Free Vibration Analysis of Straight and Horizontally Curved Steel I-girder Bridges

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Bachelor of Technology

in

Civil Engineering

by

CHANDAN RAI

Roll No. 10301014

Department of Civil Engineering,

National Institute Technology - Rourkela,

Rourkela – 769008

2007

Free Vibration Analysis of Straight and Horizontally Curved Steel I-girder Bridges

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Bachelor of Technology
in
Civil Engineering

by

CHANDAN RAI
Roll No. 10301014

Under the guidance of
Dr. M.R. BARIK
(Assistant Professor)

Department of Civil Engineering,
National Institute Technology - Rourkela
Rourkela – 769008

2007

National Institute of Technology
Rourkela

CERTIFICATE

This is to certify that the thesis entitled, "**Free Vibration Analysis of Straight and Horizontally Curved Steel I-girder Bridges** " submitted by **Sri Chandan Rai** in partial fulfillment of the requirements for the award of Bachelor of Technology Degree in Civil Engineering at the National Institute of Technology, Rourkela (Deemed University) is an authentic work carried out by him under my supervision and guidance.

Date

Dr.M.R.Barik
Dept. of Civil Engineering
National Institute of
Technology
Rourkela - 769008

National Institute of Technology

Rourkela

ACKNOWLEDGEMENT

I would like to articulate my deep gratitude to my project guide **Dr. M.R.Barik** who has always been my motivation for carrying out the project.

I would like to thank the computer lab in charge M.tech. Students for their timely inputs and fulsome support. Last, but certainly not the least, I wish to thank my friends who stood beside through all my endeavors.

Chandan Rai

Date

Dept. of Civil Engineering
National Institute of Technology
Rourkela - 769008

CONTENTS

National Institute of Technology

Rourkela

ABSTRACT

In this project report a finite element formulation for free vibration analysis of straight beam and horizontally curved steel I-girder bridges has been presented .Stiffness as well as mass matrices of the curved and the straight beam elements is formulated. Each node of both of them possesses seven degrees of freedom including the warping degree of freedom. The curved beam element is derived based on the kang and Yoo's thin walled curved beam theory in 1994.Computer programs has been presented also to carry out free vibration analysis of the various bridges. The comparison between straight beam and curved beam results has been also presented. The numerical formulation is extensively applied to investigate free vibration characteristics of the bridges considering effects of the initial curvature, boundary condition, modeling method, and degrees of freedom of cross frame. The information which help practicing engineers better understand the vibration characteristics.

FIGURES AND TABLES:

Chapter 1

INTRODUCTION

INTRODUCTION

Horizontally curved steel I-girder bridges possess numerous beneficial merits such as reduction in total construction cost and time. In addition, they can be constructed in continuous form unlike the case of straight girders, which makes the bridges aesthetically pleasing as well as more economical with the reduced girder depth . Despite the merits stated above, horizontally curved steel I-girder bridges have often failed to attract practicing design engineers simply due to the lack of information on the complex behavior.

Presented herein is a finite element formulation for dynamic analysis of horizontally curved steel I-girder bridges. Stiffness as well as mass matrices of both the curved and straight beam elements are formulated. Each node of the curved beam element and the straight beam element possesses seven degrees of freedom including the warping degree of freedom. The curved beam element is derived based on the Kang and Yoo's theory of thin-walled curved beams

A computer program is presented here to achieve dynamic analyses of various horizontally curved steel I-girder bridges. The developed computer program is applied to investigate the characteristics of free vibration of horizontally curved steel I-girder bridges with subtended angles, various conditions of cross frame, modeling method, and support conditions.

Chapter 2

Formulation of mass and stiffness matrices for straight beam element:

Formulation of mass and stiffness matrices for straight beam element:

Each node of the straight beam element possesses seven degree of freedom including warping moment.

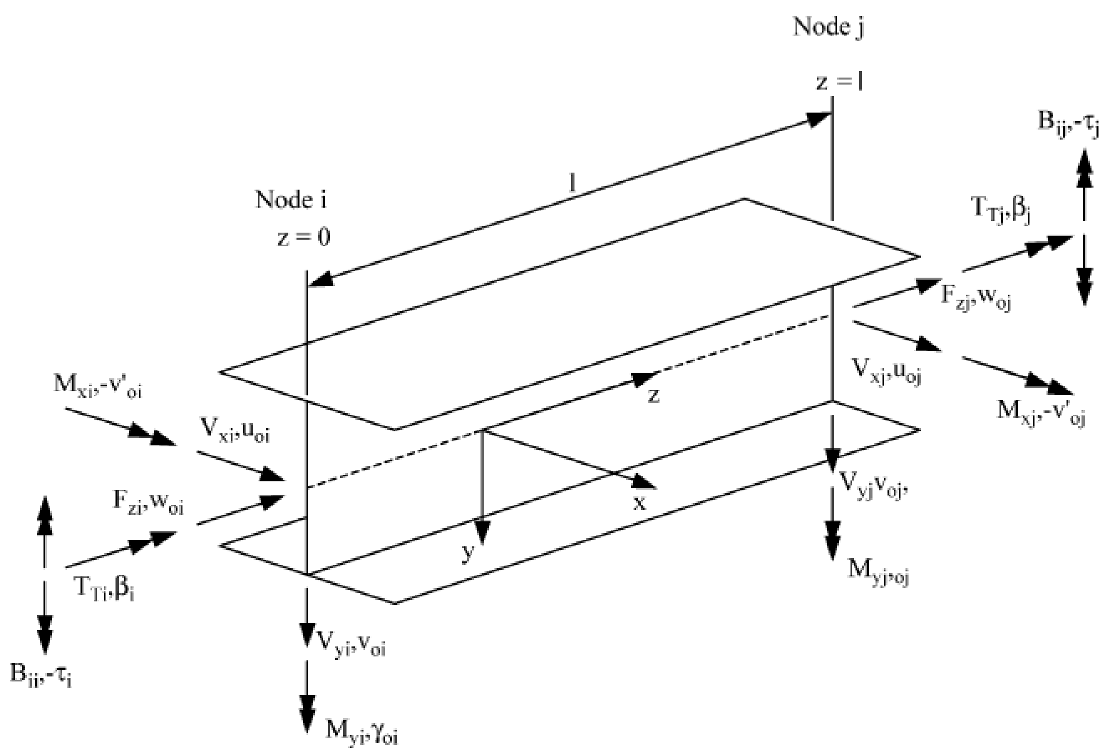

Fig. 2. Nodal forces and corresponding nodal displacements of the straight beam element.

For the linear elastic body, the variational strain energy stored in the body V is given by

$$\delta U = \int_V \varepsilon ij \quad \tau ij \quad dV \qquad (1)$$

where εij -component of strain tensor

τij –stress tensor

The variational kinetic energy of a thin walled curved beam is

$$\delta T = \int_V (\rho\, \partial^2 U_i)/(\partial t^2)\, \delta U_i \; dV \qquad (2)$$

ρ = mass density
Ui = displacement component of straight beam
t= time
the variational potential energy=

$$\delta V = \int_l q_i\, \delta U_i \, dz \qquad (3)$$

qi= distributed loads applied on the line of shear centre and l is the length of element.

2.1 Equations of motion:

$$\rho A \ddot{U}_o - \rho I_y \ddot{U}'' + E I_y U_o = q_x - m_y' \qquad (4a)$$

$$\rho A \ddot{V}_o - \rho I_x \ddot{V}'' + E I_x V_o = q_y + m_x' \qquad (4b)$$

$$\rho A \ddot{W}_o - E A W_o' = q_z \qquad (4c)$$

$$\rho (I_x + I_y) \ddot{\boldsymbol{\beta}} - \rho I_w \ddot{\rho}'' + E I_w \beta - G K_t \beta'' = m_z + m_w' \qquad (4d)$$

Here every displacements component fields Uo,Vo,Wo and β are not coupled with one another and hence can be formulated separately.

2.2 Shape matrix, Mass matrix, Stiffness matrix and Force vector

2.2.1 Case I: Bar element:

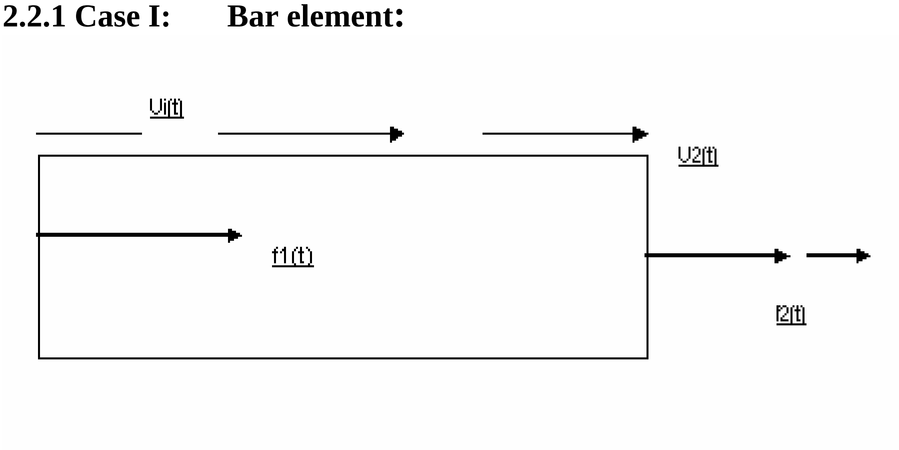

Figure 3.

U (0,t) = U1(t)
U (l,t) =U2(t)
U (z,t) =a(t) + b(t)z---------------(1)
Applying boundary condions:
a (t) = U1(t)
And a (t) + b (t) l = U2 (t)
Or b (t) = (U2 (t) – U1 (t))/l

Substituting in equation (1) we get
U (z,t) = (1-z/l)U1(t) + (z/l) U2(t)
U (z,t)= N1(z).U1(t) +N2(z).U2(t)
N1 (z) = (1-z/l)
N2 (z) = z/l;
N = [(1-z/l), z/l]
U (z,t) =(1-(z/l)) U1(t) + (z/l) U2(t)

Kinetic energy:

$$T(t) =(1/2) \int_0^L \rho A \left\{\partial U(z,t) / \partial(t)\right\}^2 .dz =(1/2) \dot{U}(t)^T[m] \dot{U}(t) \qquad (5)$$

Elements of mass matrix:

$$a11 = \int_0^l (1+z^2/l^2 - 2z/l)\, dz$$
$$=l/3$$

$$a12= a21= \int_0^l (z/l - z^2/ l^2)\, dz$$
$$=l/6$$

$$a22 = \int_0^l (z^2/ l^2)\, dz = 2l/6;$$

Consistent mass matrix:

Consistent matrix is given by:

$$[m] = (\rho\ l/6)\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

Strain energy:

$$VT(t) = (1/2)\int_0^l EI\left\{\partial^2 \dot{U}(z,t) / \partial^2(z)\right\}^2 . dz = (1/2)\, \dot{U}(t)^T [k]\, U(t) \qquad (6)$$

Elements of stiffness matrix =

$$a11 = \int_0^l (1/l^{2)} \cdot dz = 1/l;$$

$$a12 = a21 = \int_0^l (-1/l^2)\, dz = -1/l;$$

$$a22 = \int_0^l (1/l^2)\, dz = 1/l;$$

Stiffness matrix: stiffness matrix is given by:

$$[K] = (EA/l) \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

2.2.2 Case II: Beam element:

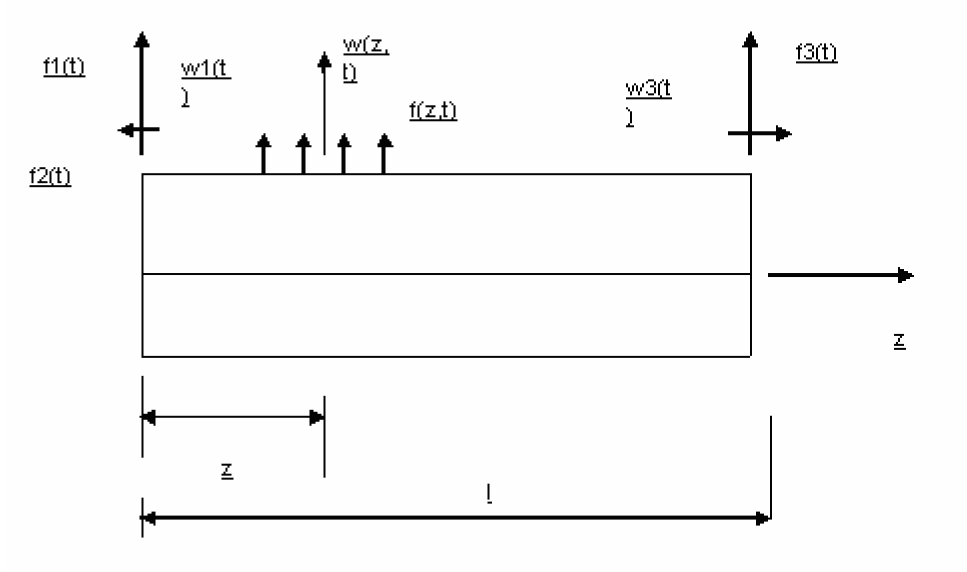

Figure 4.

Transverse force distribution f(z,t).in this case ,the joint undergo both translation and rotational displacement ,so the unknown joint displacements are labeled as w1(t),w2(t),w3(t),w4(t). Thus there will be linear joint force f1 (t), and f3 (t) corresponding to linear displacements w1 (t) and w3 (t) and rotational joint forces (bending moments) f2 (t) and f4 (t) corresponding to the rotational joint w2 (t) and w4 (t) respectively.

Transverse displacements with element are assumed to be a cubic in z.

$W(z,t) = a(t) + b(t)z + c(t)z^2 + d(t)z^3$

Boundary conditions:

$w(0,t) = w1(t)$, $\partial w/\partial t(0,t) = w2(t)$

$w(l,t) = w3(t)$, $\partial w/\partial t(l,t) = w4(t)$

So,

$a(t) = w1(t)$

$b(t) = w2(t)$

$c(t) = (1/l^2)[-3w1(t) - 2w2(t)l + 3w3(t) - w4(t)l]$

$d(t) = (1/l^3)[2w1(t) + w2(t)l - 2w3(t) + w1(t)l]$

So w (z,t) is given by:

$w(z,t) = (1-3x^2/l^2 + 2x^3/l^{3)}\ w1(t) + (z/l - 2z^2/l^2 + z^3/l^3)\ w2(t) + 3(z^2/l^2 - 2z^3/l^3)w3(t) + (-z^2/l^2 + z^3/l^3)lw4(t).$

$$w(z,t)=\sum_{i=1}^{4} Ni(t)wi(t) \qquad (7)$$

Ni (z) is the shape function given by

$N1(z) = 1-3z^2/l^2 + 2z^3/l^3$

$N2(z) = z - 2lz^2/l^2 + l\, z^3/l^3$

$N3(z) = 3z^2/l^2 - 2z^3/l^3$

$N4(z) = -l\,(z^2/l^2) + l\, z^3/l^3$

Formulation of Mass Matrix:

Kinetic energy equation:

$$T(t) = (1/2)\int_0^l \rho A\ \{\partial W(z,t)/\partial(t)\}^2 .dz = (1/2)\ \dot{W}(t)^T[m]\ \dot{W}(t)$$

Elements of mass matrix

$$a11=\int_0^l (1-3(z^2/l^2) + 2(z^2/l^3).dz$$

$= (156/420)l$

$$a21=a12=l\int_0^l (1-3(z^2/l^2) + 2(z^2/l^3).\ (z/l - 2z^2/l^2 + z^2/l^3)\ dz$$

$= (22/420)l^2$

$$a13=a31=\int_0^l (1-3(z^2/l^2) + 2(z^2/l^3).(3z^2/l^2 - 2z^3/l^3).dz$$

$= (54/420)l$

$$a14=a41=l\int_0^l (1-3(z^2/l^2) + 2(z^2/l^3).(-z^2/l^2 + z^2/l^3)dz$$

$= (-13/420)l^2$

$$a22=l^2\int_0^l (z/l - 2z^2/l^2 + z^2/l^3)^2\ dz$$

$(4\ l^2/420)$

$$a23=a32=l\int_0^l (z/l - 2z^2/l^2 + z^2/l^3)\ (3z2/l^2 - 2z^3/l^3).dz$$

$= (13/420)l^2$

$$a24=a42=l^2\int_0^l (z/l - 2z^2/l^2 + z^2/l^3)(\ -z^2/l^2 + z^2/l^3)dz$$

$= -(3/420)l^3$

$a33=\int_0^l (3z^2/l^2 - 2z^3/l^3)^2\, dz$

$= (156/420)l$

$a34=a43=l\int_0^l (3z^2/l^2 - 2z^3/l^3).(-z^2/l^2 + z^2/l^3)dz$

$= -(22/420)l^2$

$a44=\int_0^l (-z^2/l^2 + z^2/l^3)^2\, l^2\, dz$

$= (4/105)l^3$

Mass matrix:

$$[M] = (\rho Al/420)\begin{pmatrix} 156 & 22l & 54 & -13l \\ 22l & 4l^2 & 13l & -3l^2 \\ 54 & 13l & 156 & -22l \\ -13l & -3l^2 & -22l & 4l^2 \end{pmatrix} \qquad (8)$$

Formulation for stiffness matrix:

Strain energy is given by:

$$VT(t) = (1/2)\int_0^l EI\,\{\partial^2 W(z,t)/\partial^2(z)\}^2\,.dz = (1/2)\,\dot{W}(t)^T[k]\,\dot{W}(t)$$

The expression: $\partial^2 W(z,t)/\partial^2(z) = (12z/l^3 - 6/l^2)W1(t) + (-4/l^2 + 6z/l^3)lW2(t) + (6/l^2 - 12z/l^3)W3(t) + (-2/l^2 + 6z/l^3)lW4(t)$

After calculation the members of matrix are:

$a11= (12/l^3)$
$a12= (6l/l^{3)}$
$a14 = (6l/l^3)$
$a13= (-12/l^3)$
$a22= (4l^2/l^3)$
$a23= (-6l/l^3)$

a24 = $(2l^2/l^3)$
a33= $(12/l^3)$
a34= $(-6l/l^3)$
a44= $(4l^2/l^3)$

Stiffness matrix is given by:

$$[K]=(EI/l^3)\begin{pmatrix} 12 & 6l & -12 & 6l \\ 6l & 4l^2 & -6l & 2l^2 \\ -12 & -6l & 12 & -6l \\ 6l & 2l^2 & -6l & 4l^2 \end{pmatrix} \qquad (9)$$

2.3 Mass matrix and stiffness matrix for straight beam element:

Based on the calculations done above the stiffness matrix for straight beam element is given as

$$\begin{Bmatrix} Uo \\ Vo \\ Wo \\ \beta \end{Bmatrix} = \begin{pmatrix} Nu & 0 & 0 & 0 \\ 0 & Nv & 0 & 0 \\ 0 & 0 & Nw & 0 \\ 0 & 0 & 0 & N\beta \end{pmatrix} \begin{Bmatrix} d^U \\ d^V \\ d^W \\ d^\beta \end{Bmatrix} \qquad (10)$$

Where

$$Nu = \left[\; 2-(z/l)^2+2(z/l)^3 \quad l(z/l -2(z/l)^2+(z/l)^3 \quad 3(z/l)^2-2(z/l)^3 \quad l(-(z/l)^2+(z/l)^3 \;\right] \qquad (11a)$$

$$Nv = N\beta = \left|\; 1-3z^2/l^2 +2z^3/l^3 \quad z- 2lz^2/l^2 +l\, z^3/l^3 \quad 3z^2/l^2 -2z^3/l^3 \quad l\,(z^2/l^2)- l(z^3/l^3) \;\right| \qquad (11b)$$

$$Nw=\left[(1-z/l \quad z/l\right] \qquad (11c)$$

Where the nodal displacements are

d^u =[Uoi γ i Uoj γ j $]^T$ (11d)

d^V = [Voi -Voi' Voj -Voj'$]^T$ (11e)

d^w = [Woi   Woj $]^T$ (11f)

d^β = [βi -τi βi -τj]

2.3.1. Stiffness matrix:

From the variational energy and shape function formulated as above ,we can get

$$K = \begin{pmatrix} EIa & 0 & 0 & 0 \\ 0 & EIx & 0 & 0 \\ 0 & 0 & EAKc & 0 \\ 0 & 0 & 0 & EIw + GKtKe \end{pmatrix} \quad (12)$$

Where:

$$Ka = (1/l^3)\begin{pmatrix} 12 & 6l & -12 & 6l \\ 6l & 4l^2 & -6l & 2l^2 \\ -12 & -6l & 12 & -6l \\ 6l & 2l^2 & -6l & 4l^2 \end{pmatrix} \quad (13a)$$

$$Kb = (1/l^3)\begin{pmatrix} 12 & 6l & -12 & -6l \\ 6l & 4l^2 & -6l & 2l^2 \\ -12 & -6l & 12 & 6l \\ -6l & 2l^2 & 6l & 4l^2 \end{pmatrix} \quad (13b)$$

$$Kc = (1/l)\begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix}$$

(13c)

$$Kd = (1/l^3)\begin{pmatrix} 12 & -6l & -12 & -6l \\ -6l & 4l^2 & 6l & 2l^2 \\ -12 & 6l & 12 & 6l \\ -6l & 2l^2 & 6l & 4l^2 \end{pmatrix}$$

(13d)

$$Ke -(1/30\ l)\begin{pmatrix} 36 & -3l & -36 & -3l \\ -3l & 4l^2 & 3l & -l^2 \\ -36 & 3l & 36 & 3l \\ -3l & -l^2 & 3l & 4l^2 \end{pmatrix}$$

(13e)

2.3.2 Mass Matrix:

From the variational kinetic energy and shape function the mass matrix is derived:

$$M= (\rho)\begin{pmatrix} AMa+IyMe & 0 & 0 & 0 \\ 0 & AMb+IxMf & 0 & 0 \\ 0 & 0 & AMc & 0 \\ 0 & 0 & 0 & (Ix+Iy)Mb+IwMf \end{pmatrix}$$

(14)

Where:

$$Ma = (l/420)\begin{pmatrix} 156 & 22l & 54 & -13l \\ 22l & 4l^2 & 13l & -3l^2 \\ 54 & 13l & 156 & -22l \\ -13l & -3l^2 & -22l & 4l^2 \end{pmatrix} \tag{15a}$$

$$Mb = (l/420)\begin{pmatrix} 156 & -22l & 54 & -13l \\ -22l & 4l^2 & -13l & -3l^2 \\ 54 & 13l & 156 & 22l \\ -13l & -3l^2 & 22l & 4l^2 \end{pmatrix} \tag{15b}$$

$$Mc = (l/6)\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} \tag{15c}$$

$$Me = (1/30\ l)\begin{pmatrix} 36 & 3l & -36 & 3l \\ & 4l^2 & -3l & -l^2 \\ & & 36 & -3l \\ sym & & & 4l^2 \end{pmatrix} \tag{15d}$$

$$Mf = (1/30\ l)\begin{pmatrix} 36 & -3l & -36 & -3l \\ & 4l^2 & -3l & -l^2 \\ & & 36 & 3l \\ sym & & & 4l^2 \end{pmatrix} \tag{15e}$$

Chapter 3

Formulation of mass and stiffness matrices for Curved beam element:

Formulation of mass and stiffness matrices for Curved beam element:

In the present study, the curved beam element is derived based on the Kang and Yoo's theory of thin- walled curved beams. The curved beam element is shown in Fig. 1 in curvilinear coordinate system. Each node of the curved beam element possesses seven degrees of freedom including the warping degree of freedom. The cross-section applicable herein is a doubly symmetric open section .

3.1. Equations of motion:

Using Hamilton's principle, the dynamic equilibrium can be expressed in the Variation form as following

$$\int_{t_1}^{t_2} (\delta T + \delta U + \delta V)\, \mathrm{d}t = 0 \qquad (16)$$

where dT is the variational kinetic energy, dU is the variational strain energy, and dV_ is the variational potential energy loss due to applied loads. The symbol d means the first variation.

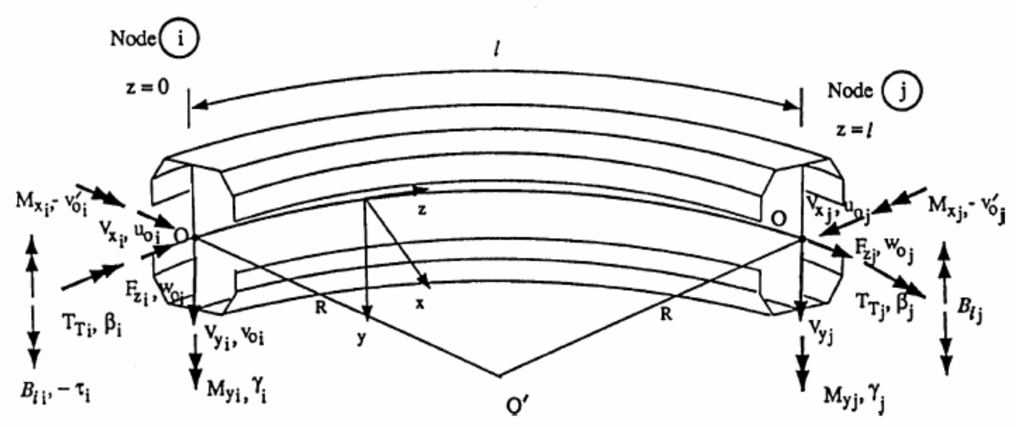

Fig. 1. Nodal forces and corresponding nodal displacements of the curved beam element.

For the linear elastic body, the variational strain energy stored in the body, V is

$$\delta U = \int_V \tau_{ij}\delta\varepsilon_{ij}\,\mathrm{d}V \tag{17}$$

where εij refers to the components of the strain tensor and τij to those of the stress tensor. On the one hand, the variational kinetic energy of a thin-walled curved beam is

$$\delta T = \int_V \rho\,\frac{\partial^2 u_i}{\partial t^2}\,\delta u_i\,\mathrm{d}V \tag{18}$$

where ρ is the mass density, ui are the displacement components of the curved beam, and t is time. On the other hand, with body forces neglected, the variational potential energy loss due to applied loads is

$$\delta \mathcal{V} = -\int_l q_i\delta u_i\,\mathrm{d}z \tag{19}$$

where qi stands for distributed loads applied on the line of shear center and l is the length of the element. Substituting the strain-displacement relationship and the stress resultant–displacement relationship into Eqs. (1)–(4), and carrying out the conventional procedure of the calculus of variation, the following set of equations of motion is obtained .

$$\rho\left(A\ddot{u}_0 - I_y\ddot{u}_0'' - \frac{2I_y}{R}\ddot{w}_0'\right) - \frac{EA}{R}\left(w_0' - \frac{u_0}{R}\right) + EI_y\left(u_0^{\mathrm{IV}} + \frac{2}{R^2}u_0'' + \frac{1}{R^3}w_0'\right) = q_x - m_y'$$

(20a)

$$\rho\left[A\ddot{v}_0 - \left(I_x + \frac{I_\omega - 2K_{xy\omega}}{R^2}\right)\ddot{v}_o'' - \frac{I_y}{R}\ddot{\beta} - \frac{I_\omega - K_{xy\omega}}{R}\ddot{\beta}''\right] + EI_x\left(v_0^{\mathrm{IV}} - \frac{1}{R}\beta''\right)$$
$$+ \frac{EK_{xy\omega}}{R}\left(\beta^{\mathrm{IV}} + \frac{2}{R}v_0^{\mathrm{IV}} - \frac{\beta''}{R^2}\right) + \frac{EI_\omega}{R}\left(\beta^{\mathrm{IV}} + \frac{v_0^{\mathrm{IV}}}{R}\right) - \frac{GK_{\mathrm{T}}}{R}\left(\beta'' + \frac{v_0''}{R}\right) = q_y + m_x'$$

$$\rho\left[\left(A + \frac{3I_y}{R^2}\right)\ddot{w}_0 + \frac{2I_y}{R}\ddot{u}_0\right] + E\left(\frac{I_y}{R^2} - A\right)\left(w_0'' - \frac{1}{R}u_0'\right) = q_z$$

$$\rho\left[(I_x+I_y)\ddot{\beta}-I_\omega\ddot{\beta}''-\frac{I_y}{R}\ddot{v}_0-\frac{I_\omega-K_{xy\omega}}{R}\ddot{v}_0''\right]-\frac{EI_x}{R}\left(v_0''-\frac{1}{R}\beta\right)$$

$$+\frac{EK_{xy\omega}}{R}\left(v_0^{\mathrm{IV}}-\frac{2}{R}\beta''-\frac{1}{R^2}v_0''\right)+EI_\omega\left(\beta^{\mathrm{IV}}+\frac{1}{R}v_0^{\mathrm{IV}}\right)-GK_{\mathrm{T}}\left(\beta''+\frac{1}{R}v_0''\right)$$

$$=m_z+m_\omega'$$

where the reference displacements u_0, v_0, w_0, and β are displacements of the centroid in the x, y, and z directions and a rotation of the cross-section about z-axis, respectively. Displacement components u_0 and w_0 are associated with in-plane of curvature displacement field while displacement components v_0 and β are referenced with out-of-plane of curvature displacement field.

The linear equations of motion given in Eqs. (5a)–(5d) are partially, if not completely, uncoupled. It is observed that u_0 and w_0 appear only in Eqs. (5a) and (5c) whereas v_0 and β are present only in Eqs. (5b) and (5d), which means that two displacement fields related with in-plane of curvature and out-of-plane of curvature, respectively are fully separated each other.

3.2. Finite element formulation:

Analytical solutions of Eqs. (5a)–(5d) require extremely complicated mathematical Processes. Furthermore, analytical solutions are quite limited to certain types of loading and boundary conditions. In the present study, among the various numerical methods, the finite element method is employed. A linear stiffness matrix and a consistent mass matrix are developed so that various analyses such as linear and free vibration analyses can be performed. Using shape functions, the dynamic equilibrium given in Eq. (1) yields a set of simultaneous equations

$$\delta T+\delta U+\delta V=\delta\mathbf{d}^{\mathrm{T}}[\mathbf{M}d+\mathbf{K}d-\mathbf{f}]=0 \quad (21)$$

From which we can obtain

$$\mathbf{Md}+\mathbf{Kd}-\mathbf{f}=0$$

where K, M, d, and f are the linear stiffness matrix, the consistent mass matrix, the nodal displacement vector, and the applied force vector of a global structural system, respectively.

The nodal forces and the corresponding nodal displacements are shown in Fig. 1 in the positive senses. Used as nodal forces in the present study are seven stress resultants, Fz, Mx, My, Bi, TT, Vx, and Vy. The corresponding nodal displacements are w0, γ, -Vo', -τi, β ,

u0, and v0 where γ and τ are defined as

$$\gamma = u_0' + \frac{w_0}{R} \tag{22}$$

$$\tau = \beta' + \frac{v_0'}{R} \tag{23}$$

w0, u0, and γ describe the in-plane displacements whereas v0, -Vo' , β , and -τ are the out-of-plane displacements. These two parts of displacement fields are not coupled with each other and can be formulated separately. It is convenient to express the element nodal displacement and force vectors in the following form

$$\mathbf{d} = [\mathbf{d}^{\text{in T}}\mathbf{d}^{\text{out T}}]^{\text{T}} = d_k, \quad k = 1, 2, 3, \ldots, \text{and } 14 \tag{24}$$

$$\mathbf{f} = [\mathbf{f}^{\text{in T}}\mathbf{f}^{\text{out T}}]^{\text{T}} = f_k, \quad k = 1, 2, 3, \ldots, \text{and } 14 \tag{25}$$

Where

$$\mathbf{d}^{\text{in}} = [\, w_{0i} \quad u_{0i} \quad \gamma_i \quad w_{0j} \quad u_{0j} \quad \gamma_j \,]^{\text{T}} = d_k^{\text{in}}, \quad k = 1, 2, \ldots, \text{and } 6$$

$$\mathbf{d}^{\text{out}} = [\, v_{0i} \quad -v_{0i}' \quad \beta_i \quad -\tau_i \quad v_{0j} \quad -v_{0j}' \quad \beta_j \quad -\tau_j \,]^{\text{T}} = d_k^{\text{out}}, \quad k = 7, 8, \ldots, \text{and } 14$$

$$\mathbf{f}^{\text{in}} = [\, F_{zi} \quad V_{xi} \quad M_{yi} \quad F_{zj} \quad V_{xj} \quad M_{yj} \,]^{\text{T}} = f_k, \quad k = 1, 2, \ldots, \text{and } 6$$

$$\mathbf{f}^{\text{out}} = [\, V_{yi} \quad M_{xi} \quad T_{\text{T}i} \quad B_{ii} \quad V_{yj} \quad M_{xj} \quad T_{\text{T}j} \quad B_{ij} \,]^{\text{T}} = f_k^{\text{out}}, \quad k = 7, 8, \ldots, \text{and } 14$$

The superscripts 'in' and 'out' indicate association with the in-the-curvature-plane and the out-of-the-curvature-plane behaviors, respectively. The subscripts i and j denote the node numbers and the superscript 'T' denotes the transposition. In the present study, the homogeneous solutions of the differential equations in static equilibrium are employed as shape functions Then, the displacement fields can be expressed in terms of nodal displacements as following

$$\mathbf{u} = \mathbf{N}\mathbf{d}$$

Where

$$\mathbf{u} = [\, u_0 \quad w_0 \quad v_0 \quad \beta \,]^{\text{T}} \tag{26}$$

$$\mathbf{N} = \begin{bmatrix} \mathbf{N}^{in}{}_{(2\times6)} & 0 \\ 0 & \mathbf{N}^{out}{}_{(2\times8)} \end{bmatrix} \tag{27}$$

3.3 Shape functions for curved beam element:

Displacements of a point in a curved beam element are approximately represented by shape functions along with nodal displacements. The best description for the displacement field can be achieved by the exact solutions of the governing differential equations if they are available. In the present study the shape functions are based on the homogeneous solutions of the linear differential equations governing the static behavior of curved beams given by kang and Yoo.

Based on a usual procedure of relating the displacement field (u) the corresponding nodal displacement vector (d) , the following equation can be established

$$\mathbf{u} = \mathbf{C}\mathbf{A}^{-1}\mathbf{d} = \mathbf{N}\mathbf{d}$$

Since the inplane and out of the plane forces,are not coupled in the static behavior, it is convenient to separate these two displacement fields. Hence

$$\mathbf{u}^{in} = \mathbf{C}^{in}(\mathbf{A}^{in})^{-1}\mathbf{d}^{in} = \mathbf{N}^{in}\mathbf{d}^{in} \tag{28a}$$

$$\mathbf{u}^{out} = \mathbf{C}^{out}(\mathbf{A}^{out})^{-1}\mathbf{d}^{out} = \mathbf{N}^{out}\mathbf{d}^{out} \tag{28b}$$

Where superscript “ -1 “ = matrix inversion

The matrices relating the displacement fields and the vector of generalized coordinates in the homogeneous solutions are given by

$$\mathbf{C}^{in} = \begin{bmatrix} 1 & c & s & \xi c & \xi s & 0 \\ F_0\xi & s & -c & c + \xi s & s - \xi c & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{C}_1(\xi)^T \\ \mathbf{C}_2(\xi)^T \end{bmatrix} \tag{29a}$$

$$\mathbf{C}^{out} = \begin{bmatrix} 1 & \xi & \text{ch}\alpha_1\xi & \text{sh}\alpha_1\xi & \text{c} & \text{s} & \xi\text{c} & \xi\text{s} \\ 0 & 0 & \frac{\alpha_3\text{ch}\alpha_1\xi}{R} & \frac{\alpha_3\text{sh}\alpha_1\xi}{R} & \frac{-\text{c}}{R} & \frac{-\text{s}}{R} & \frac{-\alpha_2\text{s} + \xi\text{c}}{R} & \frac{\alpha_2\text{c} - \xi\text{s}}{R} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{C}_3(\xi)^T \\ \mathbf{C}_4(\xi)^T \end{bmatrix} \quad (29b)$$

Or

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}^{in} & \mathbf{0} \\ \mathbf{0} & \mathbf{C}^{out} \end{bmatrix} \quad (29c)$$

The definitions are used in the above equations

$$\xi = \frac{z}{R} \quad (30a)$$

$$F_0 = \left(1 - \frac{I_y}{AR^2}\right)^{-1} \quad (30b)$$

$$\alpha_1 = \left[\frac{EI_xGK_TR^2}{EI_xEI_y - (EK_{xy\omega}/R)}\right]^{1/2} \quad (30c)$$

$$\alpha_2 = \frac{2\left(EI_x + \frac{1}{R^2}EK_{xy\omega}\right)}{EI_x + GK_T + \frac{1}{R^2}(EI_\omega + 2EK_{xy\omega})} \quad (30d)$$

$$\alpha_3 = \frac{\alpha_1^2\left(EI_x + \frac{1}{R^2}EI_\omega + \frac{2}{R^2}EK_{xy\omega}\right) - GK_T}{EI_x + GK_T + \frac{1}{R^2}EK_{xy\omega} - \left(\frac{\alpha_1}{R}\right)^2(EI_\omega + EK_{xy\omega})} \quad (30e)$$

“Cos ε” and “sin ε” are abbreviated as “c” and “s” stand for hyperbolic functions “cosh” and “sinh” ,respectively.

Finally, the matrices relating the nodal displacement vectors and vector of generalized coordinate are given by

$$
\mathbf{A}^{in} = \begin{bmatrix}
0 & 0 & -1 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \frac{2}{R} & 0 & \frac{1}{R} \\
F_0\xi_l & s_l & -c_l & c_l + \xi_l s_l & s_l - \xi_l c_l & 1 \\
1 & c_l & s_l & \xi_l c_l & \xi_l s_l & 0 \\
\frac{F_0\xi_l}{R} & 0 & 0 & \frac{2c_l}{R} & \frac{2s_l}{R} & \frac{1}{R}
\end{bmatrix}
$$

(31a)

$$
\mathbf{A}^{out} = \begin{bmatrix}
1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 \\
0 & -\frac{1}{R} & 0 & -\frac{\alpha_1}{R} & 0 & -\frac{1}{R} & -\frac{1}{R} & 0 \\
0 & 0 & \frac{\alpha_3}{R} & 0 & -\frac{1}{R} & 0 & 0 & \frac{\alpha_2}{R} \\
0 & -\frac{1}{R^2} & 0 & \frac{-\alpha_1(\alpha_3+1)}{R^2} & 0 & 0 & \frac{\alpha_2}{R^2} & 0 \\
1 & \xi_l & ch_l & sh_l & c_l & s_l & \xi_l c_l & \xi_l s_l \\
0 & -\frac{1}{R} & \frac{-\alpha_1 sh_l}{R} & \frac{-\alpha_1 ch_l}{R} & \frac{s_l}{R} & -\frac{c_l}{R} & \frac{\xi_l s_l - c_l}{R} & \frac{-\xi_l c_l - s_l}{R} \\
0 & 0 & \frac{\alpha_3 ch_l}{R} & \frac{\alpha_3 sh_l}{R} & -\frac{c_l}{R} & -\frac{s_l}{R} & \frac{-\xi_l c_l - \alpha_2 s_l}{R} & \frac{-\xi_l s_l + \alpha_2 c_l}{R} \\
0 & -\frac{1}{R^2} & \frac{-\alpha_1(\alpha_3+1)}{R^2} sh_l & \frac{-\alpha_1(\alpha_3+1)}{R^2} ch_l & 0 & 0 & \frac{\alpha_2 c_l}{R^2} & \frac{\alpha_2 s_l}{R^2}
\end{bmatrix}
$$

(31b)

Where

$$\xi_l = \frac{l}{R}$$ (32a)

$$c_l = \cos\xi_l$$ (32b)

$$s_l = \sin\xi_l$$ (32c)

$$ch_l = \cosh\alpha_1\xi_l$$ (32d)

$$sh_l = \sinh\alpha_1\xi_l$$ (32e)

It should be noted that the matrix of shape function given in equation can also be portioned separating the inplane action and that out of plane action as

$$
\mathbf{N} = \begin{bmatrix} \mathbf{N}^{in}(2 \times 6) & \mathbf{0} \\ \mathbf{0} & \mathbf{N}^{out}(2 \times 8) \end{bmatrix}
$$

(33)

where u denote the reference displacement vector and N is the shape function matrix. It is noted that the shape function matrix is also uncoupled.

3.4 Stiffness matrix:

From the variational strain energy given in Eq. (2), following relations are derived

$$\delta U^{\text{in}} = \int_V E\varepsilon_z^{\text{in}}\delta\varepsilon_z^{\text{in}}\,\mathrm{d}V = \delta d_i K_{ij}^{\text{in}} d_j, \quad i,j = 1,2,\ldots,\text{and } 6$$

$$\delta U^{\text{out}} = \int_V E\varepsilon_z^{\text{out}}\delta\varepsilon_z^{\text{out}}\,\mathrm{d}V + \int_V G(\gamma_{zx}^{\text{out}}\delta\gamma_{zx}^{\text{out}} + \gamma_{yz}^{\text{out}}\delta\gamma_{yz}^{\text{out}})\,\mathrm{d}V = \delta d_i K_{ij}^{\text{out}} d_j$$

$$i,j = 7,8,\ldots,\text{and } 14$$

Where

$$\begin{aligned}
K_{ij}^{\text{in}} = {} & EA\int_l \left[(N_{2i}'N_{2j}') - \frac{1}{R}(N_{1i}N_{2j}' + N_{2i}'N_{1j}) + \frac{1}{R^2}(N_{1i}N_{1j})\right]\mathrm{d}z \\
& + EI_y\int_l \left[(N_{1i}''N_{1j}'') + \frac{1}{R}(N_{1i}''N_{2j}' + N_{2i}'N_{1j}'') + \frac{1}{R^2}(N_{2i}'N_{2j}')\right]\mathrm{d}z \\
& - \frac{EI_y}{R}\int_l \left[(N_{2i}'N_{1j}'' + N_{1i}''N_{2j}') + \frac{1}{R}(2N_{2i}'N_{2j}' - N_{1i}N_{1j}'' - N_{1i}''N_{1j})\right]\mathrm{d}z \\
& + \frac{EI_y}{R}\int_l \left[\frac{1}{R^2}(N_{1i}N_{2j}' + N_{2i}'N_{1j})\right]\mathrm{d}z
\end{aligned}$$

(34)

$$K_{ij}^{\text{out}} = EI_x \int_l \left[(N''_{3i}N''_{3j}) - \frac{1}{R}(N_{4i}N''_{3j} + N''_{3i}N_{4j}) + \frac{1}{R^2}(N_{4i}N_{4j}) \right] \mathrm{d}z$$

$$+ EI_\omega \int_l \left[(N''_{4i}N''_{4j}) + \frac{1}{R}(N''_{3i}N''_{4j} + N''_{4i}N''_{3j}) + \frac{1}{R^2}(N''_{3i}N''_{3j}) \right] \mathrm{d}z$$

$$+ \frac{EK_{xy\omega}}{R} \int_l \left[(N''_{3i}N''_{4j} + N''_{4i}N''_{3j}) + \frac{1}{R}(2N''_{3i}N''_{3j} - N_{4i}N''_{4j} - N''_{4i}N_{4j}) \right.$$

$$\left. - \frac{1}{R^2}(N_{4i}N''_{3j} + N''_{3i}N_{4j}) \right] \mathrm{d}z$$

$$+ GK_{\mathrm{T}} \int_l \left[(N'_{4i}N'_{4j}) + \frac{1}{R}(N'_{3i}N'_{4j} + N'_{4i}N'_{3j}) + \frac{1}{R^2}(N'_{3i}N'_{3j}) \right] \mathrm{d}z$$

(35)

3.5 Consistent mass matrix:

The consistent mass matrix of a curved beam is formulated on the same manner as the linear stiffness matrix, K by the virtue of the independence of the in-plane displacements and the out-of-plane displacements

$$\delta T^{\text{in}} = \int_V (\ddot{u}^{\text{in}} \delta u^{\text{in}} + \ddot{w}^{\text{in}} \delta w^{\text{in}}) \mathrm{d}V = \delta d_i M_{ij}^{\text{in}} \ddot{d}_j, \quad i, j = 1, 2, \ldots, \text{and } 6$$

$$\delta T^{\text{out}} = \int_V (\ddot{u}^{\text{out}} \delta u^{\text{out}} + \ddot{v}^{\text{out}} \delta v^{\text{out}} + \ddot{w}^{\text{out}} \delta w^{\text{out}}) \mathrm{d}V = \delta d_i M_{ij}^{\text{out}} \ddot{d}_j, \quad i, j = 7, 8, \ldots, \text{and } 14$$

Where

$$M_{ij}^{\text{in}} = \rho A \int_l (N_{1i}N_{1j} + N_{2i}N_{2j})\,\mathrm{d}z + \rho\frac{I_y}{R}\int_l \left(N_{2i}N'_{1j} + N'_{1i}N_{2j} + \frac{2}{R}N_{2i}N_{2j}\right)\mathrm{d}z + \rho I_y \int_l \left(N'_{1i}N'_{1j} + \frac{N'_{1i}N_{2j} + N_{2i}N'_{1j}}{R} + \frac{N_{2i}N_{2j}}{R^2}\right)\mathrm{d}z$$

(36)

$$M_{ij}^{\text{out}} = \rho A \int_l N_{3i}N_{3j}\,\mathrm{d}z - \rho\frac{I_y}{R}\int_l (N_{3i}N_{4j} + N_{4i}N_{3j})\,\mathrm{d}z + \rho I_y \int_l N_{4i}N_{4j}\,\mathrm{d}z + \rho I_x \int_l N_{4i}N_{4j} + N'_{3i}N'_{3j}\,\mathrm{d}z - \rho\frac{K_{xy\omega}}{R}\int_l \left(N'_{3i}N'_{4j} + N'_{4i}N'_{3j} + \frac{2}{R}N'_{3i}N'_{3j}\right)\mathrm{d}z$$

$$+ \rho I_\omega \int_l \left(N'_{4i}N'_{4j} + \frac{N'_{4i}N'_{3j} + N'_{3i}N'_{4j}}{R} + \frac{N'_{3i}N'_{3j}}{R^2}\right)\mathrm{d}z$$

(37)

Chapter 4

Mathematical Calculations:

Mathematical Calculations:

4.1 Straight beam:

the geometric properties of the section are given below.

Geometric properties of numerical models;

Name	A (in.^{2})	I_x (in.^{4})	I_y (in.^{4})	K_T (in.^{4})	I_w (in.^{6})	E (ksi)	G (ksi)	ρ (ks^2/in^4)
Section A	66.309	34086	714.94	37.043	474960	29,000	11,154	7.35×10^{-7}

Table 1

The total centre line length between two girders is 1200inch and the girder spacing Gs is 106 inch. The geometric properties of the section are given in table 1.

From the table 1 and equations (12)-(13e) the obtained mass matrix and stiffness matrix are following

4.1.1 Stiffness matrix:

The calculation of stiffness matrix of the element has been explained in equations 12-13e. The obtained element stiffness matrix is as follows.

0.144	86.39	-0.144	86.39	0	0	0	0	0	0	0	0	0	0
86.39	69111	-86.39	34555	0	0	0	0	0	0	0	0	0	0
-0.144	-86.39	0.144	-86.39	0	0	0	0	0	0	0	0	0	0
86.39	34555	-86.39	69111	0	0	0	0	0	0	0	0	0	0
0	0	0	0	6.865	-4119	-6.865	-4119	0	0	0	0	0	0
0	0	0	0	-4119	3E+06	4119	2E+05	0	0	0	0	0	0
0	0	0	0	-6.865	4119	6.865	4119	0	0	0	0	0	0
0	0	0	0	-4119	2E+06	4119	3E+06	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1602	-1602	0	0	0	0
0	0	0	0	0	0	0	0	-1602	1602	0	0	0	0
0	0	0	0	0	0	0	0	0	0	508.8	-98709	-508.8	-98709
0	0	0	0	0	0	0	0	0	0	-98709	91825760	98709	11478160
0	0	0	0	0	0	0	0	0	0	-508.8	98709	508.8	98709
0	0	0	0	0	0	0	0	0	0	-98709	11478160	98709	91825760

4.1.2 Mass matrix:

The calculation of mass matrix of the element has been explained in equations 14-15e.The obtained element mass matrix is as follows.

0.02	3.68	0.01	-2.17	0	0	0	0	0	0	0	0	0	0
3.68	802	2.17	-602	0	0	0	0	0	0	0	0	0	0
0.01	2.17	0.02	3.68	0	0	0	0	0	0	0	0	0	0
-2.17	-602	3.68	802	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0.02	-3.68	0.01	2.27	0	0	0	0	0	0
0	0	0	0	-3.68	806	-2.17	-603	0	0	0	0	0	0
0	0	0	0	0.01	-2.17	0.02	3.68	0	0	0	0	0	0
0	0	0	0	2.17	-603	3.68	806	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0.02	0.01	0	0	0	0
0	0	0	0	0	0	0	0	0.01	0.02	0	0	0	0
0	0	0	0	0	0	0	0	0	0	11.4	-1929	3.95	1140
0	0	0	0	0	0	0	0	0	0	-1929	421008	-1140	-315728
0	0	0	0	0	0	0	0	0	0	3.95	-1140	11.4	1929
0	0	0	0	0	0	0	0	0	0	1140	315728	1929	421008

4.2 Horizontally curved steel-I girder:

4.2.1 Curved beam with subtended angle 90^0

Using equations (28a)-(37) and table 1 the different values obtained are following:

f = 1.000018
α1= 107.843468
α2= 1.999075
α3= 24282.43594
L= 1200inch
R = 763.94 inch
Where all notations are defined in (30a)-(30e)

4.2.1.1 In-plane mass and stiffness matrices:

[Ain] =

0	0	-1	1	0	1
1	1	0	0	0	0
0	0	0	0.002618	0	0.001309
1.5708	0.02741	-0.99962	1.0427	-1.5428	1
1	0.99962	0.02741	1.5702	0.043056	0
0.002056	0	0	0.002617	7.18E-05	0.001309

$[Ain]^{-1}$ =

-0.02185	-0.0003	-469.93	0.02185	-0.0003	469.93
0.021849	1.0003	469.93	-0.02185	0.000298	-469.93
-1.0003	0.64792	764.28	-0.01746	-0.64769	13.226
0.000299	-0.64792	-0.34189	0.01746	0.64769	-13.226
0.62608	-0.0003	-469.94	-0.62584	0.017461	469.75
-0.0006	1.2958	764.63	-0.03492	-1.2954	26.451

Where Ain is defined in (31a)

Shape functions for in plane bending:

The calculation of shape functions are described in equations (28a) to (31e).Using matlab ,the obtained shape functions are following:

n11=-.02185+.021849*c-1.0003*s+.000299*e*c+.62608*e*s

n12=-.0003+1.003*c+.64792*s-.64792*e*c-.0003*e*c

n13=-469.93+469.93*c+764.28*s-.31489*e*c-469.4*e*s

n14=-.02185-.02185*c-.01746*s+.01746*e*c-62854*e*s

n15=-.003+.000298*c-.4769*s+.64769*e*c+.017461*e*s

n16=469.93-469.93*c+13.226*s-13.226*e*c+469.75*e*s

n21=-.02185*f*e+.021849*s+1.003*c+.000299*(c+s*e)+.62608*(s-e*c)-.0006

n22=-.003*f*e+1.003*s-.64792*c-.6479*(c+s*e)-.003*(s-e*c)+1.2958

n23=-469.93*f*e+469.93*s-764.28*c-.34189*(c+s*e)-469.4*(s-e*c)+764.43

n24=.02185*f*e-.02185*s+.01746*c+.01746*(c+s*e)-6.2584*(s-e*c)-.0349

n25=-.0003*f*e+.000298*s+.64769*c+.64769*(c+s*e)+.017461*(s-e*c)-1.2954

n26=469.93*f*e-469.93*s-13.226*c-13.226*(c+s*e)+469.75*(s-e*c)+26.45

where nij are the elements of shape function matrix:(33)

and
e=(z/r);
c=cos (z/r),
s= sin (z/r);

Stiffness matrix:

The calculation of stiffness matrix is described in equation (34).using the matlab,the obtained element stiffness matrix is as follows.

0.0691	0.0716	-68.422	-357940	0.8585	69.2857
0.0716	0.1503	-56.245	-905300	2.1163	57.7316
-68.4222	-56.2545	89300	4150700	62.8292	-90522
-375940	-905300	4150700	1.02E+13	-22990000	-57999000
0.8585	2.1163	62.8292	-22990000	55.4333	-59.6592
69.2857	57.7316	-90522	-5799900	-59.6592	91771

Mass matrix:

The calculation of mass matrix is described in equation (36).The obtained value of element mass matrix using mass matrix is as follows.

0.0433	0.0181	2.509	538.0285	-0.0017	1.561
0.0181	0.0758	5.7261	-1675.1	-0.0055	32.12
2.509	5.7261	2462	258700	1.1533	-2081.2
538.023	-1675.1	258700	1.53E+08	3.76E+02	1.83E+06
-0.0017	-0.0055	1.1533	3.76E+02	0.0018	-6.0722
1.561	32.12	-2081.2	1825300	-6.0722	26911

4.2.1.2 Out of plane bending:

[Aout] =

1	0	1	0	1	0	0	0
0	-0.00131	0	-0.417	0	-0.00131	0	0
0	0	31.786	0	-0.00131	0	0	0.002617
0	-1.71E-06	0	-4.4873	0	0	3.43E-06	0
1	1.5708	9.6422	9.5902	0.99962	0.02741	1.5702	0.043056
0	-0.00131	-1.3538	-1.3617	3.59E-05	-0.00131	-0.00125	-0.00209
0	0	306.48	304.83	-0.00131	-3.59E-05	-0.00213	0.002599
0	-1.71E-06	-43.034	-43.267	0	0	3.42E-06	9.39E-08

$[Aout]^{-1}$ =

0.99908	-975.67	1394.6	-9395.2	0.000918	993.38	-604.69	-3307.7
-0.64358	-9.7487	-478.46	3054.9	0.64358	10.104	479.09	3058.4
-2.18E-07	-8.99E-06	-0.00049	0.22722	2.18E-07	7.93E-09	0.000495	-0.02008
2.42E-07	1.12E-05	0.000548	-0.22635	-2.42E-07	-1.21E-06	-0.00055	-0.0035
0.000918	975.67	-1394.6	9395	-0.00092	-993.38	604.69	3307.7
0.6435	-753.61	478.28	-2982.8	-0.6435	-10.104	-478.92	-3057.3
-0.00435	9.7477	478.45	-3054.9	0.004353	3.4641	-478.9	-3057.4
0.003112	488.54	-309.98	1943.3	-0.00311	-497.3	296.7	1899.8

Where Aout is defined in (31b).

4.2.2. Curved beam with subtended angle 10^0:

Using equations (28a)-(37) and table 1 the different values obtained are following:

f = 1.0000
$\alpha 1$= 970.596008
$\alpha 2$= 1.999163
$\alpha 3$= 1966853.75
L= 1200inch
R = 6875.49 inch
Where all notations are defined in (30a)-(30e).

4.2.2.1 Inpalne bending:
Inverse of Ain is used for the calculation of shape functions:

[Ain] =

0	0	-1	1	0	1
1	1	0	0	0	0
0	0	0	0.000291	0	0.000145
0.1745	0.00346	-0.99994	1.0005	-0.17145	1
1	0.99994	0.003046	0.17445	0.000532	0
2.50E-05	0	0	0.000291	1.00E-06	0.000145

$[(Ain]^{-1}$ =

-0.22434	-0.00016	-38465	0.22435	-0.00062	38465
0.22434	1.0002	38465	-0.22435	0.000617	-38465
-1.0004	5.8748	6887.6	-0.01746	-5.8751	132.36
0.000436	-5.8346	8.8789	0.017338	5.8348	-131.46
5.6084	0.003973	-38370	-5.6086	0.015434	38372
-0.00087	11.709	6878.7	-0.0348	-11.71	263.82

Where Ain is defined in equation (31a)

Shape functions for angle 10 inplane:

The calculation of shape function are described in equations (28a) to (31e).Using matlab ,the obtained shape functions are following:

n11=-.22434+.22434*c-1.0004*s+.000436*e*c+5.6084*e*s

n12=-.00016+1.0002*c+5.8748*s-5.8346*e*c+.003973*e*s

n13=-38465+38465*c+6887.6*s+8.8789*e*c-38370*e*s

n14=.22435-.22435*c-.017465*s+.017338*e*c-5.6086*e*s

n15=-.00062+.000617*c-5.8751*s+5.8348*e*c+.015434*e*s

n16=38465-38465*c+132.36*s-131.46*e*c+38372*e*s

n21=-.22434*f*e+.22434*s+1.0004*c+.000436*(c+e*s)+5.6084*(s-e*c)-.00087

n22=-.00016*f*e+1.0002*s-5.8748*c-5.8346*(c+e*s)+.003973*(s-e*c)+11.709

n23=-38465*e*f+38465*s-6887.6*c+8.8789*(c+e*s)-38370*(s-e*c)+6878.1

n24=.22435*f*e-.22435*s+.01746*c+.017338*(c*e*s)-5.6086*(s-e*c)-.0348

n25=-.00062*e*f+.000617*s+5.8751*c+5.8348*(c+e*s)+.015434*(s-e*c)-11.71

n26=38465*e*f-38465*s-132.36*c-131.46*(c+e*s)+388372*(s-e*c)+263.82

where nij are the elements of shape function matrix.(33).

And

e=(z/r);
c=cos (z/r),
s= sin (z/r);

Stiffness matrix:

The calculation of stiffness matrix is described in equation (34).using the matlab,the obtained elemnt stiffness matrix element is as follows..

0.0013	1.24E-04	-13.9794	-0.0013	-1.20E-04	13.9825
1.24E-04	1.55E-05	-1.3602	-1.24E-04	-1.50E-05	1.3065
-13.9794	-1.3602	146780	13.9839	1.2578	-146810
-0.0013	-1.24E-04	13.9839	0.0015	1.20E-04	361.3915
-1.20E-04	-1.50E-05	1.2578	1.20E-04	1.46E-05	-1.2582
13.9825	1.3065	-146810	361.3915	-1.2582	1.10E+09

Mass matrix:

The calculation of mass matrix is described in equation (36).The obtained value of element mass matrix element using matlab is as follows.

0.0582	0.0032	0.4089	-0.001	-1.97E-05	9.456
0.0032	0.0592	1.0419	-0.0034	-3.51E-04	35.6459
0.4089	1.0419	2497.5	0.4766	0.0449	-4992.9
-0.001	-0.0034	0.47766	3.49E-04	3.35E-05	-3.6551
-1.97E-05	-3.51E-04	0.0449	3.35E-05	3.37E-06	-0.3536
9.456	35.6549	-4992.9	-3.6551	-0.3536	40157

4.2.2.2 Out of plane bending:

[Aout]=

1	0	1	0	1	0	0	0
0	-0.00015	0	-0.14117	0	-0.00015	-0.00015	0
0	0	286.07	0	-0.00015	0	0	0.000291
0	-2.17E-10	0	-40.383	0	0	4.23E-08	0
1	0.1745	9.6369	9.5849	0.99994	0.003046	0.1745	0.000532
0	-0.00015	-1.3531	-1.3604	4.43E-07	-0.00015	-0.00015	-2.60E-05
0	0	2756.8	2741.9	-0.00015	-4.43E-07	-2.60E-05	0.000291
0	-2.17E-10	-387.07	-389.17	0	0	4.23E-08	1.29E-10

$[Aout]^{-1}$:

0.99986	-79921	7216.8	-51064	0.000138	79921	-76.107	4512.2
-5.7316	-97.13	-39283	2.51E+05	5.7316	97.522	39281	2.51E+05
6.74E-11	-1.02E-07	-3.73E-05	0.025135	-6.74E-11	-1.32E-08	3.73E-05	-0.00235
-7.48E-11	1.28E-07	4.13E-05	-0.02503	7.48E-11	-9.81E-11	-4.13E-05	-0.00026
0.000138	79921	-7216.8	51064	-0.00014	-79921	76.107	-4512.2
5.8325	-6921.1	1.6639	12.029	-5.8325	-97.929	0.24925	3.2171
-0.10086	121.72	39281	-250750	0.10086	0.40671	-39281	-250740
2.35E-06	39823	-122.96	735.61	-2.35E-06	-39823	1.2967	57.568

Where Aout is defined in 31(b).

Chapter 5

Assembling of finite elements

Assembling of finite elements

5.1 Co-ordinate transformation:

To assemble the total structure stiffness matrix and the total structure mass matrix of a horizontally curved steel I-girder bridge, the Cartesian coordinate system is used as the structural coordinate system. The concept is simple to derive the transformation matrix. It is to position a horizontally curved steel I-girder bridge only in the XZ-plane. Therefore, the y-axis is parallel to the Y-axis and the transformation consists of a rotation through the angle θ about the Y-axis. For the curved beam element, the components of the transformation matrix at the node i and at the node j are independently composed because the angles are not equal at the nodes.

5.2 Free vibration analysis:

A finite element analysis program is developed to achieve free vibration analyses of various horizontally curved steel I-girder bridges using traditional eigenvalue solution techniques and coded using MATLAB and C++. Numerical models are basically composed of two main girders and thirteen cross-frames as shown in . Length of the centerline between two main girders, Lc, is 1200 in. Girder spacing, Gs, is 106 in. The geometric properties are given in Table 1.

Chapter 6

Computer programs

Computer programs

The following two programs have been used for calculating natural frequencies. The program 7.1 is used to assemble the different finite elements. After this the boundries conditions are applied and program 6.1 is used for finding out the natural frequency.

6.1 Program for assembling the finite elements:

The following c++ program is used to assemble the finite elements:

```
#include<iostream.h>

#include<stdio.h>

#include<conio.h>

#include<math.h>

void assemb(void)

{

        // ********************************************************

        // ******* Formation and Assembling of the Element *******

        // *********************Stiffness Matrix *****************

        // *************** and Load vector to ********************

        // ******** Global stiffness matrix and load vector ******

        // *********************************************************

        int   l, li;

        gload = dvector(1,msvab);

        memory << "\ngload is ALLOCATED\n";

        gstif = dmatrix(1,msvab,1,hband);

        memory << "\ngstif is ALLOCATED\n";

        value = dmatrix(1,nnode,1,ndime);

        memory << "\nvalue is ALLOCATED\n";

        elcod = dmatrix(1,12,1,ndime);

        memory << "\nelcod is ALLOCATED\n";
```

```
	elxy = dmatrix(1,12,1,ndime);
	memory << "\nelxy is ALLOCATED\n";
	bmatx = dmatrix(1,3,1,nevab);
	memory << "\nbmatx is ALLOCATED\n";

	for(int isvab=1; isvab<=msvab; isvab++) {
		gload[isvab] = 0.0;
		for(int iband=1; iband<=hband; iband++) {
			gstif[isvab][iband] = 0.0;
		}
	}

	//***** Loop on the number of ELEMENTS to compute element
	//***** matrices and their assembly begins here
	cout << "\n\n		Now Processing the Elements\n";
	cout << "			Total No. of Elements : " << nelem << endl;
	for(ielem=1; ielem<=nelem; ielem++) {
		cout << "		Element No. " << setw(4) << ielem;
		for(int loop=1; loop<=28; loop++) {
			cout << "\b";
		}
		for(int inode=1; inode<=12; inode++) {
			lnode = mnods[ielem][inode];
			elcod[inode][1] = coord[lnode][1];
			elcod[inode][2] = coord[lnode][2];
			elxy[inode][1] = xynod[lnode][1];
			elxy[inode][2] = xynod[lnode][2];
			cout << "";
		}
//		a = fabs(elcod[4][1]-elcod[7][1])/2.0;
//		b = fabs(elcod[1][2]-elcod[4][2])/2.0;
```

```
a = 1.0;
b = 1.0;
stifpb();
loadpb();
//***** Assembling of the element matrices to global matrices
for(inode=1; inode<=nnode; inode++) {
    for(int jnode=1; jnode<=nnode; jnode++) {
        int iii = lnods[ielem][inode];
        int jjj = lnods[ielem][jnode];
        int ii1 = iii-1;
        int jj1 = jjj-1;
        int m = ndofn*ii1;
        int n = ndofn*jj1;
        int mi = ndofn*(inode-1);
        int nj = ndofn*(jnode-1);
        for(int k=1; k<=ndofn; k++) {
            for(int l=1; l<=ndofn; l++) {
                int mmi=m+k;
                int nnj = n+l-(mmi-1);
//              int nnj = n+l;    //****for global matrix in
full
                if(nnj > 0) {
                    int imi=mi+k;
                    int jnj=nj+l;
                    gstif[mmi][nnj] += stif[imi][jnj];
                }
            }
        }
    }
}
for(inode=1;inode<=nnode;inode++)  {
```

```
                    int iii = lnods[ielem][inode]        ;
                    int ii1 = iii - 1           ;
                    int m   = ndofn * ii1       ;
                    int mi  = ndofn *(inode - 1 ) ;
                    for(int k=1;k<=ndofn;k++) {
                            int mmi = m  + k    ;
                            int imi = mi + k    ;
                            gload[mmi] += eload[imi];
                    }
            }
            free_dvector(shape,1,nevab);
            memory << "\nshape is FREED \n";
            free_dmatrix(estif,1,nevab,1,nevab);
            memory << "\nestif is FREED \n";
//          free_dmatrix(bmatx,1,3,1,nevab);
            free_dmatrix(btrans,1,nevab,1,3);
            memory << "\nbtrans is FREED \n";
            free_dmatrix(estif1,1,nevab,1,3);
            memory << "\nestif1 is FREED \n";
            free_dmatrix(stif,1,nevab,1,nevab);
            memory << "\nstif is FREED \n";
            free_dvector(eload,1,nevab);
            memory << "\neload is FREED \n";
    }
    free_dmatrix(coord,1,npoin,1,ndime);
    memory << "\ncoord is FREED \n";
    free_dvector(props,1,nprop);
    memory << "\nprops is FREED \n";
    free_dmatrix(value,1,nnode,1,ndime);
    memory << "\nvalue is FREED \n";
    free_dmatrix(elcod,1,12,1,ndime);
```

```
        memory << "\nelcod is FREED \n";
        free_dmatrix(bcord,1,bpoin,1,ndime);
        memory << "\nbcord is FREED \n";
        if(qload == 0.0) {
                cout << "Give the conc. load value : ";
                cin >> qload;
                cout << "Give the load position in gload[] matrix :";
                int lposn;
                cin >> lposn;
                gload[lposn] = qload;
        }
}
```

6.2 Program for eigenvalues and eigenvectors:

```
/*   Program for natural frequency.cpp

   This program is used to solve the general eigenvalue problem [k]x = (w^2)[m]x
        the program is converted into a special eigenvalue problem [D]Y = (1/w^2)[I]Y
        by generating the matrix [D] using the relation
        [D] = (inverse (transpose of [U]))*[m]*(inverse ([U]))

   BK         = Array of size N by N, containing the matrix [k].
        BM         = Array of size N by N, containing the matrix [m].
        ND         = Order of the matrices [k] and [m].
        U, UI, UTI   = Array of size N by N, indication the matrices [U], U inverse and
transpose of U inverse

        BMU, UMU, EV = Arrays of size N by N each.
        XF           = Array of size N by N. The computed eigenvectors X(i) are stored
columnwise in XF

   Example: [k] = {[2, -1, 0],[-1, 2, -1],[0, -1, 1]} and
            [m] = {[1, 0, 0],[0, 1, 0],[0, 0, 1]}
*/
```

```
#include <iostream.h>
#include <stdlib.h>
#include <iomanip.h>
#include <math.h>


void decomp(double **, double **, int);
void matmul(double **, double **, double **, int, int, int);
void jacobi(double **, int, double **, double, int);


int main()
{
        int i, j, k, ii, ip, nd;
        double sum;
        double **bk, **bm, **u, **ui, **uti, **bmu, **umu, **xf, **ev;

        cout << "Please input ND:" << endl;
        cin >> nd;

   bk = new double *[nd];
        bm = new double *[nd];
   u = new double *[nd];
        ui = new double *[nd];
   uti = new double *[nd];
        bmu = new double *[nd];
   umu = new double *[nd];
        xf = new double *[nd];
   ev = new double *[nd];

        for (i = 0; i < nd; i ++)
        {
                bk[i] = new double[nd];
                bm[i] = new double[nd];
                u[i] = new double[nd];
                ui[i] = new double[nd];
                uti[i] = new double[nd];
                bmu[i] = new double[nd];
                umu[i] = new double[nd];
                xf[i] = new double[nd];
                ev[i] = new double[nd];

        }


        cout << "Please input BK matrix row by row:" << endl;
```

```
    for (i = 0; i < nd; i ++)
    {
            for (j = 0; j < nd; j ++)
                    cin >> bk[i][j];
    }

    cout << "Please input BM matrix row by row:" << endl;
    for (i = 0; i < nd; i ++)
    {
            for (j = 0; j < nd; j ++)
                    cin >> bm[i][j];
    }

    decomp(bk, u, nd);
    cout << endl << "UPPER TRIANGULAR MATRIX [U]:" << endl << endl;
    for (i = 0; i < nd; i ++)
    {
            for (j = 0; j < nd; j++)
                    cout << setprecision(8) << setiosflags(ios::fixed | ios::showpoint)
                        << setw(16) << setiosflags(ios::right) << u[i][j];
            cout << endl;

    }

    for (i = 0; i < nd; i ++)
    {
            for (j = 0; j < nd; j ++)
                    ui[i][j] = 0.0;
    }

    for (i = 0; i < nd; i ++)
            ui[i][i] = 1.0 / u[i][i];

    for (j = 0; j < nd; j ++)
    {
for (ii = 0; ii < nd; ii ++)
            {
                    i = nd - ii - 1;
                    if (i < j)
                    {
                            ip = i + 1;
                            sum = 0.0;
                            for (k = ip; k <= j; k ++)
                                    sum = sum + u[i][k] * ui[k][j];
                            ui[i][j] = - sum / u[i][i];
                    }
```

```
		}

	}

	cout << endl << "INVERSE OF THE UPPER TRIANGULAR MATRIX, [UI],"
<< endl << endl;
	for (i = 0; i < nd; i ++)
	{
		for (j = 0; j < nd; j++)
			cout << setprecision(8) << setiosflags(ios::fixed | ios::showpoint)
			    << setw(16) << setiosflags(ios::right) << ui[i][j];
		cout << endl;

	}

	for (i = 0; i < nd; i ++)
	{
		for (j = 0; j < nd; j ++)
			uti[i][j] = ui[j][i];
	}

	matmul(bmu, bm,ui, nd, nd, nd);
	matmul(umu, uti, bmu, nd, nd, nd);

	cout << endl << "MATRIX [UMU] = [UTI][M][UI]:" << endl << endl;
	for (i = 0; i < nd; i ++)
	{
		for (j = 0; j < nd; j++)
			cout << setprecision(8) << setiosflags(ios::fixed | ios::showpoint)
			    << setw(16) << setiosflags(ios::right) << umu[i][j];
		cout << endl;

	}

	jacobi(umu, nd, ev, 1.0e-5, 200);

	cout << endl << "EIGENVALUES:" << endl << endl;
	for (i = 0; i < nd; i ++)
			cout << setprecision(8) << setiosflags(ios::fixed | ios::showpoint)
			    << setw(16) << setiosflags(ios::right) << umu[i][i];

	cout << endl << endl <<"EIGENVECTORS (COLUMNWISE):" << endl <<
endl;

	matmul(xf, ui, ev, nd, nd, nd);
	for (i = 0; i < nd; i ++)
```

```
        {
                for (j = 0; j < nd; j++)
                        cout << setprecision(8) << setiosflags(ios::fixed | ios::showpoint)
                            << setw(16) << setiosflags(ios::right) << xf[i][j];
                cout << endl;

        }

        for (i = 0; i < nd; i ++ )
        {
                delete []bk[i];
                delete []bm[i];
                delete []u[i];
                delete []ui[i];
                delete []uti[i];
                delete []bmu[i];
                delete []umu[i];
                delete []xf[i];
                delete []ev[i];
        }

                delete []bk;
                delete []bm;
                delete []u;
                delete []ui;
                delete []uti;
                delete []bmu;
                delete []umu;
                delete []xf;
                delete []ev;

    return 0;





}

void matmul(double **a, double **b, double **c, int l, int m, int n)
{

        int i, j, k;
        for (i = 0; i < l; i ++)
        {
```

```
		for (j = 0; j < n; j++)
		{
			a[i][j] = 0.0;
			for (k = 0; k < m; k ++)
				a[i][j] = a[i][j] + b[i][k]*c[k][j];
		}
	}
	return;
}

void decomp(double **a, double **u, int n)
{
	int i, j, k, im;
	double sum;

	for (i = 0; i < n; i ++)
	{
		for (j = 0; j < n; j ++)
			u[i][j] = 0.0;
	}

	u[0][0] = sqrt(a[0][0]);
	for (j = 1; j < n; j++)
		u[0][j] = a[0][j] / u[0][0];
	for (i = 1; i < n; i ++)
	{
		im = i - 1;
		 sum = 0.0;
		 for (k = 0; k <= im; k ++ )
			sum = sum + u[k][i] * u[k][i];
		 u[i][i] = sqrt(a[i][i] - sum);
		 j = i + 1;
		 if (j < n)
		 {
			sum = 0.0;
		    for (k = 0; k <= im; k ++)
			    sum = sum + u[k][i] * u[k][j];
		    u[i][j] = (a[i][j] - sum) / u[i][i];
		 }

	}

	return;
}
```

```
void jacobi(double **d, int n, double **e, double eps, int itmax)
{
	int i, j;
	int iter, ir, ic, nm1, ip1;
	double zz, yy, dif, tanz, sinz, cosz, zzz, yyy;

	iter = 0;
	for (i = 0; i < n; i ++)
	{
		for (j = 0; j < n; j ++)
		{
			e[i][j] = 0.0;
			e[i][i] = 1.0;
		}

	}

	while (iter < itmax)
	{
		zz = 0.0;
		nm1 = n-1;
		for (i = 0; i < nm1; i ++)
		{
			ip1 = i + 1;
			for (j = ip1; j < n; j ++)
			{
				if (fabs(d[i][j]) > zz)
				{
					zz = fabs(d[i][j]);
					ir = i;
					ic = j;
				}
			}
		}

		if (iter == 0)
			yy = zz * eps;
		if (zz <= yy)
			return;
		else
		{
			dif = d[ir][ir] - d[ic][ic];
			tanz = (- dif + sqrt (dif * dif + 4.0 * zz * zz)) / (2.0 * d[ir][ic]);
			cosz = 1.0 / sqrt(1.0 + tanz * tanz);
			sinz = cosz * tanz;
		   for (i = 0; i < n; i ++)
```

```
				{
					zzz = e[i][ir];
					e[i][ir] = cosz * zzz + sinz * e[i][ic];
					e[i][ic] = cosz * e[i][ic] - sinz * zzz;
				}

				i = 0;
				while (i != ir)
				{
					yyy = d[i][ir];
					d[i][ir] = cosz * yyy + sinz * d[i][ic];
					d[i][ic] = cosz * d[i][ic] - sinz * yyy;
					i = i +1;
				}

				i = ir + 1;
				while (i != ic)
				{
					yyy = d[ir][i];
					d[ir][i] = cosz * yyy + sinz * d[i][ic];
					d[i][ic] = cosz * d[i][ic] - sinz * yyy;
					i = i +1;
				}
				i = ic + 1;
				while (i < n)
				{
					zzz = d[ir][i];
					d[ir][i] = cosz * zzz + sinz * d[ic][i];
					d[ic][i] = cosz * d[ic][i] - sinz * zzz;
					i = i + 1;
				}
				yyy = d[ir][ir];
				d[ir][ir] = yyy * cosz * cosz + d[ir][ic] * 2.0 * cosz * sinz +
d[ic][ic]
						* sinz * sinz;
				d[ic][ic] = d[ic][ic] * cosz * cosz + yyy * sinz * sinz - d[ir][ic] *
2.0 * cosz
						* sinz;
				d[ir][ic] = 0.0;
				iter = iter + 1;

			}
		}
		return;

}
```

Chapter 7

Results

Results

The results obtained from the computer program are following:

7.1 Problem 1.

Straight beam :

The eigenvalues are : $3.8998*10^{-3}$ and $4.012627*10^{-4}$

From the obtained eigenvalues the natural frequency can be calculated using the relation (eigenvalue=(1/(w^2)).

Natural frequency in bending

- **1st - 16.01rad/sec**
- **2nd - 49.92 rad /sec**

7.2 Curved beam: In-plane frequencies:

Problem 2:

Curved beam with subtended angle 10^0 .

The obtained output using matlab is as follows:

Upper triangular matrix [U]:

3.605551e-002	3.447184e-003	-3.877188e+002
0.000000e+000	1.895500e-003	-1.248316e+001
0.000000e+000	0.000000e+000	0.000000e+000
0.000000e+000	0.000000e+000	0.000000e+000
0.000000e+000	0.000000e+000	0.000000e+000
0.000000e+000	0.000000e+000	0.000000e+000

Inverse of the upper triangular matrix:

2.773501e+001	-5.043930e+001	0.000000e+000
0.000000e+000	5.275653e+002	0.000000e+000
0.000000e+000	0.000000e+000	0.000000e+000
0.000000e+000	0.000000e+000	0.000000e+000
0.000000e+000	0.000000e+000	0.000000e+000

0.000000e+000	0.000000e+000	0.000000e+000

Matrix [UMU]=[UTI][M][UI]:

4.476923e+001	-3.459550e+001	0.000000e+000
-3.459550e+001	1.645461e+004	0.000000e+000
0.000000e+000	0.000000e+000	-2.426916e+003
1.136328e+002	-1.949184e+002	0.000000e+000
7.768547e+001	-1.568160e+002	0.000000e+000
-5.510598e-001	1.028997e+000	0.000000e+000

Warning: Function call jacobi invokes inexact match P:\programm11\JACOBI.M.

Eigenvalues:

4.842e-003	5.284e-004

Eigenvectors (Columnwise):

-2.382186e+001	-6.762361e+001	-1.315810e+001
5.530633e+002	2.540189e+001	-2.394397e+001
2.755439e-003	2.380356e-003	-2.149277e-003
-3.600952e-001	-5.441682e+001	2.268442e+001
-2.553204e+000	-2.824755e+002	1.169774e+002
8.359799e-010	1.275246e-007	1.330885e-007

From the obtained eigenvalues the different modes of natural frequency are as follows:

- **1^{st} –14.42 rad/sec**
- **2^{nd}- 75.78 rad/sec**

Problem 3:

Curved beam with subtended angle 90^0.

The obtained output using matlab is as follows:

Upper triangular matrix [U]:

2.628688e-001	2.723792e-001	-2.602896e+002
0.000000e+000	2.758796e-001	5.311183e+001
0.000000e+000	0.000000e+000	1.368520e+002
0.000000e+000	0.000000e+000	0.000000e+000
0.000000e+000	0.000000e+000	0.000000e+000
0.000000e+000	0.000000e+000	0.000000e+000

Inverse of the upper triangular matrix:

3.804179e+000	-3.755912e+000	8.693122e+000
0.000000e+000	3.624770e+000	-1.406761e+000
0.000000e+000	0.000000e+000	7.307162e-003
0.000000e+000	0.000000e+000	0.000000e+000
0.000000e+000	0.000000e+000	0.000000e+000
0.000000e+000	0.000000e+000	0.000000e+000

Matrix [UMU]=[UTI][M][UI]:

6.266281e-001	-3.690916e-001	1.404821e+000
-3.690916e-001	1.113922e+000	-1.051507e+000
1.404821e+000	-1.051507e+000	3.311997e+000
3.295083e+000	-2.370481e+000	7.698406e+000
0.000000e+000	0.000000e+000	0.000000e+000
-2.595701e+000	2.433861e+000	-6.340982e+000

Eigenvalues:

2.960e-002 1.741e-003

Eigenvectors(Columnwise):

3.132032e+001	-1.369642e+001	2.986830e+000
-4.024532e+000	2.629069e+000	-2.243486e+000
1.967471e-002	-1.139783e-002	-1.387925e-003
9.871925e-007	-2.185781e-007	8.151047e-009
3.199306e-002	-1.991456e-002	1.775388e-003
-2.626239e-003	-1.872539e-003	5.907330e-004

From the obtained eigenvalues the different modes of natural frequency are as follows:

- **1^{st} - 5.81 rad/sec**
- **2^{nd} - 23.96 rad/sec**.

All the symbols used above are explained in program 6.2

7.3 Discussion:

From the above results it can be observed that the natural frequencies decreases as subtended angle increases . Horizontally curved steel I-girder bridges possess numerous beneficial merits such as reduction in total construction cost and time. In addition, they can be constructed in continuous form unlike the case of straight girders, which makes the bridges aesthetically pleasing as well as more economical with the reduced girder depth.

From the above results we can say that curved beams are always better option than straight beams for bridge construction.

Chapter 8

Conclusion

Concluding remarks:

In this paper, the finite element formulation of the curved beam is presented based on the Kang and Yoo's thin-walled curved beam theory. The equations of motion of the Straight beam are derived simply letting the radius of curvature approach to infinity in Those of the curved beam. In this process, the validity of the equations of motion of the curved beam is demonstrated. To achieve free vibration analyses of various horizontally curved steel I-girder bridges, the stiffness and the mass matrix of the curved and the straight beam element including the warping degree of freedom are composed and a corresponding computer program is developed.

Since the out-of-plane behavior is coupled with the bending and the torsional behavior in the horizontally curved steel I-girder bridges, the warping rigidity of a beam is significant.

From the obtained results, some characteristics of the horizontally curved steel I-girder bridges are revealed.

(1). The natural frequency of the bridge tends to decrease as the subtended angle becomes larger .

(2). In the in-plane behavior, the natural frequency is affected with the constraint direction.

References:

[1] Bathe KJ. Finite element procedure. Englewood Cliffs, NJ: Prentice-Hall; 1996.

[2] Gupta KK. Eigenproblem solution by a combined sturm sequence and inverse iteration technique. J Numer Meth Eng 1973;7:17–42.

[3] Kang YJ, Yoo CH. Thin-walled curved beams. I: formulation of nonlinear equations. J Eng Mech Div 1994; 120(EM10):2072–101.

[4] Kang YJ, Yoo CH. Thin-walled curved beams. II: analytical solutions for buckling of arches. J Eng Mech Div 1994;120(EM10):2102–25.

[5] Kang YJ, Yu CS, Jung JH, Hwang MO. Finite element discretization of a thin-walled curved beams. J Kor Soc Steel Construct 1995;7(2):215–24.

[6] Nakai H, Yoo CH. Analysis and design of curved steel bridge. New York: McGaw-Hill; 1988.

[7] Yoon KY, Kang YJ. Free vibration analysis of horizontally curved I-girder bridges using the finite element method. J Kor Soc Steel Construct 1998;10(1):47–61. K.-Y. Yoon et al. / Thin-Walled Structures 43 (2005) 679–699 699.

[8].Mechanical vibration –S.S.Rao.