-Finite Difference Method and Its Application on Bridge Engineering, The Second International Conference on Computational Structures Technology, Athens, Greece, August, 1994.

46 Ng, S.F., Cheung, M.S. and Zhao, J.Q. Analysis of Right and Skewed Box Girder Bridges by the Kantorovich - Finite Difference Method, International Conference on Computational Methods in Structural and Geotechnical Engineering, Hong Kong, December, 1994.

47 Zhao, J.Q., Ng, S.F. and Cheung M.S. The Spline Kantorovich Method and Its Application on Bridge Engineering, Proc. of the International Conference on Computer Aided Engineering and Design, Beijing, China, October, 1993, pp462-469.

48 Reismann, H. and Pawlik, P.S., Elasticity: Theory and Application, John Wiley and Sons, 1980.

49 Timoshenko, S.P. and Goodier, J.N., Theory of Elasticity, 3rd edition, McGraw Hill Book Co., New York, 1970.

50 Xu, C.D., The Method of Weighted Residual in Solid Mechanics, Publish House of Tongji University, in Chinese, 1986.

51 Coull, A. and Das, P.C., Analysis of Curved Bridge Decks, Proc. Instn Civ. Engrs, pp75-85, 1967.

52 Sawko, F. and Merriman, P.A., Finite Element Analysis of Bridge Curved in Plan, Developments in Bridge Design and Construction, Crosby Lockwood & Son Ltd., pp334-346, 1971.

53 Tham, L.G. and Cheung, Y.K., Approximate Analysis of Shear Wall Assemblies with Opening, J. Inst. Struct. Eng., Vol. 61B, pp41-45, 1983.

54 Scordelis, A.C., Analytical Solutions for Box Girder Bridges, In Development in Bridge Design and Construction, Crosby Lockwood, London, pp200-216, 1971.

55 Cope, R.J., Harris, G. and Sawko, F., A New Approach to the Analysis of Cellur Bridge Decks, Analysis of Structural Systems for Torsion: Proc. Symp. on the Annual Convention of ACI, Denver, London, pp185-210, 1971.

56 Schoenberg, I.J., Contributions to the Problem of Approximation of Equidistant Data by Analytic Functions, Q. Appl. Math., pp45-99, 1946.

57 Zhao, J.Q., The Generalized Confirming Elements in the Finite Element Method, Thesis for Doctor in Structural Engineering, Tsing Hua University, Beijing, China, 1989.

Appendix

FORTRAN SOURCE PROGRAMS

1. A brief introduction of the programs

The theoretical foundation of the programs has been described in Chapter two to Chapter five of the thesis. The programs have been tested in VM/SP CMS mainframe system at the University of Ottawa. Since there are no special elements which specify the computer hardware, these programs can be installed in any kind of computers even PCs. In the programs, the subroutines function individually to make the whole program be easily modified. Future work is needed to unify all the four programs with choices on the types of the bridge decks (general, skewed or box-girder), with plate theories employed (Mindlin or Kirchhoff) and with analysis methods (Kantorovich finite difference method or spline Kantorovich method).

2. Input description

All the needed input data are required by the main program. The physical meanings of the all variables in which the values should be provided as follows:

NSTRP	the number of equally divided strips in transverse direction
NSECT	the number of equally divided sections in longitudinal direction
NZC	the number of discrete supports
NPPOIN	The number of concentrated loads
ISUPT	boundary condition on the bridge abutment
NGUSS	number of Gussian integration points
NITEM	number of terms in displacement function along transverse direction
QLOAD	magnitude of the equally distributed load
COOXQ	array, the coordinates of the 8 control points
PPOIN	array, the magnitude and the position of the concentrated loads
IZC	array, positions of the discrete supports
EMODU	modulus of elasticity
POISS	Poisson's ratio
THICK	the thickness of the slab

3. An example of the input data file for four side simply supported slab (referring to example 1 in Chapter 2)

```
6,6,15,0,1,2,6
1.0
0.0,0.0,0.0,0.0,5.0,5.0,5.0,5.0
0.0,3.333,6.667,10.0,0.0,3.333,6.667,10.0
2,2,3,2,4,2,5,2,6,2,2,19,3,19,4,19,5,19,6,19,2,20,3,20,4,20,5,20,6,20
4044.44,0.3,1.0
```

4. Source programs

Kandif2

The FORTRAN Program for General Slab Bridge Decks by the Kantorovich Finite Difference Method Based on Mindlin Plate Theory

```fortran
C     ****************************************************************
C     *                                                              *
C     *                     THE MAIN PROGRAM                         *
C     *                                                              *
C     ****************************************************************
      DIMENSION TSTIF(600,600),PLOAD(600)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      COMMON/TDISP/PLOAD1(600)
      READ(5,*)NSTRP,NSECT,NZC,NPPOIN,ISUPT,NGUSS,NITEM
      NPOIN=(NSTRP+1)*(NSECT+1)
      WRITE(6,100)NSTRP,NSECT,NZC,NPPOIN,ISUPT
100   FORMAT(2X,'NSTRP=',I4,3X,'NSECT=',I4,
     &       3X,'NZC=',I6,2X,'NPPOIN=',I3,3X,'ISUPT=',I3/)
      READ(5,*)QLOAD
      WRITE(6,101)QLOAD
101   FORMAT(2X,'QLOAD=',E10.4/)
      READ(5,*)((COOXQ(I,J),J=1,8),I=1,2)
      WRITE(6,102)(COOXQ(1,J),J=1,8)
102   FORMAT(2X,'COORX=',8F8.2)
      WRITE(6,103)(COOXQ(2,J),J=1,8)
103   FORMAT(2X,'COORY=',8F8.2)
      IF(NPPOIN.GE.1) THEN
      READ(5,*)((PPOIN(I,J),J=1,2),I=1,NPPOIN)
      ELSE
      END IF
      IF(NZC.GE.1) THEN
      READ(5,*)((IZC(I,J),J=1,2),I=1,NZC)
      ELSE
      END IF
      READ(5,*)EMODU,POISS,THICK
      CALL MSTIF(TSTIF,PLOAD)
      CALL MATINV(TSTIF,3*NPOIN)
      DO 10 I=1,3*NPOIN
      PLOAD1(I)=0.0
      DO 10 K=1,3*NPOIN
10    PLOAD1(I)=PLOAD1(I)+TSTIF(I,K)*PLOAD(K)
      WRITE(6,104)
104   FORMAT(/2X,'DISPLACEMENTS'/)
      DO 20 I=1,3*NPOIN
      WRITE(6,105)I,PLOAD1(I)
105   FORMAT(2X,I3,8X,E15.4)
20    CONTINUE
      CALL MOMNT
      STOP
      END

      SUBROUTINE MSTIF(TSTIF,PLOAD)
C     ****************************************************************
C     *                                                              *
C     *    THE INTRODUCTION OF DISCRETE BOUNDARY CONDITIONS          *
C     *                                                              *
C     ****************************************************************
      DIMENSION TSTIF(600,600),PLOAD(600)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      CALL STIFT(TSTIF,PLOAD)
      IF(NZC.GE.1) THEN
      NPOIN=(NSTRP+1)*(NSECT+1)
```

```
      DO 20 I=1,NZC
      I1=IZC(I,1)
      I2=IZC(I,2)
      N1=(I1-1)*3*(NSTRP+1)+I2
      N2=(I2-1)*(NSECT+1)+I1
      DO 10 K=1,3*NPOIN
      TSTIF(N1,K)=0.0
      TSTIF(K,N2)=0.0
 10   CONTINUE
      TSTIF(N1,N2)=1.0
      PLOAD(N1)=0.0
 20   CONTINUE
      ELSE
      END IF
      RETURN
      END

      SUBROUTINE MOMNT
C     ***********************************************************
C     *                                                         *
C     *                    BENDING MOMENT                       *
C     *                                                         *
C     ***********************************************************
      DIMENSION BENMO(3,1),ELOAD(80),PELEM(80)
      DIMENSION SB1MA(3,12),SB2MA(3,12)
      DIMENSION SD1MA(3,12),SD2MA(3,12)
      DIMENSION P1(6,1),P2(6,1),P3(6,1)
      DIMENSION EED1(3,1),EED2(3,1),EED3(3,1),EED(3,1)
      DIMENSION DBMAT(3,3)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/TDISP/PLOAD1(600)
      CALL MATDB(DBMAT)
      H=2.0/NSECT
      DO 90 ISTRP=1,NSTRP
      WRITE(6,*)
      WRITE(6,101)ISTRP
 101  FORMAT(2X,'NO. OF STRIP=',I2)
      DO 20 I1=1,6*(NSECT+1)
 20   ELOAD(I1)=PLOAD1(3*(NSECT+1)*(ISTRP-1)+I1)
      DO 30 J=1,NSECT+1
      DO 30 K=1,6
      PELEM(K+6*(J-1))=ELOAD(J+(K-1)*(NSECT+1))
 30   CONTINUE
      DO 80 IPOIN=2,NSECT
      GY=-1.0+(IPOIN-1.0)*H
      WRITE(6,102)IPOIN
 102  FORMAT(2X,'NODE NO.=',I2)
      DO 40 K=1,6
      P1(K,1)=PELEM((IPOIN-2)*6+K)
      P2(K,1)=PELEM((IPOIN-1)*6+K)
 40   P3(K,1)=PELEM(IPOIN*6+K)
      DO 70 ILINE=1,2
      IF(ILINE.EQ.1) THEN
      GX=-1.0+2.0*(ISTRP-1.0)/NSTRP
      ELSE
      GX=-1.0+2.0*(ISTRP-1.0)/NSTRP+2.0/NSTRP
      END IF
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(EED1,SB2MA,P1,3,12,1,3,6,1)
```

```
      CALL MATRM(EED2,SB1MA,P2,3,12,1,3,6,1)
      CALL MATRM(EED3,SB2MA,P3,3,12,1,3,6,1)
      DO 60 II=1,3
 60   EED(II,1)=-EED1(II,1)/(2.0*H)+EED2(II,1)+EED3(II,1)/(2.0*H)
      CALL MATRM(BENMO,DBMAT,EED,3,3,1,3,3,1)
      WRITE(6,103)(BENMO(KK,1),KK=1,3)
 103  FORMAT(2X,F15.5,5X,F15.5,5X,F15.5)
 70   CONTINUE
 80   CONTINUE
 90   CONTINUE
      RETURN
      END

      SUBROUTINE STIFT(TSTIF,PLOAD)
C     ****************************************************************
C     *                                                              *
C     * THE TOTAL STIFFNESS MATRIX & LOAD VECTOR OF THE STRUCTURE    *
C     *                                                              *
C     ****************************************************************
      DIMENSION TSTIF(600,600),PLOAD(600),ESTIF(80,80),E13(80,80)
      DIMENSION PELEM(160),P2(80),PP(80,1),PP1(80),PP2(80,1)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      NPOIN=(NSTRP+1)*(NSECT+1)
      N2=NITEM-6
      DO 10 I=1,3*NPOIN
      DO 10 J=1,3*NPOIN
 10   TSTIF(I,J)=0.0
      DO 15 I=1,3*NPOIN
 15   PLOAD(I)=0.0
      IF(NPPOIN.GE.1) THEN
      DO 20 I=1,NPPOIN
      J=PPOIN(I,2)
 20   PLOAD(J)=PPOIN(I,1)
      ELSE
      END IF
      DO 60 ISTRP=1,NSTRP
      CALL STIFE(ESTIF,ISTRP,E13)
      DO 25 I1=1,NSECT+1
      DO 25 II=1,6
      IE=6*(I1-1)+II
      IT=3*(NSTRP+1)*(I1-1)+3*(ISTRP-1)+II
      DO 25 J1=1,2
      DO 25 JJ=1,3*(NSECT+1)
      JE=3*(NSECT+1)*(J1-1)+JJ
      JT=3*(NSECT+1)*(ISTRP+J1-2)+JJ
      TSTIF(IT,JT)=TSTIF(IT,JT)+ESTIF(IE,JE)
 25   CONTINUE
      IF(QLOAD.NE.0.0) THEN
      CALL LOADP(PELEM,ISTRP)
      DO 40 K=1,NSECT+1
      DO 30 I=1,6
 30   PP1(I+6*(K-1))=PELEM(I+NITEM*(K-1))
      DO 35 I=1,N2
 35   PP2(I+N2*(K-1),1)=PELEM(I+6+NITEM*(K-1))
 40   CONTINUE
      CALL MATRM(PP,E13,PP2,80,80,1,6*(NSECT+1),N2*(NSECT+1),1)
      DO 50 K=1,6*(NSECT+1)
 50   P2(K)=PP1(K)-PP(K,1)
      DO 55 I1=1,NSECT+1
```

```
      DO 55 II=1,6
      IE=6*(I1-1)+II
      IT=3*(NSTRP+1)*(I1-1)+3*(ISTRP-1)+II
      PLOAD(IT)=PLOAD(IT)+P2(IE)
55    CONTINUE
      ELSE
      END IF
60    CONTINUE
      RETURN
      END

      SUBROUTINE STIFE(ESTIF,ISTRP,E13)
C     ***************************************************************
C     *                                                             *
C     *          THE STIFFNESS MATRIX OF THE ITH STRIP              *
C     *                                                             *
C     ***************************************************************
      DIMENSION ESTIF1(160,160),ESTIF2(80,80),ESTIF(80,80)
      DIMENSION ESTI1(80,80),ESTI2(80,80),ESTI3(80,80),E13(80,80)
      DIMENSION AHATM(12,12),BHATM(12,12),CHATM(12,12),E132(80,80)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 10 I=1,NITEM*(NSECT+1)
      DO 10 J=1,NITEM*(NSECT+1)
10    ESTIF1(I,J)=0.0
      H=2.0/NSECT
      N1=NITEM/3
      N2=NITEM-6
      DO 30 IPOIN=1,NSECT+1
      GY=-1.0+(IPOIN-1.0)*H
      CALL MAABC(AHATM,BHATM,CHATM,ISTRP,GY,H)
      IF(IPOIN.EQ.1) THEN
      IF(ISUPT.EQ.1) THEN
      DO 15 II=1,N1
      DO 15 JJ=1,N1
      ESTIF1(3*II,3*JJ)=BHATM(3*II,3*JJ)
      ESTIF1(3*II,NITEM+3*JJ-2)=
     &      2.0*CHATM(3*II,3*JJ-2)
      ESTIF1(3*II,NITEM+3*JJ-1)=
     &      2.0*CHATM(3*II,3*JJ-1)
      ESTIF1(3*II,NITEM+3*JJ)=
     &      AHATM(3*II,3*JJ)+CHATM(3*II,3*JJ)
15    CONTINUE
      ELSE
      END IF
      ELSE IF(IPOIN.EQ.NSECT+1) THEN
      IF(ISUPT.EQ.1) THEN
      DO 20 II=1,N1
      DO 20 JJ=1,N1
      ESTIF1(3*II+NITEM*NSECT,NITEM*(NSECT-1)+3*JJ-2)=
     &      2.0*AHATM(3*II,3*JJ-2)
      ESTIF1(3*II+NITEM*NSECT,NITEM*(NSECT-1)+3*JJ-1)=
     &      2.0*AHATM(3*II,3*JJ-1)
      ESTIF1(3*II+NITEM*NSECT,NITEM*(NSECT-1)+3*JJ)=
     &      AHATM(3*II,3*JJ)+CHATM(3*II,3*JJ)
      ESTIF1(3*II+NITEM*NSECT,3*JJ+NITEM*NSECT)=BHATM(3*II,3*JJ)
20    CONTINUE
      ELSE
      END IF
      ELSE
```

```
      DO 25 II=1,NITEM
      DO 25 JJ=1,NITEM
      ESTIF1(II+NITEM*(IPOIN-1),JJ+NITEM*(IPOIN-2))=AHATM(II,JJ)
      ESTIF1(II+NITEM*(IPOIN-1),JJ+NITEM*(IPOIN-1))=BHATM(II,JJ)
      ESTIF1(II+NITEM*(IPOIN-1),JJ+NITEM*IPOIN)=CHATM(II,JJ)
25    CONTINUE
      END IF
30    CONTINUE
      DO 40 I=1,NITEM
      IF(ISUPT.EQ.1.AND.I.EQ.3) THEN
      ELSE IF(ISUPT.EQ.1.AND.I.EQ.6) THEN
      ELSE IF(ISUPT.EQ.1.AND.I.EQ.9) THEN
      ELSE IF(ISUPT.EQ.1.AND.I.EQ.12) THEN
      ELSE
      DO 35 K=1,NITEM*(NSECT+1)
      ESTIF1(I,K)=0.0
      ESTIF1(I+NITEM*NSECT,K)=0.0
      ESTIF1(K,I)=0.0
      ESTIF1(K,I+NITEM*NSECT)=0.0
35    CONTINUE
      ESTIF1(I,I)=1.0
      ESTIF1(I+NITEM*NSECT,I+NITEM*NSECT)=1.0
      END IF
40    CONTINUE
      DO 45 K1=1,NSECT+1
      DO 45 I=1,6
      DO 45 K2=1,NSECT+1
      DO 45 J=1,6
45    ESTIF(I+6*(K1-1),J+6*(K2-1))=ESTIF1(I+NITEM*(K1-1),
     &         J+NITEM*(K2-1))
      DO 50 K1=1,NSECT+1
      DO 50 I=1,6
      DO 50 K2=1,NSECT+1
      DO 50 J=1,N2
50    ESTI1(I+6*(K1-1),J+N2*(K2-1))=ESTIF1(I+NITEM*(K1-1),
     &         J+6+NITEM*(K2-1))
      DO 55 K1=1,NSECT+1
      DO 55 J=1,6
      DO 55 K2=1,NSECT+1
      DO 55 I=1,N2
55    ESTI2(I+N2*(K2-1),J+6*(K1-1))=ESTIF1(I+6+NITEM*(K2-1),
     &         J+NITEM*(K1-1))
      DO 60 K1=1,NSECT+1
      DO 60 I=1,N2
      DO 60 K2=1,NSECT+1
      DO 60 J=1,N2
60    ESTI3(I+N2*(K1-1),J+N2*(K2-1))=ESTIF1
     &         (I+6+NITEM*(K1-1),J+6+NITEM*(K2-1))
      CALL MATI3(ESTI3,N2*(NSECT+1))
      CALL MATRM(E13,ESTI1,ESTI3,80,80,80,
     &              6*(NSECT+1),N2*(NSECT+1),N2*(NSECT+1))
      CALL MATRM(E132,E13,ESTI2,80,80,80,6*(NSECT+1),
     &              N2*(NSECT+1),6*(NSECT+1))
      DO 70 I=1,6*(NSECT+1)
      DO 70 J=1,6*(NSECT+1)
      ESTIF2(I,J)=ESTIF(I,J)-E132(I,J)
70    CONTINUE
      DO 80 I=1,6*(NSECT+1)
      DO 80 K=1,6
      DO 80 J=1,NSECT+1
```

```
      ESTIF(I,J+(K-1)*(NSECT+1))=ESTIF2(I,K+6*(J-1))
80    CONTINUE
      RETURN
      END

      SUBROUTINE LOADP(PELEM,ISTRP)
C     ****************************************************************
C     *                                                              *
C     *        EQUIVELENT NODAL LOAD VECTOR OF THE ITH STRIP         *
C     *                                                              *
C     ****************************************************************
      DIMENSION P0(12),PELEM(160),PX(3),WEIGH(3)
      DIMENSION SHAPN(3,12),SHADN(3,12)
      DIMENSION XJACO(2,2),XTR2Y(2),XTRXY(2)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &              NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 10 I=1,NITEM*(NSECT+1)
10    PELEM(I)=0.0
      H=2.0/NSECT
      N1=NITEM/3
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 50 IPOIN=1,NSECT+1
      DO 20 I=1,NITEM
20    P0(I)=0.0
      GY=-1.0+(IPOIN-1.0)*H
      DO 30 IGUSS=1,3
      GX=PX(IGUSS)
      CALL SHAPE(SHAPN,SHADN,ISTRP,GX)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 25 K=1,N1
25    P0(3*K-2)=P0(3*K-2)+QLOAD*SHAPN(1,3*K-2)*DETJ*WEIGH(IGUSS)/NSTRP
30    CONTINUE
      DO 40 K=1,NITEM
40    PELEM(K+NITEM*(IPOIN-1))=PELEM(K+NITEM*(IPOIN-1))+P0(K)
50    CONTINUE
      DO 60 I=1,NITEM
      IF(ISUPT.EQ.1.AND.I.EQ.3) THEN
      ELSE IF(ISUPT.EQ.1.AND.I.EQ.6) THEN
      ELSE IF(ISUPT.EQ.1.AND.I.EQ.9) THEN
      ELSE IF(ISUPT.EQ.1.AND.I.EQ.12) THEN
      ELSE
      PELEM(I)=0.0
      PELEM(I+NITEM*NSECT)=0.0
      END IF
60    CONTINUE
      RETURN
      END

      SUBROUTINE MAABC(AHATM,BHATM,CHATM,ISTRP,GY,H)
C     ****************************************************************
C     *                                                              *
C     *                     TRANSITION MATRIX                        *
C     *                                                              *
C     ****************************************************************
      DIMENSION ABEMA(12,12),BBEMA(12,12),CBEMA(12,12)
      DIMENSION ASEMA(12,12),BSEMA(12,12),CSEMA(12,12)
      DIMENSION ABEPM(12,12),BBEPM(12,12)
```

```
      DIMENSION ASEPM(12,12),BSEPM(12,12)                               KAN03610
      DIMENSION AEMAT(12,12),BEMAT(12,12),CEMAT(12,12)                  KAN03620
      DIMENSION AEPMA(12,12),BEPMA(12,12)                               KAN03630
      DIMENSION ABARM(12,12),BBARM(12,12),CBARM(12,12)                  KAN03640
      DIMENSION AHATM(12,12),BHATM(12,12),CHATM(12,12)                  KAN03650
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,            KAN03660
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)              KAN03670
      CALL MAABE(ABEMA,ISTRP,GY)                                        KAN03680
      CALL MABBE(BBEMA,ISTRP,GY)                                        KAN03690
      CALL MACBE(CBEMA,ISTRP,GY)                                        KAN03700
      CALL MAASE(ASEMA,ISTRP,GY)                                        KAN03710
      CALL MABSE(BSEMA,ISTRP,GY)                                        KAN03720
      CALL MACSE(CSEMA,ISTRP,GY)                                        KAN03730
      CALL MABEP(ABEPM,ISTRP,GY)                                        KAN03740
      CALL MBBEP(BBEPM,ISTRP,GY)                                        KAN03750
      CALL MASEP(ASEPM,ISTRP,GY)                                        KAN03760
      CALL MBSEP(BSEPM,ISTRP,GY)                                        KAN03770
      DO 10 I=1,NITEM                                                   KAN03780
      DO 10 J=1,NITEM                                                   KAN03790
      AEMAT(I,J)=ABEMA(I,J)+ASEMA(I,J)                                  KAN03800
      BEMAT(I,J)=BBEMA(I,J)+BSEMA(I,J)                                  KAN03810
      AEPMA(I,J)=ABEPM(I,J)+ASEPM(I,J)                                  KAN03820
      BEPMA(I,J)=BBEPM(I,J)+BSEPM(I,J)                                  KAN03830
 10   CEMAT(I,J)=CBEMA(I,J)+CSEMA(I,J)                                  KAN03840
      DO 20 I=1,NITEM                                                   KAN03850
      DO 20 J=1,NITEM                                                   KAN03860
      ABARM(I,J)=AEMAT(I,J)                                             KAN03870
      BBARM(I,J)=AEPMA(I,J)+BEMAT(J,I)-BEMAT(I,J)                       KAN03880
 20   CBARM(I,J)=BEPMA(J,I)-CEMAT(I,J)                                  KAN03890
      DO 30 I=1,NITEM                                                   KAN03900
      DO 30 J=1,NITEM                                                   KAN03910
      AHATM(I,J)=ABARM(I,J)/(H*H)-BBARM(I,J)/(2.0*H)                    KAN03920
      BHATM(I,J)=-2.0*ABARM(I,J)/(H*H)+CBARM(I,J)                       KAN03930
 30   CHATM(I,J)=ABARM(I,J)/(H*H)+BBARM(I,J)/(2.0*H)                    KAN03940
      RETURN                                                            KAN03950
      END                                                               KAN03960
                                                                        KAN03970
      SUBROUTINE MAABE(ABEMA,ISTRP,GY)                                  KAN03980
C     ************************************************************      KAN03990
C     *                                                          *      KAN04000
C     *                  TRANSITION MATRIX                       *      KAN04010
C     *                                                          *      KAN04020
C     ************************************************************      KAN04030
      DIMENSION ABEMA(12,12),PX(3),DBMAT(3,3)                           KAN04040
      DIMENSION XJACO(2,2),AATR1(3,12),AATR2(12,12)                     KAN04050
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,12),SD2MA(3,12)               KAN04060
      DIMENSION SB1MA(3,12),SB2MA(3,12),WEIGH(3)                        KAN04070
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,            KAN04080
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)              KAN04090
      COMMON/PELEC/EMODU,POISS,THICK                                    KAN04100
      DATA WEIGH/0.555556,0.888889,0.555556/                            KAN04110
      DO 5 I=1,NITEM                                                    KAN04120
      DO 5 J=1,NITEM                                                    KAN04130
 5    ABEMA(I,J)=0.0                                                    KAN04140
      CALL MATDB(DBMAT)                                                 KAN04150
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP                    KAN04160
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP                    KAN04170
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP                    KAN04180
      DO 20 IGUSS=1,3                                                   KAN04190
      GX=PX(IGUSS)                                                      KAN04200
```

```
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)                   KAN04210
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,12,3,3,NITEM)                    KAN04220
      CALL MATRMT(AATR2,SB2MA,AATR1,3,12,12,3,NITEM,NITEM)              KAN04230
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)                     KAN04240
      DO 10 I=1,NITEM                                                   KAN04250
      DO 10 J=1,NITEM                                                   KAN04260
   10 ABEMA(I,J)=ABEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP          KAN04270
   20 CONTINUE                                                          KAN04280
      RETURN                                                            KAN04290
      END                                                               KAN04300
                                                                        KAN04310
      SUBROUTINE MABEP(ABEPM,ISTRP,GY)                                  KAN04320
C     ***************************************************************** KAN04330
C     *                                                               * KAN04340
C     *                     TRANSITION MATRIX                         * KAN04350
C     *                                                               * KAN04360
C     ***************************************************************** KAN04370
      DIMENSION ABEPM(12,12),PX(3),DBMAT(3,3)                           KAN04380
      DIMENSION XJACO(2,2),AATR1(3,12),AATR2(12,12),AATR3(12,12)        KAN04390
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,12),SD2MA(3,12)               KAN04400
      DIMENSION SB1MA(3,12),SB2MA(3,12),WEIGH(3)                        KAN04410
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,            KAN04420
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)              KAN04430
      COMMON/PELEC/EMODU,POISS,THICK                                    KAN04440
      DATA WEIGH/0.555556,0.888889,0.555556/                            KAN04450
      DO 5 I=1,NITEM                                                    KAN04460
      DO 5 J=1,NITEM                                                    KAN04470
    5 ABEPM(I,J)=0.0                                                    KAN04480
      CALL MATDB(DBMAT)                                                 KAN04490
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP                    KAN04500
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP                    KAN04510
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP                    KAN04520
      DO 20 IGUSS=1,3                                                   KAN04530
      GX=PX(IGUSS)                                                      KAN04540
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)                   KAN04550
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,12,3,3,NITEM)                    KAN04560
      CALL MATRMT(AATR2,SD2MA,AATR1,3,12,12,3,NITEM,NITEM)              KAN04570
      CALL MATRMT(AATR3,SB2MA,AATR1,3,12,12,3,NITEM,NITEM)              KAN04580
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)                     KAN04590
      DO 15 I=1,NITEM                                                   KAN04600
      DO 15 J=1,NITEM                                                   KAN04610
   15 ABEPM(I,J)=ABEPM(I,J)+(2.0*AATR2(I,J)*DETJ+AATR3(I,J)*DJDY)       KAN04620
     &         *WEIGH(IGUSS)/NSTRP                                      KAN04630
   20 CONTINUE                                                          KAN04640
      RETURN                                                            KAN04650
      END                                                               KAN04660
                                                                        KAN04670
      SUBROUTINE MABBE(BBEMA,ISTRP,GY)                                  KAN04680
C     ***************************************************************** KAN04690
C     *                                                               * KAN04700
C     *                     TRANSITION MATRIX                         * KAN04710
C     *                                                               * KAN04720
C     ***************************************************************** KAN04730
      DIMENSION BBEMA(12,12),PX(3),DBMAT(3,3)                           KAN04740
      DIMENSION XJACO(2,2),AATR1(3,12),AATR2(12,12)                     KAN04750
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,12),SD2MA(3,12)               KAN04760
      DIMENSION SB1MA(3,12),SB2MA(3,12),WEIGH(3)                        KAN04770
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,            KAN04780
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)              KAN04790
      COMMON/PELEC/EMODU,POISS,THICK                                    KAN04800
```

```
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
 5    BBEMA(I,J)=0.0
      CALL MATDB(DBMAT)
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 20 IGUSS=1,3
      GX=PX(IGUSS)
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,12,3,3,NITEM)
      CALL MATRMT(AATR2,SB1MA,AATR1,3,12,12,3,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
 10   BBEMA(I,J)=BBEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
 20   CONTINUE
      RETURN
      END

      SUBROUTINE MBBEP(BBEPM,ISTRP,GY)
C     ***********************************************************
C     *                                                         *
C     *                  TRANSITION MATRIX                      *
C     *                                                         *
C     ***********************************************************
      DIMENSION BBEPM(12,12),PX(3),DBMAT(3,3)
      DIMENSION XJACO(2,2),AATR1(3,12),AATR2(12,12)
      DIMENSION AATR3(3,12),AATR4(12,12),AATR5(12,12)
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,12),SD2MA(3,12)
      DIMENSION SB1MA(3,12),SB2MA(3,12),WEIGH(3)
      COMMON/PELEC/EMODU,POISS,THICK
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
 5    BBEPM(I,J)=0.0
      CALL MATDB(DBMAT)
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 20 IGUSS=1,3
      GX=PX(IGUSS)
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,12,3,3,NITEM)
      CALL MATRMT(AATR2,SD1MA,AATR1,3,12,12,3,NITEM,NITEM)
      CALL MATRM(AATR3,DBMAT,SD2MA,3,3,12,3,3,NITEM)
      CALL MATRMT(AATR4,SB1MA,AATR3,3,12,12,3,NITEM,NITEM)
      CALL MATRMT(AATR5,SB1MA,AATR1,3,12,12,3,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 15 I=1,NITEM
      DO 15 J=1,NITEM
 15   BBEPM(I,J)=BBEPM(I,J)+((AATR2(I,J)+AATR4(I,J))*DETJ+AATR5(I,J)
     &           *DJDY)*WEIGH(IGUSS)/NSTRP
 20   CONTINUE
      RETURN
      END
```

```
      SUBROUTINE MACBE(CBEMA,ISTRP,GY)
C     ***************************************************************
C     *                                                             *
C     *                     TRANSITION MATRIX                       *
C     *                                                             *
C     ***************************************************************
      DIMENSION CBEMA(12,12),DBMAT(3,3),PX(3)
      DIMENSION XJACO(2,2),AATR1(3,12),AATR2(12,12)
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,12),SD2MA(3,12)
      DIMENSION SB1MA(3,12),SB2MA(3,12),WEIGH(3)
      COMMON/PELEC/EMODU,POISS,THICK
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
5     CBEMA(I,J)=0.0
      CALL MATDB(DBMAT)
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 20 IGUSS=1,3
      GX=PX(IGUSS)
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DBMAT,SB1MA,3,3,12,3,3,NITEM)
      CALL MATRMT(AATR2,SB1MA,AATR1,3,12,12,3,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
10    CBEMA(I,J)=CBEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
20    CONTINUE
      RETURN
      END

      SUBROUTINE MAASE(ASEMA,ISTRP,GY)
C     ***************************************************************
C     *                                                             *
C     *                     TRANSITION MATRIX                       *
C     *                                                             *
C     ***************************************************************
      DIMENSION ASEMA(12,12),PX(3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      DIMENSION SS1MA(2,12),SS2MA(2,12),WEIGH(3)
      DIMENSION DSMAT(2,2)
      DIMENSION AATR1(2,12),AATR2(12,12),DS1MA(2,12),DS2MA(2,12)
      COMMON/PELEC/EMODU,POISS,THICK
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
5     ASEMA(I,J)=0.0
      CALL MATDS(DSMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 20 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DSMAT,SS2MA,2,2,12,2,2,NITEM)
      CALL MATRMT(AATR2,SS2MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
```

```
10    ASEMA(I,J)=ASEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
20    CONTINUE
      RETURN
      END

      SUBROUTINE MASEP(ASEPM,ISTRP,GY)
C     ****************************************************************
C     *                                                              *
C     *                    TRANSITION MATRIX                         *
C     *                                                              *
C     ****************************************************************
      DIMENSION ASEPM(12,12),PX(3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      DIMENSION SS1MA(2,12),SS2MA(2,12),WEIGH(3)
      DIMENSION DSMAT(2,2),AATR3(12,12)
      DIMENSION AATR1(2,12),AATR2(12,12),DS1MA(2,12),DS2MA(2,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
5     ASEPM(I,J)=0.0
      CALL MATDS(DSMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 20 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DSMAT,SS2MA,2,2,12,2,2,NITEM)
      CALL MATRMT(AATR2,DS2MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL MATRMT(AATR3,SS2MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 15 I=1,NITEM
      DO 15 J=1,NITEM
      ASEPM(I,J)=ASEPM(I,J)+(2.0*AATR2(I,J)*DETJ+AATR3(I,J)*DJDY)
     &           *WEIGH(IGUSS)/NSTRP
15    CONTINUE
20    CONTINUE
      RETURN
      END

      SUBROUTINE MABSE(BSEMA,ISTRP,GY)
C     ****************************************************************
C     *                                                              *
C     *                    TRANSITION MATRIX                         *
C     *                                                              *
C     ****************************************************************
      DIMENSION BSEMA(12,12),PX(3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      DIMENSION SS1MA(2,12),SS2MA(2,12),WEIGH(3)
      DIMENSION DSMAT(2,2)
      DIMENSION AATR1(2,12),AATR2(12,12),DS1MA(2,12),DS2MA(2,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
5     BSEMA(I,J)=0.0
      CALL MATDS(DSMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 20 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
```

```
      CALL MATRM(AATR1,DSMAT,SS2MA,2,2,12,2,2,NITEM)
      CALL MATRMT(AATR2,SS1MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
10    BSEMA(I,J)=BSEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
20    CONTINUE
      RETURN
      END

      SUBROUTINE MBSEP(BSEPM,ISTRP,GY)
C     ***********************************************************
C     *                                                         *
C     *                   TRANSITION MATRIX                     *
C     *                                                         *
C     ***********************************************************
      DIMENSION BSEPM(12,12),PX(3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      DIMENSION SS1MA(2,12),SS2MA(2,12),WEIGH(3)
      DIMENSION DSMAT(2,2),AATR3(2,12),AATR4(12,12),AATR5(12,12)
      DIMENSION AATR1(2,12),AATR2(12,12),DS1MA(2,12),DS2MA(2,12)
      COMMON/PELEC/EMODU,POISS,THICK
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
5     BSEPM(I,J)=0.0
      CALL MATDS(DSMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 20 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DSMAT,SS2MA,2,2,12,2,2,NITEM)
      CALL MATRMT(AATR2,DS1MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL MATRM(AATR3,DSMAT,DS2MA,2,2,12,2,2,NITEM)
      CALL MATRMT(AATR4,SS1MA,AATR3,2,12,12,2,NITEM,NITEM)
      CALL MATRMT(AATR5,SS1MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 15 I=1,NITEM
      DO 15 J=1,NITEM
      BSEPM(I,J)=BSEPM(I,J)+((AATR2(I,J)+AATR4(I,J))*DETJ
     &          +AATR5(I,J)*DJDY)*WEIGH(IGUSS)/NSTRP
15    CONTINUE
20    CONTINUE
      RETURN
      END

      SUBROUTINE MACSE(CSEMA,ISTRP,GY)
C     ***********************************************************
C     *                                                         *
C     *                   TRANSITION MATRIX                     *
C     *                                                         *
C     ***********************************************************
      DIMENSION CSEMA(12,12),PX(3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      DIMENSION SS1MA(2,12),SS2MA(2,12),WEIGH(3)
      DIMENSION DSMAT(2,2)
      DIMENSION AATR1(2,12),AATR2(12,12),DS1MA(2,12),DS2MA(2,12)
      COMMON/PELEC/EMODU,POISS,THICK
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 5 I=1,NITEM
```

```
      DO 5 J=1,NITEM
 5    CSEMA(I,J)=0.0
      CALL MATDS(DSMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 20 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DSMAT,SS1MA,2,2,12,2,2,NITEM)
      CALL MATRMT(AATR2,SS1MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
10    CSEMA(I,J)=CSEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
20    CONTINUE
      RETURN
      END

      SUBROUTINE MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
C     ***********************************************************
C     *                                                         *
C     *                    BENDING S MATRIX                     *
C     *                                                         *
C     ***********************************************************
      DIMENSION SB1MA(3,12),SD1MA(3,12),SD2MA(3,12)
      DIMENSION TRLB1(3,3),TRLB2(3,3),TLB1D(3,3),TLB2D(3,3)
      DIMENSION SHAPN(3,12),SHADN(3,12),SB2MA(3,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 5 I=1,3
      DO 5 J=1,NITEM
      SB1MA(I,J)=0.0
      SB2MA(I,J)=0.0
      SD1MA(I,J)=0.0
 5    SD2MA(I,J)=0.0
      CALL MATLB(TRLB1,TRLB2,GX,GY)
      CALL MALBD(TLB1D,TLB2D,GX,GY)
      CALL SHAPE(SHAPN,SHADN,ISTRP,GX)
      CALL MATRM(SB1MA,TRLB1,SHADN,3,3,12,3,3,NITEM)
      CALL MATRM(SB2MA,TRLB2,SHAPN,3,3,12,3,3,NITEM)
      CALL MATRM(SD1MA,TLB1D,SHADN,3,3,12,3,3,NITEM)
      CALL MATRM(SD2MA,TLB2D,SHAPN,3,3,12,3,3,NITEM)
      RETURN
      END

      SUBROUTINE MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
C     ***********************************************************
C     *                                                         *
C     *                     SHEAR S MATRIX                      *
C     *                                                         *
C     ***********************************************************
      DIMENSION SS1MA(2,12),SS2MA(2,12),DS1MA(2,12)
      DIMENSION TRLS0(2,3),TRLS1(2,3),TRLS2(2,3),TLS1D(2,3),TLS2D(2,3)
      DIMENSION SHAPN(3,12),SHADN(3,12),DS2MA(2,12)
      DIMENSION AATR1(2,12),AATR2(2,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 5 I=1,2
      DO 5 J=1,NITEM
      SS1MA(I,J)=0.0
      SS2MA(I,J)=0.0
```

```
      DS1MA(I,J)=0.0
 5    DS2MA(I,J)=0.0
      CALL MATLS(TRLS0,TRLS1,TRLS2,GX,GY)
      CALL MALSD(TLS1D,TLS2D,GX,GY)
      CALL SHAPE(SHAPN,SHADN,ISTRP,GX)
      CALL MATRM(AATR1,TRLS0,SHAPN,2,3,12,2,3,NITEM)
      CALL MATRM(AATR2,TRLS1,SHADN,2,3,12,2,3,NITEM)
      DO 10 I=1,2
      DO 10 J=1,NITEM
 10   SS1MA(I,J)=AATR1(I,J)+AATR2(I,J)
      CALL MATRM(SS2MA,TRLS2,SHAPN,2,3,12,2,3,NITEM)
      CALL MATRM(DS1MA,TLS1D,SHADN,2,3,12,2,3,NITEM)
      CALL MATRM(DS2MA,TLS2D,SHAPN,2,3,12,2,3,NITEM)
      RETURN
      END

      SUBROUTINE MATLB(TRLB1,TRLB2,GX,GY)
C     ****************************************************************
C     *                                                              *
C     *     OPERATOR TRANSFORMATION MATRIX FOR BENDING STRAIN        *
C     *                                                              *
C     ****************************************************************
      DIMENSION TRLB1(3,3),TRLB2(3,3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,3
      DO 10 J=1,3
      TRLB1(I,J)=0.0
 10   TRLB2(I,J)=0.0
      TRLB1(1,2)=-XJACO(2,2)/DETJ
      TRLB1(2,3)=XJACO(2,1)/DETJ
      TRLB1(3,2)=XJACO(2,1)/DETJ
      TRLB1(3,3)=-XJACO(2,2)/DETJ
      TRLB2(1,2)=XJACO(1,2)/DETJ
      TRLB2(2,3)=-XJACO(1,1)/DETJ
      TRLB2(3,2)=-XJACO(1,1)/DETJ
      TRLB2(3,3)=XJACO(1,2)/DETJ
      RETURN
      END

      SUBROUTINE MALBD(TLB1D,TLB2D,GX,GY)
C     ****************************************************************
C     *                                                              *
C     *     OPERATOR TRANSFORMATION MATRIX FOR BENDING STRAIN        *
C     *                                                              *
C     ****************************************************************
      DIMENSION TLB1D(3,3),TLB2D(3,3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,3
      DO 10 J=1,3
      TLB1D(I,J)=0.0
 10   TLB2D(I,J)=0.0
      TLB1D(1,2)=-XTR2Y(2)/DETJ+DJDY*XJACO(2,2)/(DETJ**2)
      TLB1D(2,3)=XTR2Y(1)/DETJ-DJDY*XJACO(2,1)/(DETJ**2)
      TLB1D(3,2)=TLB1D(2,3)
      TLB1D(3,3)=TLB1D(1,2)
      TLB2D(1,2)=XTRXY(2)/DETJ-DJDY*XJACO(1,2)/(DETJ**2)
      TLB2D(2,3)=-XTRXY(1)/DETJ+DJDY*XJACO(1,1)/(DETJ**2)
      TLB2D(3,2)=TLB2D(2,3)
      TLB2D(3,3)=TLB2D(1,2)
      RETURN
```

```
      END

      SUBROUTINE MATLS(TRLS0,TRLS1,TRLS2,GX,GY)
C     ****************************************************************
C     *                                                              *
C     *      OPERATOR TRANSFORMATION MATRIX FOR SHEAR STRAIN         *
C     *                                                              *
C     ****************************************************************
      DIMENSION TRLS0(2,3),TRLS1(2,3),TRLS2(2,3)
      DIMENSION XJACO(2,2),XTRXY(2),XTR2Y(2)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,2
      DO 10 J=1,3
      TRLS0(I,J)=0.0
      TRLS1(I,J)=0.0
10    TRLS2(I,J)=0.0
      TRLS0(1,2)=-1.0
      TRLS0(2,3)=-1.0
      TRLS1(1,1)=XJACO(2,2)/DETJ
      TRLS1(2,1)=-XJACO(2,1)/DETJ
      TRLS2(1,1)=-XJACO(1,2)/DETJ
      TRLS2(2,1)=XJACO(1,1)/DETJ
      RETURN
      END

      SUBROUTINE MALSD(TLS1D,TLS2D,GX,GY)
C     ****************************************************************
C     *                                                              *
C     *      OPERATOR TRANSFORMATION MATRIX FOR SHEAR STRAIN         *
C     *                                                              *
C     ****************************************************************
      DIMENSION TLS1D(2,3),TLS2D(2,3)
      DIMENSION XJACO(2,2),XTRXY(2),XTR2Y(2)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,2
      DO 10 J=1,3
      TLS1D(I,J)=0.0
10    TLS2D(I,J)=0.0
      TLS1D(1,1)=XTR2Y(2)/DETJ-DJDY*XJACO(2,2)/(DETJ**2)
      TLS1D(2,1)=-XTR2Y(1)/DETJ+DJDY*XJACO(2,1)/(DETJ**2)
      TLS2D(1,1)=-XTRXY(2)/DETJ+DJDY*XJACO(1,2)/(DETJ**2)
      TLS2D(2,1)=XTRXY(1)/DETJ-DJDY*XJACO(1,1)/(DETJ**2)
      RETURN
      END

      SUBROUTINE SHAPE(SHAPN,SHADN,ISTRP,GX)
C     ****************************************************************
C     *                                                              *
C     *          THE SHAPE FUNCTION FOR THE X-DIRECTION              *
C     *                                                              *
C     ****************************************************************
      DIMENSION SHA(4),SHA1(4),SHAPN(3,12),SHADN(3,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      N1=NITEM/3
      DO 10 I=1,3
      DO 10 J=1,NITEM
      SHAPN(I,J)=0.0
10    SHADN(I,J)=0.0
      GXX=1.0-ISTRP+(GX+1.0)*NSTRP/2.0
```

```
      IF(N1.EQ.2) THEN
      SHA(1)=1.0-GXX
      SHA(2)=GXX
      SHA1(1)=-NSTRP/2.0
      SHA1(2)=NSTRP/2.0
      ELSE IF(N1.EQ.3) THEN
      SHA(1)=1.0-3.0*GXX+2.0*GXX**2
      SHA(2)=-GXX+2.0*GXX**2
      SHA(3)=4.0*GXX-4.0*GXX**2
      SHA1(1)=(-1.5+2.0*GXX)*NSTRP
      SHA1(2)=(-0.5+2.0*GXX)*NSTRP
      SHA1(3)=(2.0-4.0*GXX)*NSTRP
      ELSE IF(N1.EQ.4) THEN
      SHA(1)=1.0-5.5*GXX+9.0*GXX**2-4.5*GXX**3
      SHA(2)=GXX-4.5*GXX**2+4.5*GXX**3
      SHA(3)=9.0*GXX-22.5*GXX**2+13.5*GXX**3
      SHA(4)=-4.5*GXX+18.0*GXX**2-13.5*GXX**3
      SHA1(1)=(-2.75+9.0*GXX-6.75*GXX**2)*NSTRP
      SHA1(2)=(0.5-4.5*GXX+6.75*GXX**2)*NSTRP
      SHA1(3)=(4.5-22.5*GXX+20.25*GXX**2)*NSTRP
      SHA1(4)=(-2.25+18.0*GXX-20.25*GXX**2)*NSTRP
      ELSE
      END IF
      DO 20 I=1,N1
      SHAPN(1,3*I-2)=SHA(I)
      SHAPN(2,3*I-1)=SHA(I)
      SHAPN(3,3*I)=SHA(I)
      SHADN(1,3*I-2)=SHA1(I)
      SHADN(2,3*I-1)=SHA1(I)
      SHADN(3,3*I)=SHA1(I)
20    CONTINUE
      RETURN
      END

      SUBROUTINE TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      ***********************************************************
      *                                                         *
      *         THE COORDINATE TRANSFORMATION SUBROUTINE        *
      *                                                         *
      ***********************************************************
      DIMENSION XJACO(2,2),SHAP(2,8),PX(8),PY(8)
      DIMENSION TRXY(8),TR2Y(8),XTRXY(2),XTR2Y(2)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DATA PX/-1.0,-1.0,-1.0,-1.0,1.0,1.0,1.0,1.0/
      DATA PY/-1.0,-0.33333,0.33333,1.0,
     &        -1.0,-0.33333,0.33333,1.0/
      DO 20 I=1,8
      IF(I.EQ.1.OR.I.EQ.4.OR.I.EQ.5.OR.I.EQ.8) THEN
      SHAP(1,I)=PX(I)*(9.0*GY*GY-1.0)*(1.0+PY(I)*GY)/32.0
      SHAP(2,I)=(1.0+PX(I)*GX)*(18.0*GY+27.0*PY(I)*GY*GY-PY(I))/32.0
      TR2Y(I)=9.0*(1.0+GX*PX(I))*(1.0+3.0*GY*PY(I))/16.0
      TRXY(I)=PX(I)*(18.0*GY+27.0*GY*GY*PY(I)-PY(I))/32.0
      ELSE
      SHAP(1,I)=9.0*PX(I)*(1.0-GY*GY)*(1.0+9.0*PY(I)*GY)/32.0
      SHAP(2,I)=9.0*(1.0+PX(I)*GX)*(-2.0*GY-27.0*PY(I)*GY*GY
     &          +9.0*PY(I))/32.0
      TR2Y(I)=9.0*(1.0+GX*PX(I))*(-1.0-27.0*GY*PY(I))/16.0
      TRXY(I)=9.0*PX(I)*(-2.0*GY-27.0*GY*GY*PY(I)+9.0*PY(I))/32.0
      END IF
```

```
 20     CONTINUE
        DO 40 I=1,2
        DO 40 J=1,2
        XJACO(I,J)=0.0
        DO 30 K=1,8
 30     XJACO(I,J)=XJACO(I,J)+SHAP(I,K)*COOXQ(J,K)
 40     CONTINUE
        DETJ=XJACO(1,1)*XJACO(2,2)-XJACO(1,2)*XJACO(2,1)
        DO 55 I=1,2
        XTR2Y(I)=0.0
        XTRXY(I)=0.0
        DO 50 K=1,8
        XTR2Y(I)=XTR2Y(I)+TR2Y(K)*COOXQ(I,K)
 50     XTRXY(I)=XTRXY(I)+TRXY(K)*COOXQ(I,K)
 55     CONTINUE
        DJDY=XJACO(1,1)*XTR2Y(2)+XJACO(2,2)*XTRXY(1)
     &      -XJACO(2,1)*XTRXY(2)-XJACO(1,2)*XTR2Y(1)
        RETURN
        END

        SUBROUTINE MATDS(DSMAT)
C       ***********************************************************
C       *                                                         *
C       *      THE MATERIAL CONSTITUTIONAL MATRIX FOR SHEAR       *
C       *                                                         *
C       ***********************************************************
        DIMENSION DSMAT(2,2)
        COMMON/PELEC/EMODU,POISS,THICK
        DO 10 I=1,2
        DO 10 J=1,2
 10     DSMAT(I,J)=0.0
        GMODU=EMODU/(2.0*(1.0+POISS))
        DSMAT(1,1)=GMODU*THICK/1.2
        DSMAT(2,2)=GMODU*THICK/1.2
        RETURN
        END

        SUBROUTINE MATDB(DBMAT)
C       ***********************************************************
C       *                                                         *
C       *      THE CONSTITUTION RELATIONSHIP FOR BENDING          *
C       *                                                         *
C       ***********************************************************
        DIMENSION DBMAT(3,3)
        COMMON/PELEC/EMODU,POISS,THICK
        DO 10 I=1,3
        DO 10 J=1,3
 10     DBMAT(I,J)=0.0
        RIGID=EMODU*THICK**3/(12.0*(1.0-POISS**2))
        DBMAT(1,1)=RIGID
        DBMAT(1,2)=POISS*RIGID
        DBMAT(2,1)=POISS*RIGID
        DBMAT(2,2)=RIGID
        DBMAT(3,3)=(1.0-POISS)*RIGID/2.0
        RETURN
        END

        SUBROUTINE GAUSS(PX,WEIGH,ISTRP)
C       ***********************************************************
C       *                                                         *
```

```
C     *     GAUSSIAN POINT COORDINATES AND WEIGHTED COFFICIENTS     *
C     *                                                             *
C     ***************************************************************
      DIMENSION PX(3),WEIGH(3)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      IF(NGUSS.EQ.3) THEN
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      WEIGH(1)=0.555556
      WEIGH(2)=0.888889
      WEIGH(3)=0.555556
      ELSE IF(NGUSS.EQ.2) THEN
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.42265/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.57735/NSTRP
      WEIGH(1)=1.0
      WEIGH(2)=1.0
      ELSE
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      WEIGH(1)=2.0
      END IF
      RETURN
      END

      SUBROUTINE MATRM(DB,D,B,L,M,N,LR,MR,NR)
C     ***************************************************************
C     *                                                             *
C     *     MATRIX MULTIPLICATION PROGRAM DB(M,N)=D(M,L)*B(L,N)     *
C     *                                                             *
C     ***************************************************************
      DIMENSION D(L,M),B(M,N),DB(L,N)
      DO 10 I=1,LR
      DO 10 J=1,NR
      DB(I,J)=0.0
      DO 10 K=1,MR
   10 DB(I,J)=DB(I,J)+D(I,K)*B(K,J)
      RETURN
      END

      SUBROUTINE MATRMT(DBT,D,B,L,M,N,LR,MR,NR)
C     ***************************************************************
C     *                                                             *
C     *        MATRIX MULTIPLICATION AFTER THE FIRST ONE            *
C     *          IS TRANSPOSED   DB(M,N)=D(L,M)*B(L,N)              *
C     *                                                             *
C     ***************************************************************
      DIMENSION D(L,M),B(L,N),DBT(M,N)
      DO 10 I=1,MR
      DO 10 J=1,NR
      DBT(I,J)=0.0
      DO 10 K=1,LR
   10 DBT(I,J)=DBT(I,J)+D(K,I)*B(K,J)
      RETURN
      END

      SUBROUTINE MATINV(A,N)
C     ***************************************************************
C     *                                                             *
C     *                 THE PROGRAM OF INVERSION                    *
```

```
C     *                                                                *
C     ******************************************************************
      DIMENSION INDEX(600,2),A(600,600)
      DO 10 I=1,N
 10   INDEX(I,1)=0
      II=0
 20   AMAX=-1.0
      DO 70 I=1,N
      IF(INDEX(I,1))70,40,70
 40   DO 60 J=1,N
      IF(INDEX(J,1))60,50,60
 50   TEMP=ABS(A(I,J))
      IF(TEMP-AMAX)60,60,55
 55   IROW=I
      ICOL=J
      AMAX=TEMP
 60   CONTINUE
 70   CONTINUE
      IF(AMAX)170,180,75
 75   INDEX(ICOL,1)=IROW
      IF(IROW-ICOL)80,100,80
 80   DO 90 J=1,N
      TEMP=A(IROW,J)
      A(IROW,J)=A(ICOL,J)
 90   A(ICOL,J)=TEMP
      II=II+1
      INDEX(II,2)=ICOL
 100  PIVOT=A(ICOL,ICOL)
      A(ICOL,ICOL)=1.0
      PIVOT=1.0/PIVOT
      DO 110 J=1,N
 110  A(ICOL,J)=A(ICOL,J)*PIVOT
      DO 140 I=1,N
      IF(I-ICOL)120,140,120
 120  TEMP=A(I,ICOL)
      A(I,ICOL)=0.0
      DO 130 J=1,N
 130  A(I,J)=A(I,J)-A(ICOL,J)*TEMP
 140  CONTINUE
      GOTO 20
 150  ICOL=INDEX(II,2)
      IROW=INDEX(ICOL,1)
      DO 160 I=1,N
      TEMP=A(I,IROW)
      A(I,IROW)=A(I,ICOL)
 160  A(I,ICOL)=TEMP
      II=II-1
 170  IF(II)150,190,150
 180  WRITE(6,1001)
 1001 FORMAT(1H0,2X,11H ZERO PIVOT)
 190  CONTINUE
      RETURN
      END

      SUBROUTINE MATI3(A,N)
C     ******************************************************************
C     *                                                                *
C     *                THE PROGRAM OF INVERSION                        *
C     *                                                                *
C     ******************************************************************
```

```fortran
      DIMENSION INDEX(80,2),A(80,80)                                    KAN11410
      DO 10 I=1,N                                                       KAN11420
10    INDEX(I,1)=0                                                      KAN11430
      II=0                                                              KAN11440
20    AMAX=-1.0                                                         KAN11450
      DO 70 I=1,N                                                       KAN11460
      IF(INDEX(I,1))70,40,70                                            KAN11470
40    DO 60 J=1,N                                                       KAN11480
      IF(INDEX(J,1))60,50,60                                            KAN11490
50    TEMP=ABS(A(I,J))                                                  KAN11500
      IF(TEMP-AMAX)60,60,55                                             KAN11510
55    IROW=I                                                            KAN11520
      ICOL=J                                                            KAN11530
      AMAX=TEMP                                                         KAN11540
60    CONTINUE                                                          KAN11550
70    CONTINUE                                                          KAN11560
      IF(AMAX)170,180,75                                                KAN11570
75    INDEX(ICOL,1)=IROW                                                KAN11580
      IF(IROW-ICOL)80,100,80                                            KAN11590
80    DO 90 J=1,N                                                       KAN11600
      TEMP=A(IROW,J)                                                    KAN11610
      A(IROW,J)=A(ICOL,J)                                               KAN11620
90    A(ICOL,J)=TEMP                                                    KAN11630
      II=II+1                                                           KAN11640
      INDEX(II,2)=ICOL                                                  KAN11650
100   PIVOT=A(ICOL,ICOL)                                                KAN11660
      A(ICOL,ICOL)=1.0                                                  KAN11670
      PIVOT=1.0/PIVOT                                                   KAN11680
      DO 110 J=1,N                                                      KAN11690
110   A(ICOL,J)=A(ICOL,J)*PIVOT                                         KAN11700
      DO 140 I=1,N                                                      KAN11710
      IF(I-ICOL)120,140,120                                             KAN11720
120   TEMP=A(I,ICOL)                                                    KAN11730
      A(I,ICOL)=0.0                                                     KAN11740
      DO 130 J=1,N                                                      KAN11750
130   A(I,J)=A(I,J)-A(ICOL,J)*TEMP                                      KAN11760
140   CONTINUE                                                          KAN11770
      GOTO 20                                                           KAN11780
150   ICOL=INDEX(II,2)                                                  KAN11790
      IROW=INDEX(ICOL,1)                                                KAN11800
      DO 160 I=1,N                                                      KAN11810
      TEMP=A(I,IROW)                                                    KAN11820
      A(I,IROW)=A(I,ICOL)                                               KAN11830
160   A(I,ICOL)=TEMP                                                    KAN11840
      II=II-1                                                           KAN11850
170   IF(II)150,190,150                                                 KAN11860
180   WRITE(6,1001)                                                     KAN11870
1001  FORMAT(1H0,2X,11H ZERO PIVOT)                                     KAN11880
190   CONTINUE                                                          KAN11890
      RETURN                                                            KAN11900
      END                                                               KAN11910
```

Kandif3

The FORTRAN Program for Skewed Bridge Decks by the Kantorovich Finite Difference Method Based on Kirchhoff Plate Theory

```fortran
C       ***************************************************************
C       *                                                             *
C       *                    THE MAIN PROGRAM                         *
C       *                                                             *
C       ***************************************************************
        DIMENSION TSTIF(600,600),PLOAD(600)
        COMMON/TDISP/PLOAD1(600)
        COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
        COMMON/PELEC/EMODU,POISS,THICK
        READ(5,*)NSTRP,NSECT,NZC,NPPOIN,ISUPT,NGUSS,ALENG,BWIDT,PARAK
        NPOIN=(NSTRP+1)*(NSECT+1)
        WRITE(6,100)NSTRP,NSECT,NZC,NPPOIN,ISUPT
 100    FORMAT(2X,'NSTRP=',I4,3X,'NSECT=',I4,
     &      3X,'NZC=',I6,2X,'NPPOIN=',I3,3X,'ISUPT=',I3/)
        READ(5,*)QLOAD
        WRITE(6,101)QLOAD
 101    FORMAT(2X,'QLOAD=',E10.4/)
        IF(NPPOIN.GE.1) THEN
        READ(5,*)((PPOIN(I,J),J=1,2),I=1,NPPOIN)
        ELSE
        END IF
        IF(NZC.GE.1) THEN
        READ(5,*)((IZC(I,J),J=1,2),I=1,NZC)
        ELSE
        END IF
        READ(5,*)EMODU,POISS,THICK
        CALL MSTIF(TSTIF,PLOAD)
        CALL MATINV(TSTIF,2*NPOIN)
        DO 10 I=1,2*NPOIN
        PLOAD1(I)=0.0
        DO 10 K=1,2*NPOIN
 10     PLOAD1(I)=PLOAD1(I)+TSTIF(I,K)*PLOAD(K)
        WRITE(6,104)
 104    FORMAT(/2X,'DISPLACEMENTS'/)
        DO 20 I=1,2*NPOIN
        WRITE(6,105)I,PLOAD1(I)
 105    FORMAT(2X,I3,8X,E16.5)
 20     CONTINUE
        CALL MOMNT
        STOP
        END

        SUBROUTINE MSTIF(TSTIF,PLOAD)
C       ***************************************************************
C       *                                                             *
C       *     THE INTRODUCTION OF DISCRETE BOUNDARY CONDITIONS        *
C       *                                                             *
C       ***************************************************************
        DIMENSION TSTIF(600,600),PLOAD(600)
        COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
        CALL STIFT(TSTIF,PLOAD)
        IF(NZC.GE.1) THEN
        NPOIN=(NSTRP+1)*(NSECT+1)
        DO 20 I=1,NZC
C       I1---NO. OF THE ALLOCATED POINT IN A NODAL LINE.
C       I2---NO. OF THE CONVERTED NODAL LINES, THERE ARE THREE CNLS
C            IN ONE NODAL LINE.
        I1=IZC(I,1)
```

```
      I2=IZC(I,2)                                                       KAN00610
      N1=(I1-1)*2*(NSTRP+1)+I2                                          KAN00620
      N2=(I2-1)*(NSECT+1)+I1                                            KAN00630
      DO 10 K=1,2*NPOIN                                                 KAN00640
      TSTIF(N1,K)=0.0                                                   KAN00650
      TSTIF(K,N2)=0.0                                                   KAN00660
 10   CONTINUE                                                          KAN00670
      TSTIF(N1,N2)=1.0                                                  KAN00680
      PLOAD(N1)=0.0                                                     KAN00690
 20   CONTINUE                                                          KAN00700
      ELSE                                                              KAN00710
      END IF                                                            KAN00720
      RETURN                                                            KAN00730
      END                                                               KAN00740
                                                                        KAN00750
      SUBROUTINE MOMNT                                                  KAN00760
C     ***************************************************************** KAN00770
C     *                                                               * KAN00780
C     *                        BENDING MOMENT                         * KAN00790
C     *                                                               * KAN00800
C     ***************************************************************** KAN00810
      DIMENSION BENMO(3,1),ELOAD(90),PELEM(90)                          KAN00820
      DIMENSION P1(4,1),P2(4,1),P3(4,1)                                 KAN00830
      DIMENSION EE1(3,4),EE2(3,4),EE3(3,4)                              KAN00840
      DIMENSION EED1(3,1),EED2(3,1),EED3(3,1),EED(3,1)                  KAN00850
      DIMENSION DBMAT(3,3),E1MAT(3,4),E2MAT(3,4),E3MAT(3,4)             KAN00860
      COMMON/TDISP/PLOAD1(600)                                          KAN00870
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,            KAN00880
     &             PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK             KAN00890
      CALL MATDB(DBMAT)                                                 KAN00900
      H=1.0/NSECT                                                       KAN00910
      DO 90 ISTRP=1,NSTRP                                               KAN00920
      WRITE(6,*)                                                        KAN00930
      WRITE(6,101)ISTRP                                                 KAN00940
101   FORMAT(2X,'NO. OF STRIP=',I2)                                     KAN00950
      DO 20 I1=1,4*(NSECT+1)                                            KAN00960
20    ELOAD(I1)=PLOAD1(2*(NSECT+1)*(ISTRP-1)+I1)                        KAN00970
      DO 30 J=1,NSECT+1                                                 KAN00980
      DO 30 K=1,4                                                       KAN00990
      PELEM(K+4*(J-1))=ELOAD(J+(K-1)*(NSECT+1))                         KAN01000
30    CONTINUE                                                          KAN01010
      DO 80 IPOIN=2,NSECT                                               KAN01020
      WRITE(6,102)IPOIN                                                 KAN01030
102   FORMAT(2X,'NODE NO.=',I2)                                         KAN01040
      DO 40 K=1,4                                                       KAN01050
      P1(K,1)=PELEM((IPOIN-2)*4+K)                                      KAN01060
      P2(K,1)=PELEM((IPOIN-1)*4+K)                                      KAN01070
40    P3(K,1)=PELEM(IPOIN*4+K)                                          KAN01080
      DO 70 ILINE=1,2                                                   KAN01090
      IF(ILINE.EQ.1) THEN                                               KAN01100
      GX=(ISTRP-1.0)/NSTRP                                              KAN01110
      ELSE                                                              KAN01120
      GX=(ISTRP-1.0)/NSTRP+1.0/NSTRP                                    KAN01130
      END IF                                                            KAN01140
      CALL MATRE(E1MAT,E2MAT,E3MAT,ISTRP,GX)                            KAN01150
      DO 50 I=1,3                                                       KAN01160
      DO 50 J=1,4                                                       KAN01170
      EE1(I,J)=-E2MAT(I,J)/(2.0*H)+E3MAT(I,J)/(H*H)                     KAN01180
      EE2(I,J)=E1MAT(I,J)-2.0*E3MAT(I,J)/(H*H)                          KAN01190
50    EE3(I,J)=E2MAT(I,J)/(2.0*H)+E3MAT(I,J)/(H*H)                      KAN01200
```

```
      CALL MATRM(EED1,EE1,P1,3,4,1,3,4,1)
      CALL MATRM(EED2,EE2,P2,3,4,1,3,4,1)
      CALL MATRM(EED3,EE3,P3,3,4,1,3,4,1)
      DO 60 II=1,3
60    EED(II,1)=EED1(II,1)+EED2(II,1)+EED3(II,1)
      CALL MATRM(BENMO,DBMAT,EED,3,3,1,3,3,1)
      WRITE(6,103)(BENMO(KK,1),KK=1,3)
103   FORMAT(2X,F15.5,5X,F15.5,5X,F15.5)
70    CONTINUE
80    CONTINUE
90    CONTINUE
      RETURN
      END

      SUBROUTINE STIFT(TSTIF,PLOAD)
C     ************************************************************
C     *                                                          *
C     * THE TOTAL STIFFNESS MATRIX & LOAD VECTOR OF THE STRUCTURE *
C     *                                                          *
C     ************************************************************
      DIMENSION TSTIF(600,600),PLOAD(600),ESTIF(90,90),PELEM(90)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
      NPOIN=(NSTRP+1)*(NSECT+1)
      DO 10 I=1,2*NPOIN
      DO 10 J=1,2*NPOIN
10    TSTIF(I,J)=0.0
      DO 15 I=1,2*NPOIN
15    PLOAD(I)=0.0
      IF(NPPOIN.GE.1) THEN
      DO 20 I=1,NPPOIN
      J=PPOIN(I,2)
20    PLOAD(J)=PPOIN(I,1)
      ELSE
      END IF
      DO 60 ISTRP=1,NSTRP
      CALL STIFE(ESTIF,ISTRP)
      DO 25 I1=1,NSECT+1
      DO 25 II=1,4
      IE=4*(I1-1)+II
      IT=2*(NSTRP+1)*(I1-1)+2*(ISTRP-1)+II
      DO 25 J1=1,2
      DO 25 JJ=1,2*(NSECT+1)
      JE=2*(NSECT+1)*(J1-1)+JJ
      JT=2*(NSECT+1)*(ISTRP+J1-2)+JJ
      TSTIF(IT,JT)=TSTIF(IT,JT)+ESTIF(IE,JE)
25    CONTINUE
60    CONTINUE
      IF(QLOAD.NE.0.0) THEN
      DO 80 ISTRP=1,NSTRP
      CALL LOADP(PELEM,ISTRP)
      DO 55 I1=1,NSECT+1
      DO 55 II=1,4
      IE=4*(I1-1)+II
      IT=2*(NSTRP+1)*(I1-1)+2*(ISTRP-1)+II
      PLOAD(IT)=PLOAD(IT)+PELEM(IE)
55    CONTINUE
80    CONTINUE
      ELSE
      END IF
```

```
      RETURN
      END

      SUBROUTINE STIFE(ESTIF,ISTRP)
C     ****************************************************************
C     *                                                              *
C     *          THE STIFFNESS MATRIX OF THE ITH STRIP               *
C     *                                                              *
C     ****************************************************************
      DIMENSION ESTIF2(90,90),ESTIF(90,90)
      DIMENSION AHATM(4,4),BHATM(4,4),CHATM(4,4),DHATM(4,4),EHATM(4,4)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
      DO 10 I=1,4*(NSECT+1)
      DO 10 J=1,4*(NSECT+1)
   10 ESTIF2(I,J)=0.0
      H=1.0/NSECT
      CALL MAABC(AHATM,BHATM,CHATM,DHATM,EHATM,ISTRP,H)
      DO 30 IPOIN=2,NSECT
      IF(IPOIN.EQ.2) THEN
      DO 15 II=1,4
      DO 15 JJ=1,4
      IF(ISUPT.EQ.1) THEN
      IF(JJ.EQ.1.OR.JJ.EQ.3) THEN
      ESTIF2(II+4,JJ+4)=-AHATM(II,JJ)+CHATM(II,JJ)
      ELSE
      ESTIF2(II+4,JJ+4)=AHATM(II,JJ)+CHATM(II,JJ)
      END IF
      ELSE IF(ISUPT.EQ.2) THEN
      ESTIF2(II+4,JJ+4)=AHATM(II,JJ)+CHATM(II,JJ)
      ELSE
      END IF
      ESTIF2(II+4,JJ+8)=DHATM(II,JJ)
      ESTIF2(II+4,JJ+12)=EHATM(II,JJ)
   15 CONTINUE
      ELSE IF(IPOIN.EQ.NSECT) THEN
      DO 20 II=1,4
      DO 20 JJ=1,4
      ESTIF2(II+4*(NSECT-1),JJ+4*(NSECT-3))=AHATM(II,JJ)
      ESTIF2(II+4*(NSECT-1),JJ+4*(NSECT-2))=BHATM(II,JJ)
      IF(ISUPT.EQ.1) THEN
      IF(JJ.EQ.1.OR.JJ.EQ.3) THEN
      ESTIF2(II+4*(NSECT-1),JJ+4*(NSECT-1))=CHATM(II,JJ)-EHATM(II,JJ)
      ELSE
      ESTIF2(II+4*(NSECT-1),JJ+4*(NSECT-1))=CHATM(II,JJ)+EHATM(II,JJ)
      END IF
      ELSE IF(ISUPT.EQ.2) THEN
      ESTIF2(II+4*(NSECT-1),JJ+4*(NSECT-1))=CHATM(II,JJ)+EHATM(II,JJ)
      ELSE
      END IF
   20 CONTINUE
      ELSE
      DO 25 II=1,4
      DO 25 JJ=1,4
      ESTIF2(II+4*(IPOIN-1),JJ+4*(IPOIN-3))=AHATM(II,JJ)
      ESTIF2(II+4*(IPOIN-1),JJ+4*(IPOIN-2))=BHATM(II,JJ)
      ESTIF2(II+4*(IPOIN-1),JJ+4*(IPOIN-1))=CHATM(II,JJ)
      ESTIF2(II+4*(IPOIN-1),JJ+4*IPOIN)=DHATM(II,JJ)
      ESTIF2(II+4*(IPOIN-1),JJ+4*(IPOIN+1))=EHATM(II,JJ)
   25 CONTINUE
```

```
      END IF
 30   CONTINUE
      DO 40 I=1,4
      DO 35 K=1,4*(NSECT+1)
      ESTIF2(K,I)=0.0
      ESTIF2(K,I+4*NSECT)=0.0
 35   CONTINUE
      ESTIF2(I,I)=1.0
      ESTIF2(I+4*NSECT,I+4*NSECT)=1.0
 40   CONTINUE
      DO 50 I=1,4*(NSECT+1)
      DO 50 K=1,4
      DO 50 J=1,NSECT+1
      ESTIF(I,J+(K-1)*(NSECT+1))=ESTIF2(I,K+4*(J-1))
 50   CONTINUE
      RETURN
      END

      SUBROUTINE LOADP(PELEM,ISTRP)
C     ************************************************************
C     *                                                          *
C     *      EQUIVELENT NODAL LOAD VECTOR OF THE ITH STRIP       *
C     *                                                          *
C     ************************************************************
      DIMENSION P0(4),PELEM(90),PX(3),WEIGH(3)
      DIMENSION SHAPN(4),SHA1N(4),SHA2N(4)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
      DO 10 I=1,4*(NSECT+1)
 10   PELEM(I)=0.0
      P0(1)=QLOAD*ALENG*BWIDT/(2.0*NSTRP)
      P0(2)=QLOAD*ALENG*BWIDT/(12.0*NSTRP**2)
      P0(3)=QLOAD*ALENG*BWIDT/(2.0*NSTRP)
      P0(4)=-QLOAD*ALENG*BWIDT/(12.0*NSTRP**2)
      DO 50 IPOIN=2,NSECT
      DO 40 K=1,4
 40   PELEM(K+4*(IPOIN-1))=PELEM(K+4*(IPOIN-1))+P0(K)
 50   CONTINUE
      RETURN
      END

      SUBROUTINE MAABC(AHATM,BHATM,CHATM,DHATM,EHATM,ISTRP,H)
C     ************************************************************
C     *                                                          *
C     *                   TRANSITION MATRIX                      *
C     *                                                          *
C     ************************************************************
      DIMENSION A1MAT(4,4),A2MAT(4,4)
      DIMENSION B1MAT(4,4),B2MAT(4,4)
      DIMENSION C1MAT(4,4),C2MAT(4,4)
      DIMENSION ABARM(4,4),BBARM(4,4),CBARM(4,4),DBARM(4,4),EBARM(4,4)
      DIMENSION AHATM(4,4),BHATM(4,4),CHATM(4,4),DHATM(4,4),EHATM(4,4)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
      CALL MATA1(A1MAT,ISTRP)
      CALL MATA2(A2MAT,ISTRP)
      CALL MATB1(B1MAT,ISTRP)
      CALL MATB2(B2MAT,ISTRP)
      CALL MATC1(C1MAT,ISTRP)
      CALL MATC2(C2MAT,ISTRP)
```

```
      DO 10 I=1,4
      DO 10 J=1,4
      ABARM(I,J)=A1MAT(I,J)
      BBARM(I,J)=B1MAT(J,I)-B1MAT(I,J)
      CBARM(I,J)=C1MAT(I,J)+C1MAT(J,I)-A2MAT(I,J)
      DBARM(I,J)=B2MAT(I,J)-B2MAT(J,I)
 10   EBARM(I,J)=C2MAT(I,J)
      DO 20 I=1,4
      DO 20 J=1,4
      AHATM(I,J)=ABARM(I,J)/H**4-BBARM(I,J)/(2.0*H**3)
      BHATM(I,J)=-4.0*ABARM(I,J)/H**4+BBARM(I,J)/H**3
     &           +CBARM(I,J)/H**2
     &           -DBARM(I,J)/(2.0*H)
      CHATM(I,J)=6.0*ABARM(I,J)/H**4-2.0*CBARM(I,J)/H**2
     &           +EBARM(I,J)
      DHATM(I,J)=-4.0*ABARM(I,J)/H**4-BBARM(I,J)/H**3+CBARM(I,J)/H**2
     &           +DBARM(I,J)/(2.0*H)
 20   EHATM(I,J)=ABARM(I,J)/H**4+BBARM(I,J)/(2.0*H**3)
      RETURN
      END

      SUBROUTINE MATA1(A1MAT,ISTRP)
C     **************************************************************
C     *                                                            *
C     *                    TRANSITION MATRIX                       *
C     *                                                            *
C     **************************************************************
      DIMENSION A1MAT(4,4),PX(3),WEIGH(3)
      DIMENSION DBMAT(3,3),E1MAT(3,4),E2MAT(3,4),E3MAT(3,4)
      DIMENSION B1(3,4),B2(4,4)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
      COMMON/PELEC/EMODU,POISS,THICK
      DO 5 I=1,4
      DO 5 J=1,4
 5    A1MAT(I,J)=0.0
      CALL MATDB(DBMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 30 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATRE(E1MAT,E2MAT,E3MAT,ISTRP,GX)
      CALL MATRM(B1,DBMAT,E3MAT,3,3,4,3,3,4)
      CALL MATRMT(B2,E3MAT,B1,3,4,4,3,4,4)
      DO 20 I=1,4
      DO 20 J=1,4
 20   A1MAT(I,J)=A1MAT(I,J)+ALENG*BWIDT*B2(I,J)*WEIGH(IGUSS)/NSTRP
 30   CONTINUE
      RETURN
      END

      SUBROUTINE MATB1(B1MAT,ISTRP)
C     **************************************************************
C     *                                                            *
C     *                    TRANSITION MATRIX                       *
C     *                                                            *
C     **************************************************************
      DIMENSION B1MAT(4,4),PX(3),WEIGH(3)
      DIMENSION DBMAT(3,3),E1MAT(3,4),E2MAT(3,4),E3MAT(3,4)
      DIMENSION B1(3,4),B2(4,4)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
```

```
     &              PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
      COMMON/PELEC/EMODU,POISS,THICK
      DO 5 I=1,4
      DO 5 J=1,4
 5    B1MAT(I,J)=0.0
      CALL MATDB(DBMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 30 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATRE(E1MAT,E2MAT,E3MAT,ISTRP,GX)
      CALL MATRM(B1,DBMAT,E3MAT,3,3,4,3,3,4)
      CALL MATRMT(B2,E2MAT,B1,3,4,4,3,4,4)
      DO 20 I=1,4
      DO 20 J=1,4
 20   B1MAT(I,J)=B1MAT(I,J)+ALENG*BWIDT*B2(I,J)*WEIGH(IGUSS)/NSTRP
 30   CONTINUE
      RETURN
      END

      SUBROUTINE MATC1(C1MAT,ISTRP)
C     ***********************************************************
C     *                                                         *
C     *                  TRANSITION MATRIX                      *
C     *                                                         *
C     ***********************************************************
      DIMENSION C1MAT(4,4),PX(3),WEIGH(3)
      DIMENSION DBMAT(3,3),E1MAT(3,4),E2MAT(3,4),E3MAT(3,4)
      DIMENSION B1(3,4),B2(4,4)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &              PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
      COMMON/PELEC/EMODU,POISS,THICK
      DO 5 I=1,4
      DO 5 J=1,4
 5    C1MAT(I,J)=0.0
      CALL MATDB(DBMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 30 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATRE(E1MAT,E2MAT,E3MAT,ISTRP,GX)
      CALL MATRM(B1,DBMAT,E3MAT,3,3,4,3,3,4)
      CALL MATRMT(B2,E1MAT,B1,3,4,4,3,4,4)
      DO 20 I=1,4
      DO 20 J=1,4
 20   C1MAT(I,J)=C1MAT(I,J)+ALENG*BWIDT*B2(I,J)*WEIGH(IGUSS)/NSTRP
 30   CONTINUE
      RETURN
      END

      SUBROUTINE MATA2(A2MAT,ISTRP)
C     ***********************************************************
C     *                                                         *
C     *                  TRANSITION MATRIX                      *
C     *                                                         *
C     ***********************************************************
      DIMENSION A2MAT(4,4),PX(3),WEIGH(3)
      DIMENSION DBMAT(3,3),E1MAT(3,4),E2MAT(3,4),E3MAT(3,4)
      DIMENSION B1(3,4),B2(4,4)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &              PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
      COMMON/PELEC/EMODU,POISS,THICK
```

```
      DO 5 I=1,4
      DO 5 J=1,4
5     A2MAT(I,J)=0.0
      CALL MATDB(DBMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 30 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATRE(E1MAT,E2MAT,E3MAT,ISTRP,GX)
      CALL MATRM(B1,DBMAT,E2MAT,3,3,4,3,3,4)
      CALL MATRMT(B2,E2MAT,B1,3,4,4,3,4,4)
      DO 20 I=1,4
      DO 20 J=1,4
20    A2MAT(I,J)=A2MAT(I,J)+ALENG*BWIDT*B2(I,J)*WEIGH(IGUSS)/NSTRP
30    CONTINUE
      RETURN
      END

      SUBROUTINE MATB2(B2MAT,ISTRP)
C     ******************************************************************
C     *                                                                *
C     *                       TRANSITION MATRIX                        *
C     *                                                                *
C     ******************************************************************
      DIMENSION B2MAT(4,4),PX(3),WEIGH(3)
      DIMENSION DBMAT(3,3),E1MAT(3,4),E2MAT(3,4),E3MAT(3,4)
      DIMENSION B1(3,4),B2(4,4)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
      COMMON/PELEC/EMODU,POISS,THICK
      DO 5 I=1,4
      DO 5 J=1,4
5     B2MAT(I,J)=0.0
      CALL MATDB(DBMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 30 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATRE(E1MAT,E2MAT,E3MAT,ISTRP,GX)
      CALL MATRM(B1,DBMAT,E2MAT,3,3,4,3,3,4)
      CALL MATRMT(B2,E1MAT,B1,3,4,4,3,4,4)
      DO 20 I=1,4
      DO 20 J=1,4
20    B2MAT(I,J)=B2MAT(I,J)+ALENG*BWIDT*B2(I,J)*WEIGH(IGUSS)/NSTRP
30    CONTINUE
      RETURN
      END

      SUBROUTINE MATC2(C2MAT,ISTRP)
C     ******************************************************************
C     *                                                                *
C     *                       TRANSITION MATRIX                        *
C     *                                                                *
C     ******************************************************************
      DIMENSION C2MAT(4,4),PX(3),WEIGH(3)
      DIMENSION DBMAT(3,3),E1MAT(3,4),E2MAT(3,4),E3MAT(3,4)
      DIMENSION B1(3,4),B2(4,4)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
      COMMON/PELEC/EMODU,POISS,THICK
      DO 5 I=1,4
      DO 5 J=1,4
```

```
 5      C2MAT(I,J)=0.0
        CALL MATDB(DBMAT)
        CALL GAUSS(PX,WEIGH,ISTRP)
        DO 30 IGUSS=1,NGUSS
        GX=PX(IGUSS)
        CALL MATRE(E1MAT,E2MAT,E3MAT,ISTRP,GX)
        CALL MATRM(B1,DBMAT,E1MAT,3,3,4,3,3,4)
        CALL MATRMT(B2,E1MAT,B1,3,4,4,3,4,4)
        DO 20 I=1,4
        DO 20 J=1,4
 20     C2MAT(I,J)=C2MAT(I,J)+ALENG*BWIDT*B2(I,J)*WEIGH(IGUSS)/NSTRP
 30     CONTINUE
        RETURN
        END

        SUBROUTINE MATRE(E1MAT,E2MAT,E3MAT,ISTRP,GX)
C       ***********************************************************
C       *                                                         *
C       *         THE SHAPE FUNCTION FOR THE X-DIRECTION          *
C       *                                                         *
C       ***********************************************************
        DIMENSION SHA1N(4),SHA2N(4),SHAPN(4)
        DIMENSION E1MAT(3,4),E2MAT(3,4),E3MAT(3,4)
        COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
        DO 10 I=1,3
        DO 10 J=1,4
        E1MAT(I,J)=0.0
        E2MAT(I,J)=0.0
 10     E3MAT(I,J)=0.0
        CALL SHAPE(SHAPN,SHA1N,SHA2N,ISTRP,GX)
        DO 20 J=1,4
        E1MAT(1,J)=-SHA2N(J)/(ALENG**2)
        E1MAT(2,J)=-PARAK**2*SHA2N(J)/(ALENG**2)
        E1MAT(3,J)=2.0*PARAK*SHA2N(J)/(ALENG**2)
        E2MAT(2,J)=2.0*PARAK*SHA1N(J)/(ALENG*BWIDT)
        E2MAT(3,J)=-2.0*SHA1N(J)/(ALENG*BWIDT)
 20     E3MAT(2,J)=-SHAPN(J)/(BWIDT**2)
        RETURN
        END

        SUBROUTINE SHAPE(SHAPN,SHA1N,SHA2N,ISTRP,GX)
C       ***********************************************************
C       *                                                         *
C       *         THE SHAPE FUNCTION FOR THE X-DIRECTION          *
C       *                                                         *
C       ***********************************************************
        DIMENSION SHA1N(4),SHA2N(4),SHAPN(4)
        COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
        GXX=1.0-ISTRP+GX*NSTRP
        SHAPN(1)=1.0-3.0*GXX*GXX+2.0*GXX**3
        SHAPN(2)=GXX*(1.0-2.0*GXX+GXX*GXX)/NSTRP
        SHAPN(3)=3.0*GXX*GXX-2.0*GXX**3
        SHAPN(4)=GXX*(GXX*GXX-GXX)/NSTRP
        SHA1N(1)=(-6.0*GXX+6.0*GXX**2)*NSTRP
        SHA1N(2)=(1.0-4.0*GXX+3.0*GXX**2)
        SHA1N(3)=(6.0*GXX-6.0*GXX**2)*NSTRP
        SHA1N(4)=(-2.0*GXX+3.0*GXX**2)
        SHA2N(1)=(-6.0+12.0*GXX)*NSTRP**2
```

```
      SHA2N(2)=(-4.0+6.0*GXX)*NSTRP
      SHA2N(3)=(6.0-12.0*GXX)*NSTRP**2
      SHA2N(4)=(-2.0+6.0*GXX)*NSTRP
      RETURN
      END

      SUBROUTINE MATDB(DBMAT)
C     ***************************************************************
C     *                                                             *
C     *         THE CONSTITUTION RELATIONSHIP FOR BENDING           *
C     *                                                             *
C     ***************************************************************
      DIMENSION DBMAT(3,3)
      COMMON/PELEC/EMODU,POISS,THICK
      DO 10 I=1,3
      DO 10 J=1,3
 10   DBMAT(I,J)=0.0
      RIGID=EMODU*THICK**3/(12.0*(1.0-POISS**2))
      DBMAT(1,1)=RIGID
      DBMAT(1,2)=POISS*RIGID
      DBMAT(2,1)=POISS*RIGID
      DBMAT(2,2)=RIGID
      DBMAT(3,3)=(1.0-POISS)*RIGID/2.0
      RETURN
      END

      SUBROUTINE GAUSS(PX,WEIGH,ISTRP)
C     ***************************************************************
C     *                                                             *
C     *    GAUSSIAN POINT COORDINATES AND WEIGHTED COFFICIENTS      *
C     *                                                             *
C     ***************************************************************
      DIMENSION PX(3),WEIGH(3)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             PPOIN(11,2),IZC(100,2),ALENG,BWIDT,PARAK
      IF(NGUSS.EQ.3) THEN
      PX(1)=(ISTRP-1.0)/NSTRP+0.22540/(2.0*NSTRP)
      PX(2)=(ISTRP-1.0)/NSTRP+1.00000/(2.0*NSTRP)
      PX(3)=(ISTRP-1.0)/NSTRP+1.77460/(2.0*NSTRP)
      WEIGH(1)=0.555556/2.0
      WEIGH(2)=0.888889/2.0
      WEIGH(3)=0.555556/2.0
      ELSE IF(NGUSS.EQ.2) THEN
      PX(1)=(ISTRP-1.0)/NSTRP+0.42265/(2.0*NSTRP)
      PX(2)=(ISTRP-1.0)/NSTRP+1.57735/(2.0*NSTRP)
      WEIGH(1)=1.0/2.0
      WEIGH(2)=1.0/2.0
      ELSE
      PX(1)=(ISTRP-1.0)/NSTRP+0.50/NSTRP
      WEIGH(1)=1.0
      END IF
      RETURN
      END

      SUBROUTINE MATRM(DB,D,B,L,M,N,LR,MR,NR)
C     ***************************************************************
C     *                                                             *
C     *     MATRIX MULTIPLICATION PROGRAM DB(M,N)=D(M,L)*B(L,N)     *
C     *                                                             *
C     ***************************************************************
```

```
      DIMENSION D(L,M),B(M,N),DB(L,N)
      DO 10 I=1,LR
      DO 10 J=1,NR
      DB(I,J)=0.0
      DO 10 K=1,MR
 10   DB(I,J)=DB(I,J)+D(I,K)*B(K,J)
      RETURN
      END

      SUBROUTINE MATRMT(DBT,D,B,L,M,N,LR,MR,NR)
C     ***********************************************************
C     *                                                         *
C     *      MATRIX MULTIPLICATION AFTER THE FIRST ONE          *
C     *        IS TRANSPOSED   DB(M,N)=D(L,M)*B(L,N)            *
C     *                                                         *
C     ***********************************************************
      DIMENSION D(L,M),B(L,N),DBT(M,N)
      DO 10 I=1,MR
      DO 10 J=1,NR
      DBT(I,J)=0.0
      DO 10 K=1,LR
 10   DBT(I,J)=DBT(I,J)+D(K,I)*B(K,J)
      RETURN
      END

      SUBROUTINE MATINV(A,N)
C     ***********************************************************
C     *                                                         *
C     *              THE PROGRAM OF INVERSION                   *
C     *                                                         *
C     ***********************************************************
      DIMENSION INDEX(600,2),A(600,600)
      DO 10 I=1,N
 10   INDEX(I,1)=0
      II=0
 20   AMAX=-1.0
      DO 70 I=1,N
      IF(INDEX(I,1))70,40,70
 40   DO 60 J=1,N
      IF(INDEX(J,1))60,50,60
 50   TEMP=ABS(A(I,J))
      IF(TEMP-AMAX)60,60,55
 55   IROW=I
      ICOL=J
      AMAX=TEMP
 60   CONTINUE
 70   CONTINUE
      IF(AMAX)170,180,75
 75   INDEX(ICOL,1)=IROW
      IF(IROW-ICOL)80,100,80
 80   DO 90 J=1,N
      TEMP=A(IROW,J)
      A(IROW,J)=A(ICOL,J)
 90   A(ICOL,J)=TEMP
      II=II+1
      INDEX(II,2)=ICOL
 100  PIVOT=A(ICOL,ICOL)
      A(ICOL,ICOL)=1.0
      PIVOT=1.0/PIVOT
      DO 110 J=1,N
```

```
110     A(ICOL,J)=A(ICOL,J)*PIVOT                                KAN06610
        DO 140 I=1,N                                             KAN06620
        IF(I-ICOL)120,140,120                                    KAN06630
120     TEMP=A(I,ICOL)                                           KAN06640
        A(I,ICOL)=0.0                                            KAN06650
        DO 130 J=1,N                                             KAN06660
130     A(I,J)=A(I,J)-A(ICOL,J)*TEMP                             KAN06670
140     CONTINUE                                                 KAN06680
        GOTO 20                                                  KAN06690
150     ICOL=INDEX(II,2)                                         KAN06700
        IROW=INDEX(ICOL,1)                                       KAN06710
        DO 160 I=1,N                                             KAN06720
        TEMP=A(I,IROW)                                           KAN06730
        A(I,IROW)=A(I,ICOL)                                      KAN06740
160     A(I,ICOL)=TEMP                                           KAN06750
        II=II-1                                                  KAN06760
170     IF(II)150,190,150                                        KAN06770
180     WRITE(6,1001)                                            KAN06780
1001    FORMAT(1H0,2X,11H ZERO PIVOT)                            KAN06790
190     CONTINUE                                                 KAN06800
        RETURN                                                   KAN06810
        END                                                      KAN06820
```

Kandif4

The FORTRAN Program for Plane Stress Problems by the Kantorovich Finite Difference Method

```
C     ******************************************************************
C     *                                                                *
C     *                      THE MAIN PROGRAM                          *
C     *                                                                *
C     ******************************************************************
      DIMENSION TSTIF(400,400),PLOAD(400)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      COMMON/TDISP/PLOAD1(400)
      READ(5,*)NSTRP,NSECT,NZC,NPPOIN,ISUPT,NGUSS,NITEM
      NPOIN=(NSTRP+1)*(NSECT+1)
      WRITE(6,100)NSTRP,NSECT,NZC,NPPOIN,ISUPT
100   FORMAT(2X,'NSTRP=',I4,3X,'NSECT=',I4,
     &      3X,'NZC=',I6,2X,'NPPOIN=',I3,3X,'ISUPT=',I3/)
      READ(5,*)QLOAD
      WRITE(6,101)QLOAD
101   FORMAT(2X,'QLOAD=',E10.4/)
      READ(5,*)((COOXQ(I,J),J=1,8),I=1,2)
      WRITE(6,102)(COOXQ(1,J),J=1,8)
102   FORMAT(2X,'COORX=',8F8.2)
      WRITE(6,103)(COOXQ(2,J),J=1,8)
103   FORMAT(2X,'COORY=',8F8.2)
      IF(NPPOIN.GE.1) THEN
      READ(5,*)((PPOIN(I,J),J=1,2),I=1,NPPOIN)
      ELSE
      END IF
      IF(NZC.GE.1) THEN
      READ(5,*)((IZC(I,J),J=1,2),I=1,NZC)
      ELSE
      END IF
      READ(5,*)EMODU,POISS,THICK
      CALL MSTIF(TSTIF,PLOAD)
      CALL MATINV(TSTIF,2*NPOIN)
      DO 10 I=1,2*NPOIN
      PLOAD1(I)=0.0
      DO 10 K=1,2*NPOIN
10    PLOAD1(I)=PLOAD1(I)+TSTIF(I,K)*PLOAD(K)
      WRITE(6,104)
104   FORMAT(/2X,'DISPLACEMENTS'/)
      DO 20 I=1,2*NPOIN
      WRITE(6,105)I,PLOAD1(I)
105   FORMAT(2X,I3,8X,E15.4)
20    CONTINUE
      CALL MOMNT
      STOP
      END

      SUBROUTINE MSTIF(TSTIF,PLOAD)
C     ******************************************************************
C     *                                                                *
C     *      THE INTRODUCTION OF DISCRETE BOUNDARY CONDITIONS          *
C     *                                                                *
C     ******************************************************************
      DIMENSION TSTIF(400,400),PLOAD(400)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      CALL STIFT(TSTIF,PLOAD)
      IF(NZC.GE.1) THEN
      NPOIN=(NSTRP+1)*(NSECT+1)
```

```
      DO 20 I=1,NZC
      I1=IZC(I,1)
      I2=IZC(I,2)
      N1=(I1-1)*2*(NSTRP+1)+I2
      N2=(I2-1)*(NSECT+1)+I1
      DO 10 K=1,2*NPOIN
      TSTIF(N1,K)=0.0
      TSTIF(K,N2)=0.0
 10   CONTINUE
      TSTIF(N1,N2)=1.0
      PLOAD(N1)=0.0
 20   CONTINUE
      ELSE
      END IF
      RETURN
      END

      SUBROUTINE MOMNT
      ***************************************************************
      *                                                             *
      *                       BENDING MOMENT                        *
      *                                                             *
      ***************************************************************
      DIMENSION BENMO(3,1),ELOAD(60),PELEM(60)
      DIMENSION P1(4,1),P2(4,1),P3(4,1)
      DIMENSION EED1(3,1),EED2(3,1),EED3(3,1),EED(3,1)
      DIMENSION DBMAT(3,3),SB1MA(3,8),SB2MA(3,8),SD1MA(3,8),SD2MA(3,8)
      COMMON/TDISP/PLOAD1(400)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      CALL MATDB(DBMAT)
      H=2.0/NSECT
      DO 90 ISTRP=1,NSTRP
      WRITE(6,*)
      WRITE(6,101)ISTRP
 101  FORMAT(2X,'NO. OF STRIP=',I2)
      DO 20 I1=1,4*(NSECT+1)
 20   ELOAD(I1)=PLOAD1(2*(NSECT+1)*(ISTRP-1)+I1)
      DO 30 J=1,NSECT+1
      DO 30 K=1,4
      PELEM(K+4*(J-1))=ELOAD(J+(K-1)*(NSECT+1))
 30   CONTINUE
      DO 80 IPOIN=2,NSECT
      GY=-1.0+(IPOIN-1.0)*H
      WRITE(6,102)IPOIN
 102  FORMAT(2X,'NODE NO.=',I2)
      DO 40 K=1,4
      P1(K,1)=PELEM((IPOIN-2)*4+K)
      P2(K,1)=PELEM((IPOIN-1)*4+K)
 40   P3(K,1)=PELEM(IPOIN*4+K)
      DO 70 ILINE=1,2
      IF(ILINE.EQ.1) THEN
      GX=-1.0+2.0*(ISTRP-1.0)/NSTRP
      ELSE
      GX=-1.0+2.0*(ISTRP-1.0)/NSTRP+2.0/NSTRP
      END IF
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(EED1,SB2MA,P1,3,8,1,3,4,1)
      CALL MATRM(EED2,SB1MA,P2,3,8,1,3,4,1)
      CALL MATRM(EED3,SB2MA,P3,3,8,1,3,4,1)
```

```
      DO 60 II=1,3
60    EED(II,1)=-EED1(II,1)/(2.0*H)+EED2(II,1)+EED3(II,1)/(2.0*H)
      CALL MATRM(BENMO,DBMAT,EED,3,3,1,3,3,1)
      WRITE(6,103)(BENMO(KK,1),KK=1,3)
103   FORMAT(2X,F15.5,5X,F15.5,5X,F15.5)
70    CONTINUE
80    CONTINUE
90    CONTINUE
      RETURN
      END

      SUBROUTINE STIFT(TSTIF,PLOAD)
C     ***************************************************************
C     *                                                             *
C     * THE TOTAL STIFFNESS MATRIX & LOAD VECTOR OF THE STRUCTURE   *
C     *                                                             *
C     ***************************************************************
      DIMENSION TSTIF(400,400),PLOAD(400),ESTIF(60,60),E13(60,60)
      DIMENSION PELEM(120),P2(60),PP(60,1),PP1(60),PP2(60,1)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      NPOIN=(NSTRP+1)*(NSECT+1)
      N2=NITEM-4
      DO 10 I=1,2*NPOIN
      DO 10 J=1,2*NPOIN
10    TSTIF(I,J)=0.0
      DO 15 I=1,2*NPOIN
15    PLOAD(I)=0.0
      IF(NPPOIN.GE.1) THEN
      DO 20 I=1,NPPOIN
      J=PPOIN(I,2)
20    PLOAD(J)=PPOIN(I,1)
      ELSE
      END IF
      DO 60 ISTRP=1,NSTRP
      CALL STIFE(ESTIF,ISTRP,E13)
      DO 25 I1=1,NSECT+1
      DO 25 II=1,4
      IE=4*(I1-1)+II
      IT=2*(NSTRP+1)*(I1-1)+2*(ISTRP-1)+II
      DO 25 J1=1,2
      DO 25 JJ=1,2*(NSECT+1)
      JE=2*(NSECT+1)*(J1-1)+JJ
      JT=2*(NSECT+1)*(ISTRP+J1-2)+JJ
      TSTIF(IT,JT)=TSTIF(IT,JT)+ESTIF(IE,JE)
25    CONTINUE
      IF(QLOAD.NE.0.0) THEN
      CALL LOADP(PELEM,ISTRP)
      DO 40 K=1,NSECT+1
      DO 30 I=1,4
30    PP1(I+4*(K-1))=PELEM(I+NITEM*(K-1))
      DO 35 I=1,N2
35    PP2(I+N2*(K-1),1)=PELEM(I+4+NITEM*(K-1))
40    CONTINUE
      CALL MATRM(PP,E13,PP2,60,60,1,4*(NSECT+1),N2*(NSECT+1),1)
      DO 50 K=1,4*(NSECT+1)
50    P2(K)=PP1(K)-PP(K,1)
      DO 55 I1=1,NSECT+1
      DO 55 II=1,4
      IE=4*(I1-1)+II
```

```
      IT=2*(NSTRP+1)*(I1-1)+2*(ISTRP-1)+II
      PLOAD(IT)=PLOAD(IT)+P2(IE)
 55   CONTINUE
      ELSE
      END IF
 60   CONTINUE
      RETURN
      END

      SUBROUTINE STIFE(ESTIF,ISTRP,E13)
C     ****************************************************************
C     *                                                              *
C     *           THE STIFFNESS MATRIX OF THE ITH STRIP              *
C     *                                                              *
C     ****************************************************************
      DIMENSION ESTIF1(120,120),ESTIF2(60,60),ESTIF(60,60)
      DIMENSION ESTI1(60,60),ESTI2(60,60),ESTI3(60,60),E13(60,60)
      DIMENSION AHATM(8,8),BHATM(8,8),CHATM(8,8),E132(60,60)
      DIMENSION AHAT1(8,8),BHAT1(8,8),CHAT1(8,8),DHAT1(8,8)
      DIMENSION AHAT2(8,8),BHAT2(8,8),CHAT2(8,8),DHAT2(8,8)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 10 I=1,NITEM*(NSECT+1)
      DO 10 J=1,NITEM*(NSECT+1)
 10   ESTIF1(I,J)=0.0
      H=2.0/NSECT
      N1=NITEM/2
      N2=NITEM-4
      DO 30 IPOIN=1,NSECT+1
      GY=-1.0+(IPOIN-1)*H
      CALL MAABC(AHATM,BHATM,CHATM,ISTRP,GY,H,
     &       AHAT1,BHAT1,CHAT1,DHAT1,AHAT2,BHAT2,CHAT2,DHAT2)
      IF(IPOIN.EQ.1) THEN
      DO 15 II=1,NITEM
      DO 15 JJ=1,NITEM
      ESTIF1(II,JJ)=AHAT1(II,JJ)
      ESTIF1(II,JJ+NITEM)=BHAT1(II,JJ)
      ESTIF1(II,JJ+NITEM*2)=CHAT1(II,JJ)
      ESTIF1(II,JJ+NITEM*3)=DHAT1(II,JJ)
 15   CONTINUE
      ELSE IF(IPOIN.EQ.NSECT+1) THEN
      DO 20 II=1,NITEM
      DO 20 JJ=1,NITEM
      ESTIF1(II+NITEM*NSECT,JJ+NITEM*(NSECT-3))=AHAT2(II,JJ)
      ESTIF1(II+NITEM*NSECT,JJ+NITEM*(NSECT-2))=BHAT2(II,JJ)
      ESTIF1(II+NITEM*NSECT,JJ+NITEM*(NSECT-1))=CHAT2(II,JJ)
      ESTIF1(II+NITEM*NSECT,JJ+NITEM*NSECT)=DHAT2(II,JJ)
 20   CONTINUE
      ELSE
      DO 25 II=1,NITEM
      DO 25 JJ=1,NITEM
      ESTIF1(II+NITEM*(IPOIN-1),JJ+NITEM*(IPOIN-2))=AHATM(II,JJ)
      ESTIF1(II+NITEM*(IPOIN-1),JJ+NITEM*(IPOIN-1))=BHATM(II,JJ)
      ESTIF1(II+NITEM*(IPOIN-1),JJ+NITEM*IPOIN)=CHATM(II,JJ)
 25   CONTINUE
      END IF
 30   CONTINUE
      DO 45 K1=1,NSECT+1
      DO 45 I=1,4
      DO 45 K2=1,NSECT+1
```

```
      DO 45 J=1,4
45    ESTIF(I+4*(K1-1),J+4*(K2-1))=ESTIF1(I+NITEM*(K1-1),
     &       J+NITEM*(K2-1))
      DO 50 K1=1,NSECT+1
      DO 50 I=1,4
      DO 50 K2=1,NSECT+1
      DO 50 J=1,N2
50    ESTI1(I+4*(K1-1),J+N2*(K2-1))=ESTIF1(I+NITEM*(K1-1),
     &       J+4+NITEM*(K2-1))
      DO 55 K1=1,NSECT+1
      DO 55 J=1,4
      DO 55 K2=1,NSECT+1
      DO 55 I=1,N2
55    ESTI2(I+N2*(K2-1),J+4*(K1-1))=ESTIF1(I+4+NITEM*(K2-1),
     &       J+NITEM*(K1-1))
      DO 60 K1=1,NSECT+1
      DO 60 I=1,N2
      DO 60 K2=1,NSECT+1
      DO 60 J=1,N2
60    ESTI3(I+N2*(K1-1),J+N2*(K2-1))=ESTIF1
     &       (I+4+NITEM*(K1-1),J+4+NITEM*(K2-1))
      CALL MATI3(ESTI3,N2*(NSECT+1))
      CALL MATRM(E13,ESTI1,ESTI3,60,60,60,
     &           4*(NSECT+1),N2*(NSECT+1),N2*(NSECT+1))
      CALL MATRM(E132,E13,ESTI2,60,60,60,4*(NSECT+1),
     &           N2*(NSECT+1),4*(NSECT+1))
      DO 70 I=1,4*(NSECT+1)
      DO 70 J=1,4*(NSECT+1)
      ESTIF2(I,J)=ESTIF(I,J)-E132(I,J)
70    CONTINUE
      DO 80 I=1,4*(NSECT+1)
      DO 80 K=1,4
      DO 80 J=1,NSECT+1
      ESTIF(I,J+(K-1)*(NSECT+1))=ESTIF2(I,K+4*(J-1))
80    CONTINUE
      RETURN
      END

      SUBROUTINE LOADP(PELEM,ISTRP)
      ****************************************************************
      *                                                              *
      *        EQUIVELENT NODAL LOAD VECTOR OF THE ITH STRIP         *
      *                                                              *
      ****************************************************************
      DIMENSION P0(8),PELEM(120),PX(3),WEIGH(3)
      DIMENSION SHAPN(2,8),SHADN(2,8)
      DIMENSION XJACO(2,2),XTR2Y(2),XTRXY(2)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 10 I=1,NITEM*(NSECT+1)
10    PELEM(I)=0.0
      H=2.0/NSECT
      N1=NITEM/2
      PX(1)=-1.0+2.0*(ISTRP-1)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1)/NSTRP+1.77460/NSTRP
      DO 50 IPOIN=1,NSECT+1
      DO 20 I=1,NITEM
20    P0(I)=0.0
```

```fortran
      GY=-1.0+(IPOIN-1)*H
      DO 30 IGUSS=1,3
      GX=PX(IGUSS)
      CALL SHAPE(SHAPN,SHADN,ISTRP,GX)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 25 K=1,N1
   25 P0(2*K-1)=P0(2*K-1)+QLOAD*SHAPN(1,2*K-1)*DETJ*WEIGH(IGUSS)/NSTRP
   30 CONTINUE
      DO 40 K=1,NITEM
   40 PELEM(K+NITEM*(IPOIN-1))=PELEM(K+NITEM*(IPOIN-1))+P0(K)
   50 CONTINUE
      RETURN
      END

      SUBROUTINE MAABC(AHATM,BHATM,CHATM,ISTRP,GY,H,
     &           AHAT1,BHAT1,CHAT1,DHAT1,AHAT2,BHAT2,CHAT2,DHAT2)
C     ****************************************************************
C     *                                                              *
C     *                      TRANSITION MATRIX                       *
C     *                                                              *
C     ****************************************************************
      DIMENSION ABEMA(8,8),BBEMA(8,8),CBEMA(8,8)
      DIMENSION ABEPM(8,8),BBEPM(8,8)
      DIMENSION ABARM(8,8),BBARM(8,8),CBARM(8,8)
      DIMENSION AHATM(8,8),BHATM(8,8),CHATM(8,8)
      DIMENSION AHAT1(8,8),BHAT1(8,8),CHAT1(8,8),DHAT1(8,8)
      DIMENSION AHAT2(8,8),BHAT2(8,8),CHAT2(8,8),DHAT2(8,8)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      CALL MAABE(ABEMA,ISTRP,GY)
      CALL MABBE(BBEMA,ISTRP,GY)
      CALL MACBE(CBEMA,ISTRP,GY)
      CALL MABEP(ABEPM,ISTRP,GY)
      CALL MBBEP(BBEPM,ISTRP,GY)
      DO 20 I=1,NITEM
      DO 20 J=1,NITEM
      ABARM(I,J)=ABEMA(I,J)
      BBARM(I,J)=ABEPM(I,J)+BBEMA(J,I)-BBEMA(I,J)
   20 CBARM(I,J)=BBEPM(J,I)-CBEMA(I,J)
      DO 30 I=1,NITEM
      DO 30 J=1,NITEM
      AHATM(I,J)=ABARM(I,J)/(H*H)-BBARM(I,J)/(2.0*H)
      BHATM(I,J)=-2.0*ABARM(I,J)/(H*H)+CBARM(I,J)
   30 CHATM(I,J)=ABARM(I,J)/(H*H)+BBARM(I,J)/(2.0*H)
      DO 40 I=1,NITEM
      DO 40 J=1,NITEM
      AHAT1(I,J)=-2.0*ABARM(I,J)/(H*H)-3.0*BBARM(I,J)/(2.0*H)
     &           +CBARM(I,J)
      BHAT1(I,J)=5.0*ABARM(I,J)/(H*H)+2.0*BBARM(I,J)/H
      CHAT1(I,J)=-4.0*ABARM(I,J)/(H*H)-BBARM(I,J)/(2.0*H)
   40 DHAT1(I,J)=ABARM(I,J)/(H*H)
      DO 50 I=1,NITEM
      DO 50 J=1,NITEM
      AHAT2(I,J)=-ABARM(I,J)/(H*H)
      BHAT2(I,J)=4.0*ABARM(I,J)/(H*H)+BBARM(I,J)/(2.0*H)
      CHAT2(I,J)=-5.0*ABARM(I,J)/(H*H)-2.0*BBARM(I,J)/H
   50 DHAT2(I,J)=2.0*ABARM(I,J)/(H*H)+3.0*BBARM(I,J)/(2.0*H)
     &           +CBARM(I,J)
      RETURN
      END
```

```
      SUBROUTINE MAABE(ABEMA,ISTRP,GY)
C     ****************************************************************
C     *                                                              *
C     *                     TRANSITION MATRIX                        *
C     *                                                              *
C     ****************************************************************
      DIMENSION ABEMA(8,8),PX(3),DBMAT(3,3)
      DIMENSION XJACO(2,2),AATR1(3,8),AATR2(8,8)
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,8),SD2MA(3,8)
      DIMENSION SB1MA(3,8),SB2MA(3,8),WEIGH(3)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
 5    ABEMA(I,J)=0.0
      CALL MATDB(DBMAT)
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 20 IGUSS=1,3
      GX=PX(IGUSS)
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,8,3,3,NITEM)
      CALL MATRMT(AATR2,SB2MA,AATR1,3,8,8,3,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
 10   ABEMA(I,J)=ABEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
 20   CONTINUE
      RETURN
      END

      SUBROUTINE MABEP(ABEPM,ISTRP,GY)
C     ****************************************************************
C     *                                                              *
C     *                     TRANSITION MATRIX                        *
C     *                                                              *
C     ****************************************************************
      DIMENSION ABEPM(8,8),PX(3),DBMAT(3,3)
      DIMENSION XJACO(2,2),AATR1(3,8),AATR2(8,8),AATR3(8,8)
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,8),SD2MA(3,8)
      DIMENSION SB1MA(3,8),SB2MA(3,8),WEIGH(3)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
 5    ABEPM(I,J)=0.0
      CALL MATDB(DBMAT)
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 20 IGUSS=1,3
      GX=PX(IGUSS)
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,8,3,3,NITEM)
```

```
      CALL MATRMT(AATR2,SD2MA,AATR1,3,8,8,3,NITEM,NITEM)
      CALL MATRMT(AATR3,SB2MA,AATR1,3,8,8,3,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 15 I=1,NITEM
      DO 15 J=1,NITEM
 15   ABEPM(I,J)=ABEPM(I,J)+(2.0*AATR2(I,J)*DETJ+AATR3(I,J)*DJDY)
     &           *WEIGH(IGUSS)/NSTRP
 20   CONTINUE
      RETURN
      END

      SUBROUTINE MABBE(BBEMA,ISTRP,GY)
C     *****************************************************************
C     *                                                               *
C     *                    TRANSITION MATRIX                          *
C     *                                                               *
C     *****************************************************************
      DIMENSION BBEMA(8,8),PX(3),DBMAT(3,3)
      DIMENSION XJACO(2,2),AATR1(3,8),AATR2(8,8)
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,8),SD2MA(3,8)
      DIMENSION SB1MA(3,8),SB2MA(3,8),WEIGH(3)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
 5    BBEMA(I,J)=0.0
      CALL MATDB(DBMAT)
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 20 IGUSS=1,3
      GX=PX(IGUSS)
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,8,3,3,NITEM)
      CALL MATRMT(AATR2,SB1MA,AATR1,3,8,8,3,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
 10   BBEMA(I,J)=BBEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
 20   CONTINUE
      RETURN
      END

      SUBROUTINE MBBEP(BBEPM,ISTRP,GY)
C     *****************************************************************
C     *                                                               *
C     *                    TRANSITION MATRIX                          *
C     *                                                               *
C     *****************************************************************
      DIMENSION BBEPM(8,8),PX(3),DBMAT(3,3)
      DIMENSION XJACO(2,2),AATR1(3,8),AATR2(8,8)
      DIMENSION AATR3(3,8),AATR4(8,8),AATR5(8,8)
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,8),SD2MA(3,8)
      DIMENSION SB1MA(3,8),SB2MA(3,8),WEIGH(3)
      COMMON/PELEC/EMODU,POISS,THICK
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DATA WEIGH/0.555556,0.888889,0.555556/
```

```
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
    5 BBEPM(I,J)=0.0
      CALL MATDB(DBMAT)
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 20 IGUSS=1,3
      GX=PX(IGUSS)
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,8,3,3,NITEM)
      CALL MATRMT(AATR2,SD1MA,AATR1,3,8,8,3,NITEM,NITEM)
      CALL MATRM(AATR3,DBMAT,SD2MA,3,3,8,3,3,NITEM)
      CALL MATRMT(AATR4,SB1MA,AATR3,3,8,8,3,NITEM,NITEM)
      CALL MATRMT(AATR5,SB1MA,AATR1,3,8,8,3,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 15 I=1,NITEM
      DO 15 J=1,NITEM
   15 BBEPM(I,J)=BBEPM(I,J)+((AATR2(I,J)+AATR4(I,J))*DETJ+AATR5(I,J)
     &             *DJDY)*WEIGH(IGUSS)/NSTRP
   20 CONTINUE
      RETURN
      END

      SUBROUTINE MACBE(CBEMA,ISTRP,GY)
C     ************************************************************
C     *                                                          *
C     *                  TRANSITION MATRIX                       *
C     *                                                          *
C     ************************************************************
      DIMENSION CBEMA(8,8),DBMAT(3,3),PX(3)
      DIMENSION XJACO(2,2),AATR1(3,8),AATR2(8,8)
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,8),SD2MA(3,8)
      DIMENSION SB1MA(3,8),SB2MA(3,8),WEIGH(3)
      COMMON/PELEC/EMODU,POISS,THICK
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
    5 CBEMA(I,J)=0.0
      CALL MATDB(DBMAT)
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 20 IGUSS=1,3
      GX=PX(IGUSS)
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DBMAT,SB1MA,3,3,8,3,3,NITEM)
      CALL MATRMT(AATR2,SB1MA,AATR1,3,8,8,3,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
   10 CBEMA(I,J)=CBEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
   20 CONTINUE
      RETURN
      END

      SUBROUTINE MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
C     ************************************************************
```

```
C     *                                                                  *
C     *                     BENDING S MATRIX                             *
C     *                                                                  *
C     ********************************************************************
      DIMENSION SB1MA(3,8),SD1MA(3,8),SD2MA(3,8)
      DIMENSION TRLB1(3,2),TRLB2(3,2),TLB1D(3,2),TLB2D(3,2)
      DIMENSION SHAPN(2,8),SHADN(2,8),SB2MA(3,8)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 5 I=1,3
      DO 5 J=1,NITEM
      SB1MA(I,J)=0.0
      SB2MA(I,J)=0.0
      SD1MA(I,J)=0.0
 5    SD2MA(I,J)=0.0
      CALL MATLB(TRLB1,TRLB2,GX,GY)
      CALL MALBD(TLB1D,TLB2D,GX,GY)
      CALL SHAPE(SHAPN,SHADN,ISTRP,GX)
      CALL MATRM(SB1MA,TRLB1,SHADN,3,2,8,3,2,NITEM)
      CALL MATRM(SB2MA,TRLB2,SHAPN,3,2,8,3,2,NITEM)
      CALL MATRM(SD1MA,TLB1D,SHADN,3,2,8,3,2,NITEM)
      CALL MATRM(SD2MA,TLB2D,SHAPN,3,2,8,3,2,NITEM)
      RETURN
      END

      SUBROUTINE MATLB(TRLB1,TRLB2,GX,GY)
C     ********************************************************************
C     *
C     *     OPERATOR TRANSFORMATION MATRIX FOR BENDING STRAIN            *
C     *                                                                  *
C     ********************************************************************
      DIMENSION TRLB1(3,2),TRLB2(3,2),XJACO(2,2),XTRXY(2),XTR2Y(2)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,3
      DO 10 J=1,2
      TRLB1(I,J)=0.0
 10   TRLB2(I,J)=0.0
      TRLB1(1,1)=XJACO(2,2)/DETJ
      TRLB1(2,2)=-XJACO(2,1)/DETJ
      TRLB1(3,1)=-XJACO(2,1)/DETJ
      TRLB1(3,2)=XJACO(2,2)/DETJ
      TRLB2(1,1)=-XJACO(1,2)/DETJ
      TRLB2(2,2)=XJACO(1,1)/DETJ
      TRLB2(3,1)=XJACO(1,1)/DETJ
      TRLB2(3,2)=-XJACO(1,2)/DETJ
      RETURN
      END

      SUBROUTINE MALBD(TLB1D,TLB2D,GX,GY)
C     ********************************************************************
C     *
C     *     OPERATOR TRANSFORMATION MATRIX FOR BENDING STRAIN            *
C     *                                                                  *
C     ********************************************************************
      DIMENSION TLB1D(3,2),TLB2D(3,2),XJACO(2,2),XTRXY(2),XTR2Y(2)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,3
      DO 10 J=1,2
      TLB1D(I,J)=0.0
 10   TLB2D(I,J)=0.0
```

```
      TLB1D(1,1)=XTR2Y(2)/DETJ-DJDY*XJACO(2,2)/(DETJ**2)
      TLB1D(2,2)=-XTR2Y(1)/DETJ+DJDY*XJACO(2,1)/(DETJ**2)
      TLB1D(3,1)=TLB1D(2,2)
      TLB1D(3,2)=TLB1D(1,1)
      TLB2D(1,1)=-XTRXY(2)/DETJ+DJDY*XJACO(1,2)/(DETJ**2)
      TLB2D(2,2)=XTRXY(1)/DETJ-DJDY*XJACO(1,1)/(DETJ**2)
      TLB2D(3,1)=TLB2D(2,2)
      TLB2D(3,2)=TLB2D(1,1)
      RETURN
      END

      SUBROUTINE SHAPE(SHAPN,SHADN,ISTRP,GX)
C     ****************************************************************
C     *                                                              *
C     *          THE SHAPE FUNCTION FOR THE X-DIRECTION              *
C     *                                                              *
C     ****************************************************************
      DIMENSION SHA(4),SHA1(4),SHAPN(2,8),SHADN(2,8)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &          NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      N1=NITEM/2
      DO 10 I=1,2
      DO 10 J=1,NITEM
      SHAPN(I,J)=0.0
 10   SHADN(I,J)=0.0
      GXX=1.0-ISTRP+(GX+1.0)*NSTRP/2.0
      IF(N1.EQ.2) THEN
      SHA(1)=1.0-GXX
      SHA(2)=GXX
      SHA1(1)=-NSTRP/2.0
      SHA1(2)=NSTRP/2.0
      ELSE IF(N1.EQ.3) THEN
      SHA(1)=1.0-3.0*GXX+2.0*GXX**2
      SHA(2)=-GXX+2.0*GXX**2
      SHA(3)=4.0*GXX-4.0*GXX**2
      SHA1(1)=(-1.5+2.0*GXX)*NSTRP
      SHA1(2)=(-0.5+2.0*GXX)*NSTRP
      SHA1(3)=(2.0-4.0*GXX)*NSTRP
      ELSE IF(N1.EQ.4) THEN
      SHA(1)=1.0-5.5*GXX+9.0*GXX**2-4.5*GXX**3
      SHA(2)=GXX-4.5*GXX**2+4.5*GXX**3
      SHA(3)=9.0*GXX-22.5*GXX**2+13.5*GXX**3
      SHA(4)=-4.5*GXX+18.0*GXX**2-13.5*GXX**3
      SHA1(1)=(-2.75+9.0*GXX-6.75*GXX**2)*NSTRP
      SHA1(2)=(0.5-4.5*GXX+6.75*GXX**2)*NSTRP
      SHA1(3)=(4.5-22.5*GXX+20.25*GXX**2)*NSTRP
      SHA1(4)=(-2.25+18.0*GXX-20.25*GXX**2)*NSTRP
      ELSE
      END IF
      DO 20 I=1,N1
      SHAPN(1,2*I-1)=SHA(I)
      SHAPN(2,2*I)=SHA(I)
      SHADN(1,2*I-1)=SHA1(I)
      SHADN(2,2*I)=SHA1(I)
 20   CONTINUE
      RETURN
      END

      SUBROUTINE TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
C     ****************************************************************
```

```
C     *
C     *             THE COORDINATE TRANSFORMATION SUBROUTINE              *
C     *                                                                   *
C     ********************************************************************
      DIMENSION XJACO(2,2),SHAP(2,8),PX(8),PY(8)
      DIMENSION TRXY(8),TR2Y(8),XTRXY(2),XTR2Y(2)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &              NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DATA PX/-1.0,-1.0,-1.0,-1.0,1.0,1.0,1.0,1.0/
      DATA PY/-1.0,-0.33333,0.33333,1.0,
     &        -1.0,-0.33333,0.33333,1.0/
      DO 20 I=1,8
      IF(I.EQ.1.OR.I.EQ.4.OR.I.EQ.5.OR.I.EQ.8) THEN
      SHAP(1,I)=PX(I)*(9.0*GY*GY-1.0)*(1.0+PY(I)*GY)/32.0
      SHAP(2,I)=(1.0+PX(I)*GX)*(18.0*GY+27.0*PY(I)*GY*GY-PY(I))/32.0
      TR2Y(I)=9.0*(1.0+GX*PX(I))*(1.0+3.0*GY*PY(I))/16.0
      TRXY(I)=PX(I)*(18.0*GY+27.0*GY*GY*PY(I)-PY(I))/32.0
      ELSE
      SHAP(1,I)=9.0*PX(I)*(1.0-GY*GY)*(1.0+9.0*PY(I)*GY)/32.0
      SHAP(2,I)=9.0*(1.0+PX(I)*GX)*(-2.0*GY-27.0*PY(I)*GY*GY
     &          +9.0*PY(I))/32.0
      TR2Y(I)=9.0*(1.0+GX*PX(I))*(-1.0-27.0*GY*PY(I))/16.0
      TRXY(I)=9.0*PX(I)*(-2.0*GY-27.0*GY*GY*PY(I)+9.0*PY(I))/32.0
      END IF
 20   CONTINUE
      DO 40 I=1,2
      DO 40 J=1,2
      XJACO(I,J)=0.0
      DO 30 K=1,8
 30   XJACO(I,J)=XJACO(I,J)+SHAP(I,K)*COOXQ(J,K)
 40   CONTINUE
      DETJ=XJACO(1,1)*XJACO(2,2)-XJACO(1,2)*XJACO(2,1)
      DO 55 I=1,2
      XTR2Y(I)=0.0
      XTRXY(I)=0.0
      DO 50 K=1,8
      XTR2Y(I)=XTR2Y(I)+TR2Y(K)*COOXQ(I,K)
 50   XTRXY(I)=XTRXY(I)+TRXY(K)*COOXQ(I,K)
 55   CONTINUE
      DJDY=XJACO(1,1)*XTR2Y(2)+XJACO(2,2)*XTRXY(1)
     &     -XJACO(2,1)*XTRXY(2)-XJACO(1,2)*XTR2Y(1)
      RETURN
      END

      SUBROUTINE MATDB(DBMAT)
C     ********************************************************************
C     *
C     *         THE CONSTITUTION RELATIONSHIP FOR BENDING                 *
C     *                                                                   *
C     ********************************************************************
      DIMENSION DBMAT(3,3)
      COMMON/PELEC/EMODU,POISS,THICK
      DO 10 I=1,3
      DO 10 J=1,3
 10   DBMAT(I,J)=0.0
      RIGID=EMODU*THICK/(1.0-POISS**2)
      DBMAT(1,1)=RIGID
      DBMAT(1,2)=POISS*RIGID
      DBMAT(2,1)=POISS*RIGID
      DBMAT(2,2)=RIGID
```

```
      DBMAT(3,3)=(1.0-POISS)*RIGID/2.0
      RETURN
      END

      SUBROUTINE GAUSS(PX,WEIGH,ISTRP)
C**********************************************************************
C*                                                                    *
C*    GAUSSIAN POINT COORDINATES AND WEIGHTED COFFICIENTS             *
C*                                                                    *
C**********************************************************************
      DIMENSION PX(3),WEIGH(3)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      IF(NGUSS.EQ.3) THEN
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      WEIGH(1)=0.555556
      WEIGH(2)=0.888889
      WEIGH(3)=0.555556
      ELSE IF(NGUSS.EQ.2) THEN
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.42265/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.57735/NSTRP
      WEIGH(1)=1.0
      WEIGH(2)=1.0
      ELSE
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      WEIGH(1)=2.0
      END IF
      RETURN
      END

      SUBROUTINE MATRM(DB,D,B,L,M,N,LR,MR,NR)
C**********************************************************************
C*                                                                    *
C*    MATRIX MULTIPLICATION PROGRAM DB(M,N)=D(M,L)*B(L,N)             *
C*                                                                    *
C**********************************************************************
      DIMENSION D(L,M),B(M,N),DB(L,N)
      DO 10 I=1,LR
      DO 10 J=1,NR
      DB(I,J)=0.0
      DO 10 K=1,MR
10    DB(I,J)=DB(I,J)+D(I,K)*B(K,J)
      RETURN
      END

      SUBROUTINE MATRMT(DBT,D,B,L,M,N,LR,MR,NR)
C**********************************************************************
C*                                                                    *
C*        MATRIX MULTIPLICATION AFTER THE FIRST ONE                   *
C*          IS TRANSPOSED     DB(M,N)=D(L,M)*B(L,N)                   *
C*                                                                    *
C**********************************************************************
      DIMENSION D(L,M),B(L,N),DBT(M,N)
      DO 10 I=1,MR
      DO 10 J=1,NR
      DBT(I,J)=0.0
      DO 10 K=1,LR
10    DBT(I,J)=DBT(I,J)+D(K,I)*B(K,J)
```

```
      RETURN
      END

      SUBROUTINE MATINV(A,N)
C     ****************************************************************
C     *                                                              *
C     *                THE PROGRAM OF INVERSION                      *
C     *                                                              *
C     ****************************************************************
      DIMENSION INDEX(400,2),A(400,400)
      DO 10 I=1,N
 10   INDEX(I,1)=0
      II=0
 20   AMAX=-1.0
      DO 70 I=1,N
      IF(INDEX(I,1))70,40,70
 40   DO 60 J=1,N
      IF(INDEX(J,1))60,50,60
 50   TEMP=ABS(A(I,J))
      IF(TEMP-AMAX)60,60,55
 55   IROW=I
      ICOL=J
      AMAX=TEMP
 60   CONTINUE
 70   CONTINUE
      IF(AMAX)170,180,75
 75   INDEX(ICOL,1)=IROW
      IF(IROW-ICOL)80,100,80
 80   DO 90 J=1,N
      TEMP=A(IROW,J)
      A(IROW,J)=A(ICOL,J)
 90   A(ICOL,J)=TEMP
      II=II+1
      INDEX(II,2)=ICOL
 100  PIVOT=A(ICOL,ICOL)
      A(ICOL,ICOL)=1.0
      PIVOT=1.0/PIVOT
      DO 110 J=1,N
 110  A(ICOL,J)=A(ICOL,J)*PIVOT
      DO 140 I=1,N
      IF(I-ICOL)120,140,120
 120  TEMP=A(I,ICOL)
      A(I,ICOL)=0.0
      DO 130 J=1,N
 130  A(I,J)=A(I,J)-A(ICOL,J)*TEMP
 140  CONTINUE
      GOTO 20
 150  ICOL=INDEX(II,2)
      IROW=INDEX(ICOL,1)
      DO 160 I=1,N
      TEMP=A(I,IROW)
      A(I,IROW)=A(I,ICOL)
 160  A(I,ICOL)=TEMP
      II=II-1
 170  IF(II)150,190,150
 180  WRITE(6,1001)
 1001 FORMAT(1H0,2X,11H ZERO PIVOT)
 190  CONTINUE
      RETURN
      END
```

```
      SUBROUTINE MATI3(A,N)
C     *************************************************************
C     *                                                           *
C     *               THE PROGRAM OF INVERSION                    *
C     *                                                           *
C     *************************************************************
      DIMENSION INDEX(60,2),A(60,60)
      DO 10 I=1,N
 10   INDEX(I,1)=0
      II=0
 20   AMAX=-1.0
      DO 70 I=1,N
      IF(INDEX(I,1))70,40,70
 40   DO 60 J=1,N
      IF(INDEX(J,1))60,50,60
 50   TEMP=ABS(A(I,J))
      IF(TEMP-AMAX)60,60,55
 55   IROW=I
      ICOL=J
      AMAX=TEMP
 60   CONTINUE
 70   CONTINUE
      IF(AMAX)170,180,75
 75   INDEX(ICOL,1)=IROW
      IF(IROW-ICOL)80,100,80
 80   DO 90 J=1,N
      TEMP=A(IROW,J)
      A(IROW,J)=A(ICOL,J)
 90   A(ICOL,J)=TEMP
      II=II+1
      INDEX(II,2)=ICOL
100   PIVOT=A(ICOL,ICOL)
      A(ICOL,ICOL)=1.0
      PIVOT=1.0/PIVOT
      DO 110 J=1,N
110   A(ICOL,J)=A(ICOL,J)*PIVOT
      DO 140 I=1,N
      IF(I-ICOL)120,140,120
120   TEMP=A(I,ICOL)
      A(I,ICOL)=0.0
      DO 130 J=1,N
130   A(I,J)=A(I,J)-A(ICOL,J)*TEMP
140   CONTINUE
      GOTO 20
150   ICOL=INDEX(II,2)
      IROW=INDEX(ICOL,1)
      DO 160 I=1,N
      TEMP=A(I,IROW)
      A(I,IROW)=A(I,ICOL)
160   A(I,ICOL)=TEMP
      II=II-1
170   IF(II)150,190,150
180   WRITE(6,1001)
1001  FORMAT(1H0,2X,11H ZERO PIVOT)
190   CONTINUE
      RETURN
      END
```

Kandif5

The FORTRAN Program for General Slab Bridge Decks by the Spline Kantorovich Method Based on Mindlin Plate Theory

```
C       *************************************************************
C       *                                                           *
C       *                  THE MAIN PROGRAM                         *
C       *                                                           *
C       *************************************************************
        DIMENSION TSTIF(600,600),PLOAD(600),PLOAD1(600)
        COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100)
        COMMON/PELEC/EMODU,POISS,THICK
        COMMON/PENAL/NZC1,IZC1(20,2)
        COMMON/COFFE/ALPHA
        READ(5,*)NSTRP,NSECT,NZC,NPPOIN,ISUPT,NGUSS,NITEM,ALPHA
        NPOIN=(NSTRP+1)*(NSECT+1)
        WRITE(6,100)NSTRP,NSECT,NZC,NPPOIN,ISUPT
 100    FORMAT(2X,'NSTRP=',I4,3X,'NSECT=',I4,
     &       3X,'NZC=',I6,2X,'NPPOIN=',I3,3X,'ISUPT=',I3/)
        READ(5,*)QLOAD
        WRITE(6,101)QLOAD
 101    FORMAT(2X,'QLOAD=',E10.4/)
        READ(5,*)((COOXQ(I,J),J=1,8),I=1,2)
        WRITE(6,102)(COOXQ(1,J),J=1,8)
 102    FORMAT(2X,'COORX=',8F8.2)
        WRITE(6,103)(COOXQ(2,J),J=1,8)
 103    FORMAT(2X,'COORY=',8F8.2)
        IF(NPPOIN.GE.1) THEN
        READ(5,*)((PPOIN(I,J),J=1,2),I=1,NPPOIN)
        ELSE
        END IF
        IF(NZC.GE.1) THEN
        READ(5,*)(IZC(I),I=1,NZC)
        ELSE
        END IF
        READ(5,*)EMODU,POISS,THICK
        READ(5,*)NZC1
        IF(NZC1.GE.1) THEN
        READ(5,*)((IZC1(I,J),J=1,2),I=1,NZC1)
        ELSE
        END IF
        CALL MSTIF(TSTIF,PLOAD)
        CALL MATINV(TSTIF,3*NPOIN)
        DO 15 I=1,3*NPOIN
        PLOAD1(I)=0.0
        DO 15 K=1,3*NPOIN
 15     PLOAD1(I)=PLOAD1(I)+TSTIF(I,K)*PLOAD(K)
        WRITE(6,104)
 104    FORMAT(/2X,'THE DISPLACEMENT PARAMETERS'/)
        DO 18 I=1,3*NPOIN
        WRITE(6,105)I,PLOAD1(I)
 105    FORMAT(2X,I3,8X,E15.4)
 18     CONTINUE
        STOP
        END

        SUBROUTINE MSTIF(TSTIF,PLOAD)
C       *************************************************************
C       *                                                           *
C       *        THE INTRODUCTION OF BOUNDARY CONDITIONS            *
C       *                                                           *
C       *************************************************************
        DIMENSION TSTIF(400,400),PLOAD(400)
```

```
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100)
      COMMON/COFFE/ALPHA
      COMMON/PENAL/NZC1,IZC1(20,2)
      CALL STIFT(TSTIF,PLOAD)
      NPOIN=(NSTRP+1)*(NSECT+1)
      IF(NZC1.GE.1) THEN
      DO 20 I=1,NZC1
      N1=IZC1(I,1)
      N2=(N1-1)*(NSECT+1)+IZC1(I,2)
      DO 10 K=1,NSECT+1
      IF(N2.GT.1) THEN
      TSTIF(N1+(K-1)*3*(NSTRP+1),N2-1)=TSTIF(N1+(K-1)*3*(NSTRP+1),
     &                                 N2-1)+ALPHA/6.0
      ELSE
      END IF
      TSTIF(N1+(K-1)*3*(NSTRP+1),N2)=TSTIF(N1+(K-1)*3*(NSTRP+1),N2)
     &                               +2.0*ALPHA/3.0
      IF(N2.LT.3*NPOIN) THEN
      TSTIF(N1+(K-1)*3*(NSTRP+1),N2+1)=TSTIF(N1+(K-1)*3*(NSTRP+1),
     &                                 N2+1)+ALPHA/6.0
      ELSE
      END IF
10    CONTINUE
20    CONTINUE
      ELSE
      END IF
      IF(NZC.GE.1) THEN
      DO 50 I=1,NZC
      N=IZC(I)
      DO 40 IPOIN=1,NSECT+1
      DO 30 K=1,3*NPOIN
      TSTIF(N+(IPOIN-1)*3*(NSTRP+1),K)=0.0
      TSTIF(K,IPOIN+(N-1)*(NSECT+1))=0.0
30    CONTINUE
      TSTIF(N+(IPOIN-1)*3*(NSTRP+1),IPOIN+(N-1)*(NSECT+1))=1.0
      PLOAD(N+(IPOIN-1)*3*(NSTRP+1))=0.0
40    CONTINUE
50    CONTINUE
      ELSE
      END IF
      RETURN
      END

      SUBROUTINE STIFT(TSTIF,PLOAD)
C     *************************************************************
C     *                                                           *
C     * THE TOTAL STIFFNESS MATRIX & LOAD VECTOR OF THE STRUCTURE *
C     *                                                           *
C     *************************************************************
      DIMENSION TSTIF(600,600),PLOAD(600),ESTIF(80,80),E13(80,80)
      DIMENSION PELEM(160),P2(80),PP(80,1),PP1(80),PP2(80,1)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100)
      NPOIN=(NSTRP+1)*(NSECT+1)
      N2=NITEM-6
      DO 10 I=1,3*NPOIN
      DO 10 J=1,3*NPOIN
10    TSTIF(I,J)=0.0
      DO 15 I=1,3*NPOIN
```

```fortran
15    PLOAD(I)=0.0
      IF(NPPOIN.GE.1) THEN
      DO 18 I=1,NPPOIN
      J=PPOIN(I,2)
18    PLOAD(J)=PPOIN(I,1)
      ELSE
      END IF
      DO 58 ISTRP=1,NSTRP
      CALL STIFE(ESTIF,ISTRP,E13)
      DO 20 I1=1,NSECT+1
      DO 20 II=1,6
      IE=6*(I1-1)+II
      IT=3*(NSTRP+1)*(I1-1)+3*(ISTRP-1)+II
      DO 20 J1=1,2
      DO 20 JJ=1,3*(NSECT+1)
      JE=3*(NSECT+1)*(J1-1)+JJ
      JT=3*(NSECT+1)*(ISTRP+J1-2)+JJ
      TSTIF(IT,JT)=TSTIF(IT,JT)+ESTIF(IE,JE)
20    CONTINUE
      IF(QLOAD.NE.0.0) THEN
      CALL LOADP(PELEM,ISTRP)
      DO 40 K=1,NSECT+1
      DO 30 I=1,6
30    PP1(I+6*(K-1))=PELEM(I+NITEM*(K-1))
      DO 35 I=1,N2
35    PP2(I+N2*(K-1),1)=PELEM(I+6+NITEM*(K-1))
40    CONTINUE
      CALL MATRM(PP,E13,PP2,80,80,1,6*(NSECT+1),N2*(NSECT+1),1)
      DO 50 K=1,6*(NSECT+1)
50    P2(K)=PP1(K)-PP(K,1)
      DO 55 I1=1,NSECT+1
      DO 55 II=1,6
      IE=6*(I1-1)+II
      IT=3*(NSTRP+1)*(I1-1)+3*(ISTRP-1)+II
      PLOAD(IT)=PLOAD(IT)+P2(IE)
55    CONTINUE
      ELSE
      END IF
58    CONTINUE
      RETURN
      END

      SUBROUTINE STIFE(ESTIF,ISTRP,E13)
C     ***************************************************************
C     *                                                             *
C     *           THE STIFFNESS MATRIX OF THE ITH STRIP             *
C     *                                                             *
C     ***************************************************************
      DIMENSION ESTIF1(160,160),ESTIF2(80,80),ESTIF(80,80)
      DIMENSION ESTI1(80,80),ESTI2(80,80),ESTI3(80,80),E13(80,80)
      DIMENSION PHETA(12,160),PHET1(12,160),E132(80,80)
      DIMENSION PHET2(12,160),D2(12,160),D1(12,160),D0(12,160)
      DIMENSION ABARM(12,12),BBARM(12,12),CBARM(12,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &           NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100)
      DO 5 I=1,NITEM*(NSECT+1)
      DO 5 J=1,NITEM*(NSECT+1)
5     ESTIF1(I,J)=0.0
      H=2.0/NSECT
      N1=NITEM/3
```

```
      N2=NITEM-6                                                        KAN01810
      DO 50 IPOIN=1,NSECT+1                                             KAN01820
      GY=-1.0+(IPOIN-1)*H                                               KAN01830
      CALL MAABC(ABARM,BBARM,CBARM,ISTRP,GY,H)                          KAN01840
      DO 10 I=1,NITEM                                                   KAN01850
      DO 10 J=1,NITEM*(NSECT+1)                                         KAN01860
      PHETA(I,J)=0.0                                                    KAN01870
      PHET1(I,J)=0.0                                                    KAN01880
10    PHET2(I,J)=0.0                                                    KAN01890
      IF(IPOIN.EQ.1) THEN                                               KAN01900
      IF(ISUPT.EQ.1) THEN                                               KAN01910
      DO 12 K=1,N1                                                      KAN01920
      DO 11 J=1,2                                                       KAN01930
      PHET1(3*(K-1)+J,1+(3*K+J-4)*(NSECT+1))=NSECT                      KAN01940
      PHET2(3*(K-1)+J,1+(3*K+J-4)*(NSECT+1))=-6.0/H**2                  KAN01950
      PHET2(3*(K-1)+J,2+(3*K+J-4)*(NSECT+1))=3.0/H**2                   KAN01960
11    CONTINUE                                                          KAN01970
      PHETA(3*K,1+(3*K-1)*(NSECT+1))=2.0/3.0                            KAN01980
      PHET2(3*K,1+(3*K-1)*(NSECT+1))=-2.0/H**2                          KAN01990
      PHET2(3*K,2+(3*K-1)*(NSECT+1))=3.0/H**2                           KAN02000
12    CONTINUE                                                          KAN02010
      ELSE                                                              KAN02020
      DO 14 K=1,N1                                                      KAN02030
      PHET2(3*K-2,2+3*(K-1)*(NSECT+1))=3.0/H**2                         KAN02040
      DO 13 J=2,3                                                       KAN02050
      PHET1(3*(K-1)+J,1+(3*K+J-4)*(NSECT+1))=NSECT                      KAN02060
      PHET2(3*(K-1)+J,1+(3*K+J-4)*(NSECT+1))=-6.0/H**2                  KAN02070
      PHET2(3*(K-1)+J,2+(3*K+J-4)*(NSECT+1))=3.0/H**2                   KAN02080
13    CONTINUE                                                          KAN02090
14    CONTINUE                                                          KAN02100
      END IF                                                            KAN02110
      ELSE IF(IPOIN.EQ.2) THEN                                          KAN02120
      IF(ISUPT.EQ.1) THEN                                               KAN02130
      DO 15 K=1,NITEM                                                   KAN02140
      PHETA(K,1+(K-1)*(NSECT+1))=1.0/6.0                                KAN02150
      PHETA(K,2+(K-1)*(NSECT+1))=7.0/12.0                               KAN02160
      PHETA(K,3+(K-1)*(NSECT+1))=1.0/6.0                                KAN02170
      PHET1(K,1+(K-1)*(NSECT+1))=-NSECT/4.0                             KAN02180
      PHET1(K,2+(K-1)*(NSECT+1))=NSECT/8.0                              KAN02190
      PHET1(K,3+(K-1)*(NSECT+1))=NSECT/4.0                              KAN02200
      PHET2(K,1+(K-1)*(NSECT+1))=1.0/H**2                               KAN02210
      PHET2(K,2+(K-1)*(NSECT+1))=-2.5/H**2                              KAN02220
      PHET2(K,3+(K-1)*(NSECT+1))=1.0/H**2                               KAN02230
15    CONTINUE                                                          KAN02240
      ELSE                                                              KAN02250
      DO 17 K=1,N1                                                      KAN02260
      PHETA(3*K-2,2+3*(K-1)*(NSECT+1))=7.0/12.0                         KAN02270
      PHETA(3*K-2,3+3*(K-1)*(NSECT+1))=1.0/6.0                          KAN02280
      PHET1(3*K-2,2+3*(K-1)*(NSECT+1))=NSECT/8.0                        KAN02290
      PHET1(3*K-2,3+3*(K-1)*(NSECT+1))=NSECT/4.0                        KAN02300
      PHET2(3*K-2,2+3*(K-1)*(NSECT+1))=-2.5/H**2                        KAN02310
      PHET2(3*K-2,3+3*(K-1)*(NSECT+1))=1.0/H**2                         KAN02320
      DO 16 J=2,3                                                       KAN02330
      PHETA(3*(K-1)+J,1+(3*K+J-4)*(NSECT+1))=1.0/6.0                    KAN02340
      PHETA(3*(K-1)+J,2+(3*K+J-4)*(NSECT+1))=7.0/12.0                   KAN02350
      PHETA(3*(K-1)+J,3+(3*K+J-4)*(NSECT+1))=1.0/6.0                    KAN02360
      PHET1(3*(K-1)+J,1+(3*K+J-4)*(NSECT+1))=-NSECT/4.0                 KAN02370
      PHET1(3*(K-1)+J,2+(3*K+J-4)*(NSECT+1))=NSECT/8.0                  KAN02380
      PHET1(3*(K-1)+J,3+(3*K+J-4)*(NSECT+1))=NSECT/4.0                  KAN02390
      PHET2(3*(K-1)+J,1+(3*K+J-4)*(NSECT+1))=1.0/H**2                   KAN02400
```

```
      PHET2(3*(K-1)+J,2+(3*K+J-4)*(NSECT+1))=-2.5/H**2
      PHET2(3*(K-1)+J,3+(3*K+J-4)*(NSECT+1))=1.0/H**2
16    CONTINUE
17    CONTINUE
      END IF
      ELSE IF(IPOIN.EQ.NSECT) THEN
      IF(ISUPT.EQ.1) THEN
      DO 18 K=1,NITEM
      PHETA(K,NSECT-1+(K-1)*(NSECT+1))=1.0/6.0
      PHETA(K,NSECT+(K-1)*(NSECT+1))=7.0/12.0
      PHETA(K,NSECT+1+(K-1)*(NSECT+1))=1.0/6.0
      PHET1(K,NSECT-1+(K-1)*(NSECT+1))=-NSECT/4.0
      PHET1(K,NSECT+(K-1)*(NSECT+1))=-NSECT/8.0
      PHET1(K,NSECT+1+(K-1)*(NSECT+1))=NSECT/4.0
      PHET2(K,NSECT-1+(K-1)*(NSECT+1))=1.0/H**2
      PHET2(K,NSECT+(K-1)*(NSECT+1))=-2.5/H**2
      PHET2(K,NSECT+1+(K-1)*(NSECT+1))=1.0/H**2
18    CONTINUE
      ELSE
      DO 20 K=1,N1
      PHETA(3*K-2,NSECT-1+3*(K-1)*(NSECT+1))=1.0/6.0
      PHETA(3*K-2,NSECT+3*(K-1)*(NSECT+1))=7.0/12.0
      PHET1(3*K-2,NSECT-1+3*(K-1)*(NSECT+1))=-NSECT/4.0
      PHET1(3*K-2,NSECT+3*(K-1)*(NSECT+1))=-NSECT/8.0
      PHET2(3*K-2,NSECT-1+3*(K-1)*(NSECT+1))=1.0/H**2
      PHET2(3*K-2,NSECT+3*(K-1)*(NSECT+1))=-2.5/H**2
      DO 19 J=2,3
      PHETA(3*(K-1)+J,NSECT-1+(3*K+J-4)*(NSECT+1))=1.0/6.0
      PHETA(3*(K-1)+J,NSECT+(3*K+J-4)*(NSECT+1))=7.0/12.0
      PHETA(3*(K-1)+J,NSECT+1+(3*K+J-4)*(NSECT+1))=1.0/6.0
      PHET1(3*(K-1)+J,NSECT-1+(3*K+J-4)*(NSECT+1))=-NSECT/4.0
      PHET1(3*(K-1)+J,NSECT+(3*K+J-4)*(NSECT+1))=-NSECT/8.0
      PHET1(3*(K-1)+J,NSECT+1+(3*K+J-4)*(NSECT+1))=NSECT/4.0
      PHET2(3*(K-1)+J,NSECT-1+(3*K+J-4)*(NSECT+1))=1.0/H**2
      PHET2(3*(K-1)+J,NSECT+(3*K+J-4)*(NSECT+1))=-2.5/H**2
      PHET2(3*(K-1)+J,NSECT+1+(3*K+J-4)*(NSECT+1))=1.0/H**2
19    CONTINUE
20    CONTINUE
      END IF
      ELSE IF(IPOIN.EQ.NSECT+1) THEN
      IF(ISUPT.EQ.1) THEN
      DO 22 K=1,N1
      DO 21 J=1,2
      PHET1(3*(K-1)+J,NSECT+1+(3*K+J-4)*(NSECT+1))=-NSECT
      PHET2(3*(K-1)+J,NSECT+(3*K+J-4)*(NSECT+1))=3.0/H**2
      PHET2(3*(K-1)+J,NSECT+1+(3*K+J-4)*(NSECT+1))=-6.0/H**2
21    CONTINUE
      PHETA(3*K,NSECT+1+(3*K-1)*(NSECT+1))=2.0/3.0
      PHET2(3*K,NSECT+(3*K-1)*(NSECT+1))=3.0/H**2
      PHET2(3*K,NSECT+1+(3*K-1)*(NSECT+1))=-2.0/H**2
22    CONTINUE
      ELSE
      DO 24 K=1,N1
      PHET2(3*K-2,NSECT+3*(K-1)*(NSECT+1))=3.0/H**2
      DO 23 J=2,3
      PHET1(3*(K-1)+J,NSECT+1+(3*K+J-4)*(NSECT+1))=-NSECT
      PHET2(3*(K-1)+J,NSECT+(3*K+J-4)*(NSECT+1))=3.0/H**2
      PHET2(3*(K-1)+J,NSECT+1+(3*K+J-4)*(NSECT+1))=-6.0/H**2
23    CONTINUE
24    CONTINUE
```

```
      END IF
      ELSE
      DO 30 K=1,NITEM
      PHETA(K,IPOIN-1+(K-1)*(NSECT+1))=1.0/6.0
      PHETA(K,IPOIN+(K-1)*(NSECT+1))=2.0/3.0
      PHETA(K,IPOIN+1+(K-1)*(NSECT+1))=1.0/6.0
      PHET1(K,IPOIN-1+(K-1)*(NSECT+1))=-NSECT/4.0
      PHET1(K,IPOIN+1+(K-1)*(NSECT+1))=NSECT/4.0
      PHET2(K,IPOIN-1+(K-1)*(NSECT+1))=1.0/H**2
      PHET2(K,IPOIN+(K-1)*(NSECT+1))=-2.0/H**2
      PHET2(K,IPOIN+1+(K-1)*(NSECT+1))=1.0/H**2
30    CONTINUE
      END IF
      CALL MATRM(D2,ABARM,PHET2,12,12,160,NITEM,NITEM,NITEM*(NSECT+1))
      CALL MATRM(D1,BBARM,PHET1,12,12,160,NITEM,NITEM,NITEM*(NSECT+1))
      CALL MATRM(D0,CBARM,PHETA,12,12,160,NITEM,NITEM,NITEM*(NSECT+1))
      DO 40 I=1,NITEM
      DO 40 J=1,NITEM*(NSECT+1)
40    ESTIF1(I+(IPOIN-1)*NITEM,J)=D2(I,J)+D1(I,J)+D0(I,J)
50    CONTINUE
      IF(ISUPT.EQ.2) THEN
      DO 60 K1=1,N1
      DO 55 K2=1,NITEM*(NSECT+1)
      ESTIF1(3*K1-2,K2)=0.0
      ESTIF1(3*K1-2+NSECT*NITEM,K2)=0.0
      ESTIF1(K2,1+3*(K1-1)*(NSECT+1))=0.0
      ESTIF1(K2,NSECT+1+3*(K1-1)*(NSECT+1))=0.0
55    CONTINUE
      ESTIF1(3*K1-2,1+3*(K1-1)*(NSECT+1))=1.0
      ESTIF1(3*K1-2+NSECT*NITEM,NSECT+1+3*(K1-1)*(NSECT+1))=1.0
60    CONTINUE
      ELSE
      END IF
      DO 61 K1=1,NSECT+1
      DO 61 I=1,6
      DO 61 J=1,6*(NSECT+1)
61    ESTIF2(I+6*(K1-1),J)=ESTIF1(I+NITEM*(K1-1),J)
      DO 62 K1=1,NSECT+1
      DO 62 I=1,6
      DO 62 J=1,N2*(NSECT+1)
62    ESTI1(I+6*(K1-1),J)=ESTIF1(I+NITEM*(K1-1),J+6*(NSECT+1))
      DO 64 J=1,6*(NSECT+1)
      DO 64 K2=1,NSECT+1
      DO 64 I=1,N2
64    ESTI2(I+N2*(K2-1),J)=ESTIF1(I+6+NITEM*(K2-1),J)
      DO 66 K1=1,NSECT+1
      DO 66 I=1,N2
      DO 66 J=1,N2*(NSECT+1)
66    ESTI3(I+N2*(K1-1),J)=ESTIF1
     &        (I+6+NITEM*(K1-1),J+6*(NSECT+1))
      CALL MATI3(ESTI3,N2*(NSECT+1))
      CALL MATRM(E13,ESTI1,ESTI3,80,80,80,
     &              6*(NSECT+1),N2*(NSECT+1),N2*(NSECT+1))
      CALL MATRM(E132,E13,ESTI2,80,80,80,6*(NSECT+1),
     &              N2*(NSECT+1),6*(NSECT+1))
      DO 70 I=1,6*(NSECT+1)
      DO 70 J=1,6*(NSECT+1)
      ESTIF(I,J)=ESTIF2(I,J)-E132(I,J)
70    CONTINUE
      RETURN
```

```
      END

      SUBROUTINE LOADP(PELEM,ISTRP)
C     ******************************************************************
C     *                                                                *
C     *       EQUIVELENT NODAL LOAD VECTOR OF THE ITH STRIP            *
C     *                                                                *
C     ******************************************************************
      DIMENSION P0(12),PELEM(160),PX(3),WEIGH(3)
      DIMENSION SHAPN(3,12),SHADN(3,12)
      DIMENSION XJACO(2,2),XTR2Y(2),XTRXY(2)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 10 I=1,NITEM*(NSECT+1)
 10   PELEM(I)=0.0
      H=2.0/NSECT
      N1=NITEM/3
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 50 IPOIN=1,NSECT+1
      DO 20 I=1,NITEM
 20   P0(I)=0.0
      GY=-1.0+(IPOIN-1.0)*H
      DO 30 IGUSS=1,3
      GX=PX(IGUSS)
      CALL SHAPE(SHAPN,SHADN,ISTRP,GX)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 25 K=1,N1
 25   P0(3*K-2)=P0(3*K-2)+QLOAD*SHAPN(1,3*K-2)*DETJ*WEIGH(IGUSS)/NSTRP
 30   CONTINUE
      DO 40 K=1,NITEM
 40   PELEM(K+NITEM*(IPOIN-1))=PELEM(K+NITEM*(IPOIN-1))+P0(K)
 50   CONTINUE
      DO 60 I=1,NITEM
      IF(ISUPT.EQ.1.AND.I.EQ.3) THEN
      ELSE IF(ISUPT.EQ.1.AND.I.EQ.6) THEN
      ELSE IF(ISUPT.EQ.1.AND.I.EQ.9) THEN
      ELSE IF(ISUPT.EQ.1.AND.I.EQ.12) THEN
      ELSE
      PELEM(I)=0.0
      PELEM(I+NITEM*NSECT)=0.0
      END IF
 60   CONTINUE
      RETURN
      END

      SUBROUTINE MAABC(AHATM,BHATM,CHATM,ISTRP,GY,H)
C     ******************************************************************
C     *                                                                *
C     *                      TRANSITION MATRIX                         *
C     *                                                                *
C     ******************************************************************
      DIMENSION ABEMA(12,12),BBEMA(12,12),CBEMA(12,12)
      DIMENSION ASEMA(12,12),BSEMA(12,12),CSEMA(12,12)
      DIMENSION ABEPM(12,12),BBEPM(12,12)
      DIMENSION ASEPM(12,12),BSEPM(12,12)
      DIMENSION AEMAT(12,12),BEMAT(12,12),CEMAT(12,12)
      DIMENSION AEPMA(12,12),BEPMA(12,12)
```

```
      DIMENSION ABARM(12,12),BBARM(12,12),CBARM(12,12)
      DIMENSION AHATM(12,12),BHATM(12,12),CHATM(12,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      CALL MAABE(ABEMA,ISTRP,GY)
      CALL MABBE(BBEMA,ISTRP,GY)
      CALL MACBE(CBEMA,ISTRP,GY)
      CALL MAASE(ASEMA,ISTRP,GY)
      CALL MABSE(BSEMA,ISTRP,GY)
      CALL MACSE(CSEMA,ISTRP,GY)
      CALL MABEP(ABEPM,ISTRP,GY)
      CALL MBBEP(BBEPM,ISTRP,GY)
      CALL MASEP(ASEPM,ISTRP,GY)
      CALL MBSEP(BSEPM,ISTRP,GY)
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
      AEMAT(I,J)=ABEMA(I,J)+ASEMA(I,J)
      BEMAT(I,J)=BBEMA(I,J)+BSEMA(I,J)
      AEPMA(I,J)=ABEPM(I,J)+ASEPM(I,J)
      BEPMA(I,J)=BBEPM(I,J)+BSEPM(I,J)
10    CEMAT(I,J)=CBEMA(I,J)+CSEMA(I,J)
      DO 20 I=1,NITEM
      DO 20 J=1,NITEM
      ABARM(I,J)=AEMAT(I,J)
      BBARM(I,J)=AEPMA(I,J)+BEMAT(J,I)-BEMAT(I,J)
20    CBARM(I,J)=BEPMA(J,I)-CEMAT(I,J)
      DO 30 I=1,NITEM
      DO 30 J=1,NITEM
      AHATM(I,J)=ABARM(I,J)/(H*H)-BBARM(I,J)/(2.0*H)
      BHATM(I,J)=-2.0*ABARM(I,J)/(H*H)+CBARM(I,J)
30    CHATM(I,J)=ABARM(I,J)/(H*H)+BBARM(I,J)/(2.0*H)
      RETURN
      END

      SUBROUTINE MAABE(ABEMA,ISTRP,GY)
*********************************************************
*                                                       *
*                   TRANSITION MATRIX                   *
*                                                       *
*********************************************************
      DIMENSION ABEMA(12,12),PX(3),DBMAT(3,3)
      DIMENSION XJACO(2,2),AATR1(3,12),AATR2(12,12)
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,12),SD2MA(3,12)
      DIMENSION SB1MA(3,12),SB2MA(3,12),WEIGH(3)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
5     ABEMA(I,J)=0.0
      CALL MATDB(DBMAT)
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 20 IGUSS=1,3
      GX=PX(IGUSS)
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,12,3,3,NITEM)
      CALL MATRMT(AATR2,SB2MA,AATR1,3,12,12,3,NITEM,NITEM)
```

```
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
10    ABEMA(I,J)=ABEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
20    CONTINUE
      RETURN
      END

      SUBROUTINE MABEP(ABEPM,ISTRP,GY)
C     ****************************************************************
C     *                                                              *
C     *                      TRANSITION MATRIX                       *
C     *                                                              *
C     ****************************************************************
      DIMENSION ABEPM(12,12),PX(3),DBMAT(3,3)
      DIMENSION XJACO(2,2),AATR1(3,12),AATR2(12,12),AATR3(12,12)
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,12),SD2MA(3,12)
      DIMENSION SB1MA(3,12),SB2MA(3,12),WEIGH(3)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
5     ABEPM(I,J)=0.0
      CALL MATDB(DBMAT)
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
      DO 20 IGUSS=1,3
      GX=PX(IGUSS)
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,12,3,3,NITEM)
      CALL MATRMT(AATR2,SD2MA,AATR1,3,12,12,3,NITEM,NITEM)
      CALL MATRMT(AATR3,SB2MA,AATR1,3,12,12,3,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 15 I=1,NITEM
      DO 15 J=1,NITEM
15    ABEPM(I,J)=ABEPM(I,J)+(2.0*AATR2(I,J)*DETJ+AATR3(I,J)*DJDY)
     &          *WEIGH(IGUSS)/NSTRP
20    CONTINUE
      RETURN
      END

      SUBROUTINE MABBE(BBEMA,ISTRP,GY)
C     ****************************************************************
C     *                                                              *
C     *                      TRANSITION MATRIX                       *
C     *                                                              *
C     ****************************************************************
      DIMENSION BBEMA(12,12),PX(3),DBMAT(3,3)
      DIMENSION XJACO(2,2),AATR1(3,12),AATR2(12,12)
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,12),SD2MA(3,12)
      DIMENSION SB1MA(3,12),SB2MA(3,12),WEIGH(3)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      DATA WEIGH/0.555556,0.888889,0.555556/
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
```

```
 5    BBEMA(I,J)=0.0                                                    KAN05410
      CALL MATDB(DBMAT)                                                 KAN05420
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP                    KAN05430
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP                    KAN05440
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP                    KAN05450
      DO 20 IGUSS=1,3                                                   KAN05460
      GX=PX(IGUSS)                                                      KAN05470
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)                   KAN05480
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,12,3,3,NITEM)                    KAN05490
      CALL MATRMT(AATR2,SB1MA,AATR1,3,12,12,3,NITEM,NITEM)              KAN05500
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)                     KAN05510
      DO 10 I=1,NITEM                                                   KAN05520
      DO 10 J=1,NITEM                                                   KAN05530
10    BBEMA(I,J)=BBEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP          KAN05540
20    CONTINUE                                                          KAN05550
      RETURN                                                            KAN05560
      END                                                               KAN05570
                                                                        KAN05580
      SUBROUTINE MBBEP(BBEPM,ISTRP,GY)                                  KAN05590
C     ***************************************************************** KAN05600
C     *                                                               * KAN05610
C     *                        TRANSITION MATRIX                      * KAN05620
C     *                                                               * KAN05630
C     ***************************************************************** KAN05640
      DIMENSION BBEPM(12,12),PX(3),DBMAT(3,3)                           KAN05650
      DIMENSION XJACO(2,2),AATR1(3,12),AATR2(12,12)                     KAN05660
      DIMENSION AATR3(3,12),AATR4(12,12),AATR5(12,12)                   KAN05670
      DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,12),SD2MA(3,12)               KAN05680
      DIMENSION SB1MA(3,12),SB2MA(3,12),WEIGH(3)                        KAN05690
      COMMON/PELEC/EMODU,POISS,THICK                                    KAN05700
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,            KAN05710
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)               KAN05720
      DATA WEIGH/0.555556,0.888889,0.555556/                            KAN05730
      DO 5 I=1,NITEM                                                    KAN05740
      DO 5 J=1,NITEM                                                    KAN05750
 5    BBEPM(I,J)=0.0                                                    KAN05760
      CALL MATDB(DBMAT)                                                 KAN05770
      PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP                    KAN05780
      PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP                    KAN05790
      PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP                    KAN05800
      DO 20 IGUSS=1,3                                                   KAN05810
      GX=PX(IGUSS)                                                      KAN05820
      CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)                   KAN05830
      CALL MATRM(AATR1,DBMAT,SB2MA,3,3,12,3,3,NITEM)                    KAN05840
      CALL MATRMT(AATR2,SD1MA,AATR1,3,12,12,3,NITEM,NITEM)              KAN05850
      CALL MATRM(AATR3,DBMAT,SD2MA,3,3,12,3,3,NITEM)                    KAN05860
      CALL MATRMT(AATR4,SB1MA,AATR3,3,12,12,3,NITEM,NITEM)              KAN05870
      CALL MATRMT(AATR5,SB1MA,AATR1,3,12,12,3,NITEM,NITEM)              KAN05880
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)                     KAN05890
      DO 15 I=1,NITEM                                                   KAN05900
      DO 15 J=1,NITEM                                                   KAN05910
15    BBEPM(I,J)=BBEPM(I,J)+((AATR2(I,J)+AATR4(I,J))*DETJ+AATR5(I,J)    KAN05920
     &          *DJDY)*WEIGH(IGUSS)/NSTRP                               KAN05930
20    CONTINUE                                                          KAN05940
      RETURN                                                            KAN05950
      END                                                               KAN05960
                                                                        KAN05970
      SUBROUTINE MACBE(CBEMA,ISTRP,GY)                                  KAN05980
C     ***************************************************************** KAN05990
C     *                                                               * KAN06000
```

```
C      *                      TRANSITION MATRIX                       *
C      *                                                              *
C      ****************************************************************
       DIMENSION CBEMA(12,12),DBMAT(3,3),PX(3)
       DIMENSION XJACO(2,2),AATR1(3,12),AATR2(12,12)
       DIMENSION XTRXY(2),XTR2Y(2),SD1MA(3,12),SD2MA(3,12)
       DIMENSION SB1MA(3,12),SB2MA(3,12),WEIGH(3)
       COMMON/PELEC/EMODU,POISS,THICK
       COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
       DATA WEIGH/0.555556,0.888889,0.555556/
       DO 5 I=1,NITEM
       DO 5 J=1,NITEM
 5     CBEMA(I,J)=0.0
       CALL MATDB(DBMAT)
       PX(1)=-1.0+2.0*(ISTRP-1.0)/NSTRP+0.22540/NSTRP
       PX(2)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.00000/NSTRP
       PX(3)=-1.0+2.0*(ISTRP-1.0)/NSTRP+1.77460/NSTRP
       DO 20 IGUSS=1,3
       GX=PX(IGUSS)
       CALL MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
       CALL MATRM(AATR1,DBMAT,SB1MA,3,3,12,3,3,NITEM)
       CALL MATRMT(AATR2,SB1MA,AATR1,3,12,12,3,NITEM,NITEM)
       CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
       DO 10 I=1,NITEM
       DO 10 J=1,NITEM
 10    CBEMA(I,J)=CBEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
 20    CONTINUE
       RETURN
       END

       SUBROUTINE MAASE(ASEMA,ISTRP,GY)
C      ****************************************************************
C      *                                                              *
C      *                      TRANSITION MATRIX                       *
C      *                                                              *
C      ****************************************************************
       DIMENSION ASEMA(12,12),PX(3),XJACO(2,2),XTRXY(2),XTR2Y(2)
       DIMENSION SS1MA(2,12),SS2MA(2,12),WEIGH(3)
       DIMENSION DSMAT(2,2)
       DIMENSION AATR1(2,12),AATR2(12,12),DS1MA(2,12),DS2MA(2,12)
       COMMON/PELEC/EMODU,POISS,THICK
       COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
       DO 5 I=1,NITEM
       DO 5 J=1,NITEM
 5     ASEMA(I,J)=0.0
       CALL MATDS(DSMAT)
       CALL GAUSS(PX,WEIGH,ISTRP)
       DO 20 IGUSS=1,NGUSS
       GX=PX(IGUSS)
       CALL MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
       CALL MATRM(AATR1,DSMAT,SS2MA,2,2,12,2,2,NITEM)
       CALL MATRMT(AATR2,SS2MA,AATR1,2,12,12,2,NITEM,NITEM)
       CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
       DO 10 I=1,NITEM
       DO 10 J=1,NITEM
 10    ASEMA(I,J)=ASEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
 20    CONTINUE
       RETURN
```

```
      END

      SUBROUTINE MASEP(ASEPM,ISTRP,GY)
C     ******************************************************************
C     *                                                                *
C     *                    TRANSITION MATRIX                           *
C     *                                                                *
C     ******************************************************************
      DIMENSION ASEPM(12,12),PX(3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      DIMENSION SS1MA(2,12),SS2MA(2,12),WEIGH(3)
      DIMENSION DSMAT(2,2),AATR3(12,12)
      DIMENSION AATR1(2,12),AATR2(12,12),DS1MA(2,12),DS2MA(2,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
 5    ASEPM(I,J)=0.0
      CALL MATDS(DSMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 20 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DSMAT,SS2MA,2,2,12,2,2,NITEM)
      CALL MATRMT(AATR2,DS2MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL MATRMT(AATR3,SS2MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 15 I=1,NITEM
      DO 15 J=1,NITEM
      ASEPM(I,J)=ASEPM(I,J)+(2.0*AATR2(I,J)*DETJ+AATR3(I,J)*DJDY)
     &         *WEIGH(IGUSS)/NSTRP
 15   CONTINUE
 20   CONTINUE
      RETURN
      END

      SUBROUTINE MABSE(BSEMA,ISTRP,GY)
C     ******************************************************************
C     *                                                                *
C     *                    TRANSITION MATRIX                           *
C     *                                                                *
C     ******************************************************************
      DIMENSION BSEMA(12,12),PX(3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      DIMENSION SS1MA(2,12),SS2MA(2,12),WEIGH(3)
      DIMENSION DSMAT(2,2)
      DIMENSION AATR1(2,12),AATR2(12,12),DS1MA(2,12),DS2MA(2,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      COMMON/PELEC/EMODU,POISS,THICK
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
 5    BSEMA(I,J)=0.0
      CALL MATDS(DSMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 20 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DSMAT,SS2MA,2,2,12,2,2,NITEM)
      CALL MATRMT(AATR2,SS1MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
```

```
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
10    BSEMA(I,J)=BSEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
20    CONTINUE
      RETURN
      END

      SUBROUTINE MBSEP(BSEPM,ISTRP,GY)
C     **********************************************************
C     *                                                        *
C     *                  TRANSITION MATRIX                     *
C     *                                                        *
C     **********************************************************
      DIMENSION BSEPM(12,12),PX(3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      DIMENSION SS1MA(2,12),SS2MA(2,12),WEIGH(3)
      DIMENSION DSMAT(2,2),AATR3(2,12),AATR4(12,12),AATR5(12,12)
      DIMENSION AATR1(2,12),AATR2(12,12),DS1MA(2,12),DS2MA(2,12)
      COMMON/PELEC/EMODU,POISS,THICK
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
5     BSEPM(I,J)=0.0
      CALL MATDS(DSMAT)
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 20 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DSMAT,SS2MA,2,2,12,2,2,NITEM)
      CALL MATRMT(AATR2,DS1MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL MATRM(AATR3,DSMAT,DS2MA,2,2,12,2,2,NITEM)
      CALL MATRMT(AATR4,SS1MA,AATR3,2,12,12,2,NITEM,NITEM)
      CALL MATRMT(AATR5,SS1MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 15 I=1,NITEM
      DO 15 J=1,NITEM
      BSEPM(I,J)=BSEPM(I,J)+((AATR2(I,J)+AATR4(I,J))*DETJ
     &          +AATR5(I,J)*DJDY)*WEIGH(IGUSS)/NSTRP
15    CONTINUE
20    CONTINUE
      RETURN
      END

      SUBROUTINE MACSE(CSEMA,ISTRP,GY)
C     **********************************************************
C     *                                                        *
C     *                  TRANSITION MATRIX                     *
C     *                                                        *
C     **********************************************************
      DIMENSION CSEMA(12,12),PX(3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      DIMENSION SS1MA(2,12),SS2MA(2,12),WEIGH(3)
      DIMENSION DSMAT(2,2)
      DIMENSION AATR1(2,12),AATR2(12,12),DS1MA(2,12),DS2MA(2,12)
      COMMON/PELEC/EMODU,POISS,THICK
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 5 I=1,NITEM
      DO 5 J=1,NITEM
5     CSEMA(I,J)=0.0
      CALL MATDS(DSMAT)
```

```
      CALL GAUSS(PX,WEIGH,ISTRP)
      DO 20 IGUSS=1,NGUSS
      GX=PX(IGUSS)
      CALL MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
      CALL MATRM(AATR1,DSMAT,SS1MA,2,2,12,2,2,NITEM)
      CALL MATRMT(AATR2,SS1MA,AATR1,2,12,12,2,NITEM,NITEM)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,NITEM
      DO 10 J=1,NITEM
10    CSEMA(I,J)=CSEMA(I,J)+AATR2(I,J)*DETJ*WEIGH(IGUSS)/NSTRP
20    CONTINUE
      RETURN
      END

      SUBROUTINE MATSB(SB1MA,SB2MA,SD1MA,SD2MA,ISTRP,GX,GY)
C     **********************************************************
C     *                                                        *
C     *                 BENDING S MATRIX                       *
C     *                                                        *
C     **********************************************************
      DIMENSION SB1MA(3,12),SD1MA(3,12),SD2MA(3,12)
      DIMENSION TRLB1(3,3),TRLB2(3,3),TLB1D(3,3),TLB2D(3,3)
      DIMENSION SHAPN(3,12),SHADN(3,12),SB2MA(3,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 5 I=1,3
      DO 5 J=1,NITEM
      SB1MA(I,J)=0.0
      SB2MA(I,J)=0.0
      SD1MA(I,J)=0.0
5     SD2MA(I,J)=0.0
      CALL MATLB(TRLB1,TRLB2,GX,GY)
      CALL MALBD(TLB1D,TLB2D,GX,GY)
      CALL SHAPE(SHAPN,SHADN,ISTRP,GX)
      CALL MATRM(SB1MA,TRLB1,SHADN,3,3,12,3,3,NITEM)
      CALL MATRM(SB2MA,TRLB2,SHAPN,3,3,12,3,3,NITEM)
      CALL MATRM(SD1MA,TLB1D,SHADN,3,3,12,3,3,NITEM)
      CALL MATRM(SD2MA,TLB2D,SHAPN,3,3,12,3,3,NITEM)
      RETURN
      END

      SUBROUTINE MATSS(SS1MA,SS2MA,DS1MA,DS2MA,ISTRP,GX,GY)
C     **********************************************************
C     *                                                        *
C     *                  SHEAR S MATRIX                        *
C     *                                                        *
C     **********************************************************
      DIMENSION SS1MA(2,12),SS2MA(2,12),DS1MA(2,12)
      DIMENSION TRLS0(2,3),TRLS1(2,3),TRLS2(2,3),TLS1D(2,3),TLS2D(2,3)
      DIMENSION SHAPN(3,12),SHADN(3,12),DS2MA(2,12)
      DIMENSION AATR1(2,12),AATR2(2,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DO 5 I=1,2
      DO 5 J=1,NITEM
      SS1MA(I,J)=0.0
      SS2MA(I,J)=0.0
      DS1MA(I,J)=0.0
5     DS2MA(I,J)=0.0
      CALL MATLS(TRLS0,TRLS1,TRLS2,GX,GY)
```

```
      CALL MALSD(TLS1D,TLS2D,GX,GY)
      CALL SHAPE(SHAPN,SHADN,ISTRP,GX)
      CALL MATRM(AATR1,TRLS0,SHAPN,2,3,12,2,3,NITEM)
      CALL MATRM(AATR2,TRLS1,SHADN,2,3,12,2,3,NITEM)
      DO 10 I=1,2
      DO 10 J=1,NITEM
10    SS1MA(I,J)=AATR1(I,J)+AATR2(I,J)
      CALL MATRM(SS2MA,TRLS2,SHAPN,2,3,12,2,3,NITEM)
      CALL MATRM(DS1MA,TLS1D,SHADN,2,3,12,2,3,NITEM)
      CALL MATRM(DS2MA,TLS2D,SHAPN,2,3,12,2,3,NITEM)
      RETURN
      END

      SUBROUTINE MATLB(TRLB1,TRLB2,GX,GY)
C     ****************************************************************
C     *                                                              *
C     *     OPERATOR TRANSFORMATION MATRIX FOR BENDING STRAIN        *
C     *                                                              *
C     ****************************************************************
      DIMENSION TRLB1(3,3),TRLB2(3,3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,3
      DO 10 J=1,3
      TRLB1(I,J)=0.0
10    TRLB2(I,J)=0.0
      TRLB1(1,2)=-XJACO(2,2)/DETJ
      TRLB1(2,3)=XJACO(2,1)/DETJ
      TRLB1(3,2)=XJACO(2,1)/DETJ
      TRLB1(3,3)=-XJACO(2,2)/DETJ
      TRLB2(1,2)=XJACO(1,2)/DETJ
      TRLB2(2,3)=-XJACO(1,1)/DETJ
      TRLB2(3,2)=-XJACO(1,1)/DETJ
      TRLB2(3,3)=XJACO(1,2)/DETJ
      RETURN
      END

      SUBROUTINE MALBD(TLB1D,TLB2D,GX,GY)
C     ****************************************************************
C     *                                                              *
C     *     OPERATOR TRANSFORMATION MATRIX FOR BENDING STRAIN        *
C     *                                                              *
C     ****************************************************************
      DIMENSION TLB1D(3,3),TLB2D(3,3),XJACO(2,2),XTRXY(2),XTR2Y(2)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,3
      DO 10 J=1,3
      TLB1D(I,J)=0.0
10    TLB2D(I,J)=0.0
      TLB1D(1,2)=-XTR2Y(2)/DETJ+DJDY*XJACO(2,2)/(DETJ**2)
      TLB1D(2,3)=XTR2Y(1)/DETJ-DJDY*XJACO(2,1)/(DETJ**2)
      TLB1D(3,2)=TLB1D(2,3)
      TLB1D(3,3)=TLB1D(1,2)
      TLB2D(1,2)=XTRXY(2)/DETJ-DJDY*XJACO(1,2)/(DETJ**2)
      TLB2D(2,3)=-XTRXY(1)/DETJ+DJDY*XJACO(1,1)/(DETJ**2)
      TLB2D(3,2)=TLB2D(2,3)
      TLB2D(3,3)=TLB2D(1,2)
      RETURN
      END

      SUBROUTINE MATLS(TRLS0,TRLS1,TRLS2,GX,GY)
```

```
C     ****************************************************************
C     *                                                              *
C     *       OPERATOR TRANSFORMATION MATRIX FOR SHEAR STRAIN        *
C     *                                                              *
C     ****************************************************************
      DIMENSION TRLS0(2,3),TRLS1(2,3),TRLS2(2,3)
      DIMENSION XJACO(2,2),XTRXY(2),XTR2Y(2)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,2
      DO 10 J=1,3
      TRLS0(I,J)=0.0
      TRLS1(I,J)=0.0
 10   TRLS2(I,J)=0.0
      TRLS0(1,2)=-1.0
      TRLS0(2,3)=-1.0
      TRLS1(1,1)=XJACO(2,2)/DETJ
      TRLS1(2,1)=-XJACO(2,1)/DETJ
      TRLS2(1,1)=-XJACO(1,2)/DETJ
      TRLS2(2,1)=XJACO(1,1)/DETJ
      RETURN
      END

      SUBROUTINE MALSD(TLS1D,TLS2D,GX,GY)
C     ****************************************************************
C     *                                                              *
C     *       OPERATOR TRANSFORMATION MATRIX FOR SHEAR STRAIN        *
C     *                                                              *
C     ****************************************************************
      DIMENSION TLS1D(2,3),TLS2D(2,3)
      DIMENSION XJACO(2,2),XTRXY(2),XTR2Y(2)
      CALL TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      DO 10 I=1,2
      DO 10 J=1,3
      TLS1D(I,J)=0.0
 10   TLS2D(I,J)=0.0
      TLS1D(1,1)=XTR2Y(2)/DETJ-DJDY*XJACO(2,2)/(DETJ**2)
      TLS1D(2,1)=-XTR2Y(1)/DETJ+DJDY*XJACO(2,1)/(DETJ**2)
      TLS2D(1,1)=-XTRXY(2)/DETJ+DJDY*XJACO(1,2)/(DETJ**2)
      TLS2D(2,1)=XTRXY(1)/DETJ-DJDY*XJACO(1,1)/(DETJ**2)
      RETURN
      END

      SUBROUTINE SHAPE(SHAPN,SHADN,ISTRP,GX)
C     ****************************************************************
C     *                                                              *
C     *         THE SHAPE FUNCTION FOR THE X-DIRECTION               *
C     *                                                              *
C     ****************************************************************
      DIMENSION SHA(4),SHA1(4),SHAPN(3,12),SHADN(3,12)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &            NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      N1=NITEM/3
      DO 10 I=1,3
      DO 10 J=1,NITEM
      SHAPN(I,J)=0.0
 10   SHADN(I,J)=0.0
      GXX=1.0-ISTRP+(GX+1.0)*NSTRP/2.0
      IF(N1.EQ.2) THEN
      SHA(1)=1.0-GXX
      SHA(2)=GXX
```

```
      SHA1(1)=-NSTRP/2.0
      SHA1(2)=NSTRP/2.0
      ELSE IF(N1.EQ.3) THEN
      SHA(1)=1.0-3.0*GXX+2.0*GXX**2
      SHA(2)=-GXX+2.0*GXX**2
      SHA(3)=4.0*GXX-4.0*GXX**2
      SHA1(1)=(-1.5+2.0*GXX)*NSTRP
      SHA1(2)=(-0.5+2.0*GXX)*NSTRP
      SHA1(3)=(2.0-4.0*GXX)*NSTRP
      ELSE IF(N1.EQ.4) THEN
      SHA(1)=1.0-5.5*GXX+9.0*GXX**2-4.5*GXX**3
      SHA(2)=GXX-4.5*GXX**2+4.5*GXX**3
      SHA(3)=9.0*GXX-22.5*GXX**2+13.5*GXX**3
      SHA(4)=-4.5*GXX+18.0*GXX**2-13.5*GXX**3
      SHA1(1)=(-2.75+9.0*GXX-6.75*GXX**2)*NSTRP
      SHA1(2)=(0.5-4.5*GXX+6.75*GXX**2)*NSTRP
      SHA1(3)=(4.5-22.5*GXX+20.25*GXX**2)*NSTRP
      SHA1(4)=(-2.25+18.0*GXX-20.25*GXX**2)*NSTRP
      ELSE
      END IF
      DO 20 I=1,N1
      SHAPN(1,3*I-2)=SHA(I)
      SHAPN(2,3*I-1)=SHA(I)
      SHAPN(3,3*I)=SHA(I)
      SHADN(1,3*I-2)=SHA1(I)
      SHADN(2,3*I-1)=SHA1(I)
      SHADN(3,3*I)=SHA1(I)
20    CONTINUE
      RETURN
      END

      SUBROUTINE TRANS(XJACO,XTRXY,XTR2Y,DETJ,DJDY,GX,GY)
      ***********************************************************
      *                                                         *
      *      THE COORDINATE TRANSFORMATION SUBROUTINE           *
      *                                                         *
      ***********************************************************
      DIMENSION XJACO(2,2),SHAP(2,8),PX(8),PY(8)
      DIMENSION TRXY(8),TR2Y(8),XTRXY(2),XTR2Y(2)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100,2)
      DATA PX/-1.0,-1.0,-1.0,-1.0,1.0,1.0,1.0,1.0/
      DATA PY/-1.0,-0.33333,0.33333,1.0,
     &        -1.0,-0.33333,0.33333,1.0/
      DO 20 I=1,8
      IF(I.EQ.1.OR.I.EQ.4.OR.I.EQ.5.OR.I.EQ.8) THEN
      SHAP(1,I)=PX(I)*(9.0*GY*GY-1.0)*(1.0+PY(I)*GY)/32.0
      SHAP(2,I)=(1.0+PX(I)*GX)*(18.0*GY+27.0*PY(I)*GY*GY-PY(I))/32.0
      TR2Y(I)=9.0*(1.0+GX*PX(I))*(1.0+3.0*GY*PY(I))/16.0
      TRXY(I)=PX(I)*(18.0*GY+27.0*GY*GY*PY(I)-PY(I))/32.0
      ELSE
      SHAP(1,I)=9.0*PX(I)*(1.0-GY*GY)*(1.0+9.0*PY(I)*GY)/32.0
      SHAP(2,I)=9.0*(1.0+PX(I)*GX)*(-2.0*GY-27.0*PY(I)*GY*GY
     &          +9.0*PY(I))/32.0
      TR2Y(I)=9.0*(1.0+GX*PX(I))*(-1.0-27.0*GY*PY(I))/16.0
      TRXY(I)=9.0*PX(I)*(-2.0*GY-27.0*GY*GY*PY(I)+9.0*PY(I))/32.0
      END IF
20    CONTINUE
      DO 40 I=1,2
      DO 40 J=1,2
```

```
      XJACO(I,J)=0.0
      DO 30 K=1,8
30    XJACO(I,J)=XJACO(I,J)+SHAP(I,K)*COOXQ(J,K)
40    CONTINUE
      DETJ=XJACO(1,1)*XJACO(2,2)-XJACO(1,2)*XJACO(2,1)
      DO 55 I=1,2
      XTR2Y(I)=0.0
      XTRXY(I)=0.0
      DO 50 K=1,8
      XTR2Y(I)=XTR2Y(I)+TR2Y(K)*COOXQ(I,K)
50    XTRXY(I)=XTRXY(I)+TRXY(K)*COOXQ(I,K)
55    CONTINUE
      DJDY=XJACO(1,1)*XTR2Y(2)+XJACO(2,2)*XTRXY(1)
     &     -XJACO(2,1)*XTRXY(2)-XJACO(1,2)*XTR2Y(1)
      RETURN
      END

      SUBROUTINE MATDS(DSMAT)
*************************************************************
*                                                           *
*        THE MATERIAL CONSTITUTIONAL MATRIX FOR SHEAR       *
*                                                           *
*************************************************************
      DIMENSION DSMAT(2,2)
      COMMON/PELEC/EMODU,POISS,THICK
      DO 10 I=1,2
      DO 10 J=1,2
10    DSMAT(I,J)=0.0
      GMODU=EMODU/(2.0*(1.0+POISS))
      DSMAT(1,1)=GMODU*THICK/1.2
      DSMAT(2,2)=GMODU*THICK/1.2
      RETURN
      END

      SUBROUTINE MATDB(DBMAT)
*************************************************************
*                                                           *
*        THE CONSTITUTION RELATIONSHIP FOR BENDING          *
*                                                           *
*************************************************************
      DIMENSION DBMAT(3,3)
      COMMON/PELEC/EMODU,POISS,THICK
      DO 10 I=1,3
      DO 10 J=1,3
10    DBMAT(I,J)=0.0
      RIGID=EMODU*THICK**3/(12.0*(1.0-POISS**2))
      DBMAT(1,1)=RIGID
      DBMAT(1,2)=POISS*RIGID
      DBMAT(2,1)=POISS*RIGID
      DBMAT(2,2)=RIGID
      DBMAT(3,3)=(1.0-POISS)*RIGID/2.0
      RETURN
      END

      SUBROUTINE GAUSS(PX,WEIGH,ISTRP)
*************************************************************
*                                                           *
*   GAUSSIAN POINT COORDINATES AND WEIGHTED COFFICIENTS     *
*                                                           *
*************************************************************
```

```
      DIMENSION PX(3),WEIGH(3)
      COMMON/ELEMT/NSTRP,NSECT,NZC,NPPOIN,QLOAD,ISUPT,NGUSS,
     &             NITEM,COOXQ(2,8),PPOIN(11,2),IZC(100)
      IF(NGUSS.EQ.3) THEN
      PX(1)=-1.0+2.0*(ISTRP-1)/NSTRP+0.22540/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1)/NSTRP+1.00000/NSTRP
      PX(3)=-1.0+2.0*(ISTRP-1)/NSTRP+1.77460/NSTRP
      WEIGH(1)=0.555556
      WEIGH(2)=0.888889
      WEIGH(3)=0.555556
      ELSE IF(NGUSS.EQ.2) THEN
      PX(1)=-1.0+2.0*(ISTRP-1)/NSTRP+0.42265/NSTRP
      PX(2)=-1.0+2.0*(ISTRP-1)/NSTRP+1.57735/NSTRP
      WEIGH(1)=1.0
      WEIGH(2)=1.0
      ELSE
      PX(1)=-1.0+2.0*(ISTRP-1)/NSTRP+1.00000/NSTRP
      WEIGH(1)=2.0
      END IF
      RETURN
      END

      SUBROUTINE MATRM(DB,D,B,L,M,N,LR,MR,NR)
C     ****************************************************************
C     *                                                              *
C     *     MATRIX MULTIPLICATION PROGRAM DB(M,N)=D(M,L)*B(L,N)      *
C     *                                                              *
C     ****************************************************************
      DIMENSION D(L,M),B(M,N),DB(L,N)
      DO 10 I=1,LR
      DO 10 J=1,NR
      DB(I,J)=0.0
      DO 10 K=1,MR
10    DB(I,J)=DB(I,J)+D(I,K)*B(K,J)
      RETURN
      END

      SUBROUTINE MATRMT(DBT,D,B,L,M,N,LR,MR,NR)
C     ****************************************************************
C     *                                                              *
C     *         MATRIX MULTIPLICATION AFTER THE FIRST ONE            *
C     *            IS TRANSPOSED   DB(M,N)=D(L,M)*B(L,N)             *
C     *                                                              *
C     ****************************************************************
      DIMENSION D(L,M),B(L,N),DBT(M,N)
      DO 10 I=1,MR
      DO 10 J=1,NR
      DBT(I,J)=0.0
      DO 10 K=1,LR
10    DBT(I,J)=DBT(I,J)+D(K,I)*B(K,J)
      RETURN
      END

      SUBROUTINE MATINV(A,N)
C     ****************************************************************
C     *                                                              *
C     *                  THE PROGRAM OF INVERSION                    *
C     *                                                              *
C     ****************************************************************
      DIMENSION INDEX(600,2),A(600,600)
```

```
      DO 10 I=1,N
10    INDEX(I,1)=0
      II=0
20    AMAX=-1.0
      DO 70 I=1,N
      IF(INDEX(I,1))70,40,70
40    DO 60 J=1,N
      IF(INDEX(J,1))60,50,60
50    TEMP=ABS(A(I,J))
      IF(TEMP-AMAX)60,60,55
55    IROW=I
      ICOL=J
      AMAX=TEMP
60    CONTINUE
70    CONTINUE
      IF(AMAX)170,180,75
75    INDEX(ICOL,1)=IROW
      IF(IROW-ICOL)80,100,80
80    DO 90 J=1,N
      TEMP=A(IROW,J)
      A(IROW,J)=A(ICOL,J)
90    A(ICOL,J)=TEMP
      II=II+1
      INDEX(II,2)=ICOL
100   PIVOT=A(ICOL,ICOL)
      A(ICOL,ICOL)=1.0
      PIVOT=1.0/PIVOT
      DO 110 J=1,N
110   A(ICOL,J)=A(ICOL,J)*PIVOT
      DO 140 I=1,N
      IF(I-ICOL)120,140,120
120   TEMP=A(I,ICOL)
      A(I,ICOL)=0.0
      DO 130 J=1,N
130   A(I,J)=A(I,J)-A(ICOL,J)*TEMP
140   CONTINUE
      GOTO 20
150   ICOL=INDEX(II,2)
      IROW=INDEX(ICOL,1)
      DO 160 I=1,N
      TEMP=A(I,IROW)
      A(I,IROW)=A(I,ICOL)
160   A(I,ICOL)=TEMP
      II=II-1
170   IF(II)150,190,150
180   WRITE(6,1001)
1001  FORMAT(1H0,2X,11H ZERO PIVOT)
190   CONTINUE
      RETURN
      END

      SUBROUTINE MATI3(A,N)
C     ************************************************************
C     *                                                          *
C     *                THE PROGRAM OF INVERSION                  *
C     *                                                          *
C     ************************************************************
      DIMENSION INDEX(80,2),A(80,80)
      DO 10 I=1,N
10    INDEX(I,1)=0
```

```
      II=0
 20   AMAX=-1.0
      DO 70 I=1,N
      IF(INDEX(I,1))70,40,70
 40   DO 60 J=1,N
      IF(INDEX(J,1))60,50,60
 50   TEMP=ABS(A(I,J))
      IF(TEMP-AMAX)60,60,55
 55   IROW=I
      ICOL=J
      AMAX=TEMP
 60   CONTINUE
 70   CONTINUE
      IF(AMAX)170,180,75
 75   INDEX(ICOL,1)=IROW
      IF(IROW-ICOL)80,100,80
 80   DO 90 J=1,N
      TEMP=A(IROW,J)
      A(IROW,J)=A(ICOL,J)
 90   A(ICOL,J)=TEMP
      II=II+1
      INDEX(II,2)=ICOL
 100  PIVOT=A(ICOL,ICOL)
      A(ICOL,ICOL)=1.0
      PIVOT=1.0/PIVOT
      DO 110 J=1,N
 110  A(ICOL,J)=A(ICOL,J)*PIVOT
      DO 140 I=1,N
      IF(I-ICOL)120,140,120
 120  TEMP=A(I,ICOL)
      A(I,ICOL)=0.0
      DO 130 J=1,N
 130  A(I,J)=A(I,J)-A(ICOL,J)*TEMP
 140  CONTINUE
      GOTO 20
 150  ICOL=INDEX(II,2)
      IROW=INDEX(ICOL,1)
      DO 160 I=1,N
      TEMP=A(I,IROW)
      A(I,IROW)=A(I,ICOL)
 160  A(I,ICOL)=TEMP
      II=II-1
 170  IF(II)150,190,150
 180  WRITE(6,1001)
 1001 FORMAT(1H0,2X,11H ZERO PIVOT)
 190  CONTINUE
      RETURN
      END
```