PDHonline Course S274 (6 PDH)

The Direct Method in Steel

Instructor: Marvin Liebler, P.E.

2013

PDH Online | PDH Center

5272 Meadow Estates Drive
Fairfax, VA 22030-6658
Phone & Fax: 703-988-0088
www.PDHonline.org
www.PDHcenter.com

An Approved Continuing Education Provider

1. INTRODUCTION

The direct method in steel addresses the design of steel structures for stability. It is defined in Chapter C of the 2010 AISC steel code as consideration of the following :
(1) flexural, shear, and axial deformations
(2) connection deformations
(3) P-Δ and P-δ effects
(4) geometric imperfections
(5) inelasticity
(6) uncertainty in stiffness and strength
Items (1) through (3) are covered by a second-order elastic analysis, (4) and (5) by approximations, and (6) by LRFD.

The arrangement of this paper is taken in order of the design process after a plane frame has been selected. Determination of loads is briefly covered in section 2, and load factors assigned in section 3.

The backbone of the paper is the second-order elastic analysis of a plane frame. It is the author's opinion that knowledge of this process is best obtained by initially studying first-order analysis, including generation of basic beam stiffness matrix entries, fixed end forces, shear effects on moment and transverse displacement, and semi-rigid (partially-restrained) end connections. These topics are covered in section 4, leading to creation of a full first-order structure analysis program, including generation of all beam matrices in local coordinates, conversion to global coordinates, assembly and subsequent inversion of the global stiffness matrix, solution of displacements, and calculation of member forces.

The only new concepts in second-order elastic analysis are now stability functions and iteration of structural calculations, which are described in section 5. Output of these functions are compared with AISC benchmarks.

Section 6 concludes the paper with a fourth benchmark example, notional loads and reduction of section moduli to meet points (4) and (5) above.

References and three appendices, although not part of the subject matter, are provided for those wishing to pursue the subject further. Note that in 2nd order analysis, each load combination must be separately analyzed, superposition of load cases may not be used.

2. LOADS

Loads to be resisted are usually set down in the governing building code where the structure is to be located. The loads generally include vertical types due to the weights of the structure itself, occupying personnel, equipment, rain, snow, and any others. Horizontal loads include the forces due to wind and earthquake. One typical code, the 2012 International Building Code, reference 1, is freely available from the website shown. Besides giving the individual load cases, this code also gives the load combinations to be resisted safely by the structure. Using load and resistance factor design :

(1) 1.4*(D+F)
(2) 1.2*(D+F) + 1.6*(L+H) + 0.5(Lr or S or R)
(3) 1.2*(D+F) +1.6*(Lr or S or R) + 1.6*H + (f1*L or 0.5*W)
(4) 1.2*(D+F) + 1.0*W + f1*L + 1.6*H + 0.5*(Lr or S or R)
(5) 1.2*(D+F) + 1.0*E + f1*L + 1.6*H + f2*S
(6) 0.9*D + 1.0*W + 1.6*H
(7) 0.9*(D+F) + 1.0*E + 1.6*H

where * denotes multiplication and :
D = dead load
E = combined effect of horizontal and vertical earthquake induced forces as defined in ASCE 7
H = load due to lateral earth pressure, ground water pressure, or pressure of bulk materials
L = roof live load > 20 psf and floor live load
Lr = roof live load <= 20 psf
R = rain load
S = snow load
W = load due to wind pressure
f1 = 1 for places of public assembly live loads > 100 psf and parking garages, and 0.5 for other live loads
f2 = 0.7 for roof configurations (such as sawtooth) that do not shed snow off the structure, and 0.2 for other roof structures

In each combination, the most critical effects must be used. For example, uniform, partial, or shadow snow load may govern.

The loads on the previous page must, in turn, be multiplied by a second factor in addition to the one shown. This factor is determined by the "Risk Category", which has four possible values, I, II, III, and IV, defined as follows :

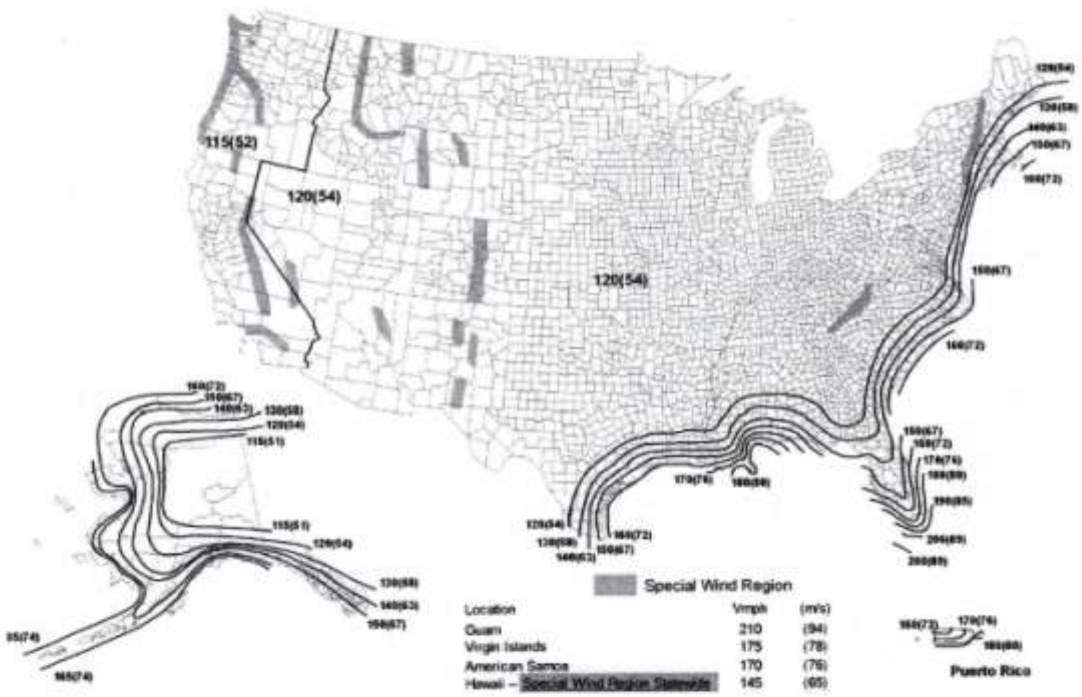
I : Structures which represent a low hazard in the event of failure, such as agricultural facilities and minor storage facilities.

II : Structures that don't fall into I, III, or IV.

III : Structures that represent a substantial hazard to human life, such as schools and places of public assembly.

IV : Structures designated as essential facilities. Examples are hospitals, emergency shelters, and power Generating stations.

The factor used increases with Risk Category number. Snow and earthquake factors are given in the respective chapters in ASCE 7 – "Minimum Design Loads for Buildings and Other Structures". Wind factors are, however, given by separate maps for each Risk Category, as shown below for Risk Categories III and IV from reference [1], in mph (meters/sec) for 3 second wind gusts.



3. LOAD AND RESISTANCE FACTORS DESIGN

The 2012 International Building Code specifies the design, fabrication, and erection of buildings and other structures shall be in accordance with AISC 360 [2]. AISC 360 requires that the design strength of each structural component equals or exceeds the required strength determined on the basis of the LRFD load combinations. In equation form,

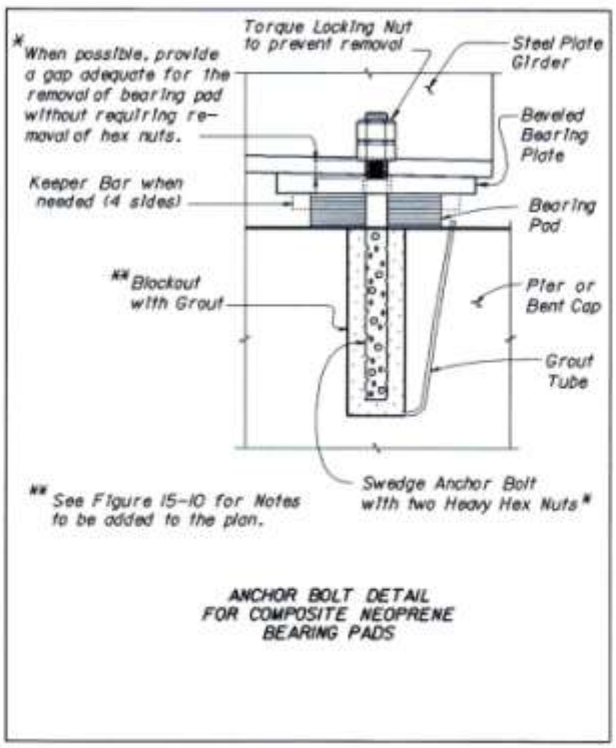
Ru <= Φ*Rn where

Ru	**=**	**required strength using the LFRD load combinations. These combinations are multiples of the maximum expected loads, as shown in section 2 above. The expected maximum loads multiplied by the load factors (which differ for different combinations) are called factored loads.**
Rn	**=**	**nominal strength = capacity of a structure or component to resist the effects of loads**
Φ	**=**	**resistance factor – accounts for unavoidable deviations of the actual strength from the nominal value and the manner and consequences of failure**
Φ*Rn	**=**	**design strength = maximum strength of member**

Note that load factors amplify the loads, while resistance factors reduce strength.

Item	Φ
Tensile yielding in gross section	0.90
Tensile rupture in net section	0.75
Compression	0.90
Flexure	0.90
Shear	0.90
Welded joints (Table J2.5)	0.75
Bolts, combined tension and shear	0.75
Bolts, slip-critical	
Standard and short slot perpendicular to load	1.00
Oversize and short-slot parallel to load	0.85
long-slotted holes	0.70
Bolts, bearing	0.75

The AISC specification also allows an "Allowable Strength Methods", which is not as detailed as LRFD since only one load factor (instead of a load factor and a resistance factor) is used. Some jurisdictions, however, not only mandate the use of LRFD only, but also provide detailed guidelines [3], and preferred details [4]. An example of the latter from the Florida Department of Transportation "Detailing Manual for Load and Resistance Factor Design" is shown below.



4. FIRST ORDER ELASTIC ANALYSIS

4.1 2D LINEAR STIFFNESS MATRIX

The basis of the finite element method is the analysis of a structure composed of finite (non-infinitesimal) elements connected at nodes. Forces and displacements at the nodes due to applied forces analyzed in a linear manner, i.e., the forces are always directly proportional to the displacements. The development here is similar to that given in reference 5.

The basic 2D beam element is a six degree-of-freedom beam segment, as shown on the page following. It is assumed that the segment has a constant cross-section (prismatic) and a constant modulus of elasticity. The degrees of freedom (d.o.f.) are the relative displacements (rotations) of the ends of the beam segment under load.

The beam forces are related to the beam displacements as:

$$\begin{bmatrix} p0 \\ f0 \\ m0 \\ p1 \\ f1 \\ m1 \end{bmatrix} = \begin{bmatrix} k00 & k01 & k02 & k03 & k04 & k05 \\ k10 & k11 & k12 & k13 & k14 & k15 \\ k20 & k21 & k22 & k23 & k24 & k25 \\ k30 & k31 & k32 & k33 & k34 & k35 \\ k40 & k41 & k42 & k43 & k44 & k45 \\ k50 & k51 & k52 & k53 & k54 & k55 \end{bmatrix} \begin{bmatrix} u0 \\ v0 \\ \Theta 0 \\ u1 \\ v1 \\ \Theta 1 \end{bmatrix}$$

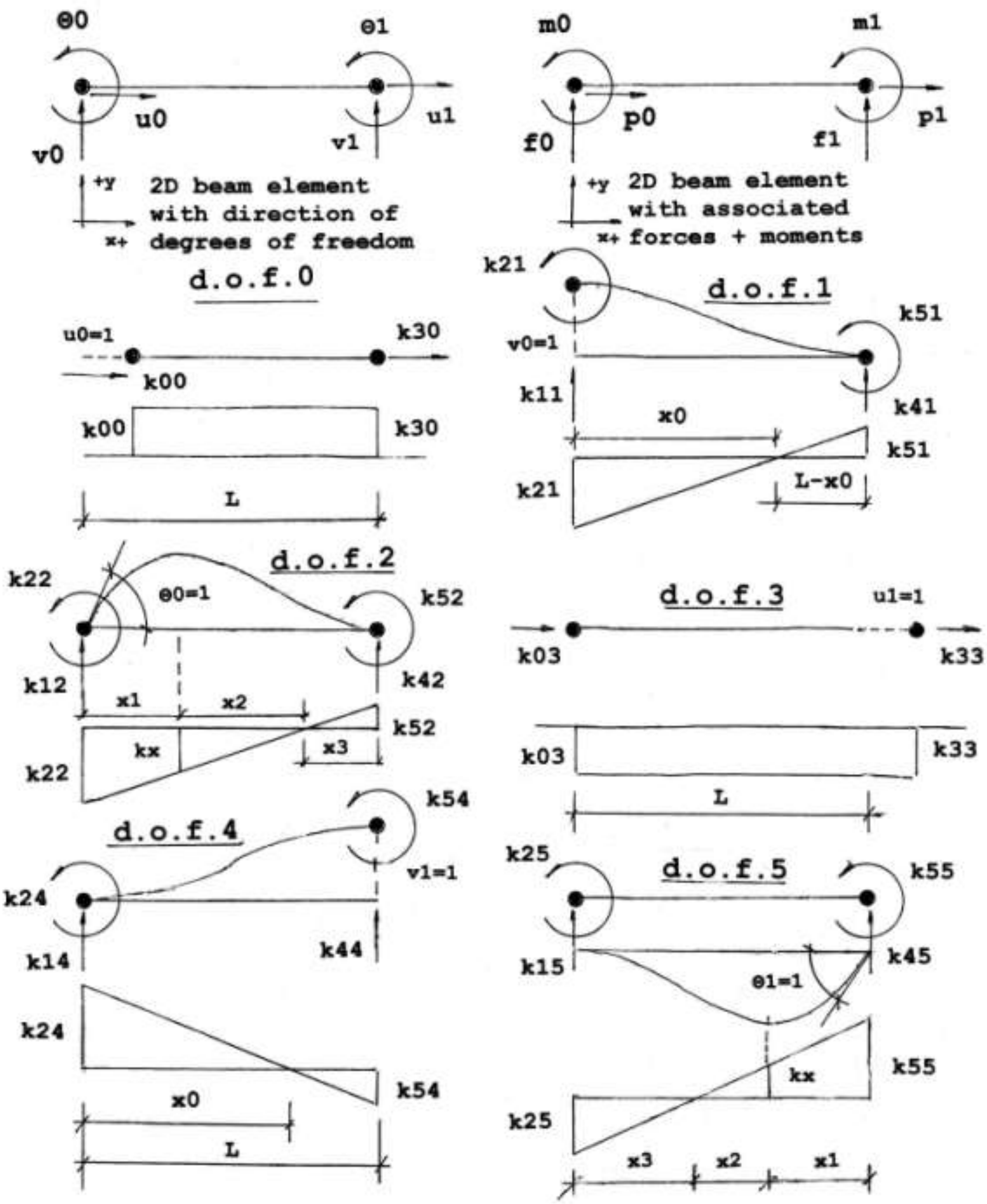
The coefficients of the stiffness matrix, kij, are found below, using the equations above. Note that the coefficient kij is defined as the resultant force (moment) at d.o.f. i due to a unit displacement (rotation) at d.o.f. j, all other displacements (rotations) set equal to zero. i.e. fixed. For example, to solve for k11, all except v0 = 0.

$$f0 = k10*u0+k11*v0+k12*\Theta 0+k13*u1+k14*v1+k15*\Theta 1$$

Thus all terms in the equation above are zero except

$$f0 = k11*v0$$

f0, and hence k11, is solved by analyzing the free body diagram of the line segment under a unit positive displacement of d.o.f. 1, namely v0. All the coefficients are solved in this manner.



θ0
θ1
u0
v0
v1
u1
+y
x+
2D beam element with direction of degrees of freedom
m0
m1
p0
f0
f1
p1
2D beam element with associated forces + moments
d.o.f.0
u0=1
k00
k30
L
d.o.f.1
k21
v0=1
k51
k11
k41
x0
L-x0
d.o.f.2
k22
θ0=1
k52
k12
k42
x1
x2
x3
kx
d.o.f.3
u1=1
k03
k33
L
d.o.f.4
k54
v1=1
k24
k14
k44
x0
L
d.o.f.5
k25
k55
k15
k45
θ1=1
kx
x3
x2
x1

d.o.f.0

Here the beam is in compression.For equilibrium,k30 = -k00. Since Δ = p*L/E*A and Δ = 1, p = k00 = L/E*A. Now the only resisting force is axial,so that k10 = k20 = k40 = k50 = 0. Also axial forces do not resist vertical forces or angular rotations, giving k01 = k02 = k04 = k05 = 0.

d.o.f.1

In this case, k21 and k51 must both be positive to maintain the beam configuration. Since the moment diagram shown is the internal beam moments, the left portion is negative and the right portion positive.
By the first moment-area theorem, the change in slope from point 0 to point 1 is equal to the area of the m/E*I diagram between 0 and 1.

Therefore, k21*x0 = k51*(L-x0),
and, since the diagram is a straight line,

$$\frac{k21}{x0} = \frac{k51}{L-x0} \rightarrow x0 = \frac{k21}{k21+k51}$$

Substituting the second equation into the first gives :

K21^2 + k21*k51 = k21*k51 + k51^2

The only way this can be true is if k21 = k51.
By the second moment-area theorem, the deflection at point 1 with respect to a tangent at point 0 is equal to the first moment of the m/E*I diagram between 0 and 1, taken about point 1.

Δ = ((-5L/6)*(1/2)*k21*(L/2)+(L/6)*(1/2)*k21*(L/2))/E*I
Δ = -1

$$\text{This solves for } k21 = k51 = \frac{6*E*I}{L^2}$$

Taking moments about point 1,

$$K11*L = k21 + k51 \rightarrow k11 = \frac{12*E*I}{L^3}$$

For vertical equilibrium, k41 = - k11

d.o.f.2

Let the point of maximum displacement and zero slope to be labeled 0.5, a distance x1 from node 0. Using the first moment-area theorem, taking the area from node 0 to 0.5,

$$\Theta 0.5/0 = -(kx*x1+(k22-kx)*x1/2)/E*I = -1$$

This solves to

$$x1 = \frac{2*E*I}{K22+kx} \qquad (1)$$

Using the first moment-area theorem now between point 0.5 and node 1, the formed positive and negative triangular areas must sum to zero.
The only way for these right triangles to be equal in area, since they are similar, is for x2 = x3, and kx = k52

Thus equation (1) becomes

$$x1 = \frac{2*E*I}{K22+k52} \qquad (2)$$

The equation of moment is

$$m = \frac{K22+k52}{L}*x - k22$$

At x = x1+x2, m=0, which gives

$$x1+x2 = \frac{K22}{K22+k52}*L \qquad (3)$$

By similar triangles,

$$\frac{K22}{x1+x2} = \frac{k52}{x2} \rightarrow k22*x2 = k52(x1+x2) \qquad (4)$$

Summing lengths,

$$x1+2*x2 = L \qquad (5)$$

Equations (2)-(5) are four equations, in four unknowns, namely x1, x2, k22, and k52.

Equations (2) and (5) can be used to eliminate x1 and x2,

$$x1 = \frac{2*E*I}{k22+k52} \quad \text{and} \quad x2 = \frac{L}{2} - \frac{E*I}{k22+k52} \qquad (6,7)$$

$$x1+x2 = \frac{L}{2} + \frac{E*I}{k22+k52}$$

Substituting this result into (3) reduces to : **(8)**

$$K22 - k52 = \frac{2*E*I}{L}$$

Finally, using the second moment-area theorem, the deflection of node 0 with respect to the tangent at node 1 is the first moment of the moment diagram about node 0, and is equal to zero in this case.

$$-(1/3)*(x1+x2)*(1/2)*(x1+x2)*k22+(L-x2/3)*(1/2)*x2*k52 = 0$$

This reduces to $-(x1+x2)\text{^}2*k22 + (3*L-x2)*x2*k52 = 0$

Substituting (6) and (7) into this equation and then using (8) to eliminate k52 gives:

$-(\zeta+1)*(1+1/\zeta)\text{^}2 + (5+1/\zeta)*(1-1/\zeta)*(\zeta-1) = 0$, **where**

$$\zeta = \frac{L*k22}{E*I} - 1 \rightarrow \zeta = 0 \text{ or } 3, \text{ where 0 implies k52 negative.}$$

Thus $\zeta = 3$, $k22 = \dfrac{4*E*I}{L}$, **and** $k52 = \dfrac{2*E*I}{L}$

For equilibrium of moments, $k12*L = k22+k52 \rightarrow k12 = \dfrac{6*E*I}{L\text{^}2}$,

and for vertical equilibrium, $k42 = -k12 = -\dfrac{6*E*I}{L\text{^}2}$

d.o.f.3

In this case, the element is in tension, leading to

$$K33 = +\frac{E*A}{L} \quad \text{and} \quad k03 = -\frac{E*A}{L}$$

d.o.f.4

The development here is similar to that of d.o.f.2, giving

$$K14 = -\frac{12*E*I}{L\text{^}3} \qquad k24 = -\frac{6*E*I}{L\text{^}2}$$

$$K44 = +\frac{12*E*I}{L\text{^}3} \qquad k54 = -\frac{6*E*I}{L\text{^}2}$$

d.o.f.5

The development here is similar to that of d.o.f.3, giving

$$K15 = +\frac{6 E I}{L^2} \qquad k25 = +\frac{2 E I}{L}$$

$$K45 = -\frac{6 E I}{L^2} \qquad k55 = +\frac{4 E I}{L}$$

Collecting results from above, we have the basic two dimensional stiffness matrix for a prismatic beam and first order analysis, symmetric about the top left to lower right diagonal.

$$\begin{bmatrix} +\frac{EA}{L} & 0 & 0 & -\frac{EA}{L} & 0 & 0 \\ 0 & +\frac{12EI}{L^3} & +\frac{6EI}{L^2} & 0 & -\frac{12EI}{L^3} & +\frac{6EI}{L^2} \\ 0 & +\frac{6EI}{L^2} & +\frac{4EI}{L} & 0 & -\frac{6EI}{L^2} & +\frac{2EI}{L} \\ -\frac{EA}{L} & 0 & 0 & +\frac{EA}{L} & 0 & 0 \\ 0 & -\frac{12EI}{L^3} & -\frac{6EI}{L^2} & 0 & +\frac{12EI}{L^3} & -\frac{6EI}{L^2} \\ 0 & +\frac{6EI}{L^2} & +\frac{2EI}{L} & 0 & -\frac{6EI}{L^2} & +\frac{4EI}{L} \end{bmatrix}$$

Example 1

Using the stiffness matrix above, find all forces and deflections of a cantilever beam, given the forces F1, V1, and M1 at the free end. Model the beam by a single prismatic element.

Solution

A cantilever beam has zero deflections at the fixed end, and the fixed end forces cannot be specified independently. The basic stiffness equation is reduced to:

$$\begin{bmatrix} F1 \\ V1 \\ M1 \end{bmatrix} = \begin{bmatrix} \frac{EA}{L} & 0 & 0 \\ 0 & \frac{12*E*I}{L^3} & -\frac{6*E*I}{L^2} \\ 0 & -\frac{6*E*I}{L^2} & \frac{4*E*I}{L} \end{bmatrix} * \begin{bmatrix} u1 \\ v1 \\ \Theta 0 \end{bmatrix}$$

where u1, v1, and ϴ0 are the axial, shear, and rotational displacements at the free end. The equation is shown in symbolic form as [f1] = [K1]*[Δ], [K] being the f0ull 6x6 stiffness matrix.
This equation may be solved for the displacements by multiplying each side by the inverse of [K1],say [K1]^(-1). Using the standard method for finding the inverse of a square, non-singular matrix, namely dividing the transpose of the signed minor matrix by the determinant of the original matrix, we have

$$[K1]^{(-1)} = \begin{vmatrix} L/EA & 0 & 0 \\ 0 & L^3/3*E*I & L^2/2*E*I \\ 0 & L^2/2*E*I & L/E*I \end{vmatrix}$$

Now [Δ] = [K1]^(-1)*[f1] is solved for [Δ].

$$[\Delta]=[F*L/E*A \;\; L^3*V/3*E*I+L^2*M/2*E*I \;\; L^2*V/2*E*I+L*M/E*I]^T$$

where the 'T' exponent denotes transpose, i.e., a row matrix is shown instead of the actual column matrix.

The remaining unknown forces at the fixed end may now be found as [K]*[total displacements].
This gives F0 = -F1, V0 = -V1, and M0 = -M, which may have obtained from force and moment balance, as this system is statically determinant. The procedure was used, however, to

illustrate the method of solving a system with some free d.o.f.'s and specified loads at these d.o.f.'s. The method is summarized as:

(1) Given the stiffness matrix [K}, reduce it to a smaller matrix [K1] by eliminating the terms with respect to the fixed d.o.f.'s. [K1} represents the relationship between the free displacements and their loads.

(2) Invert this matrix to [K1]^(-1).

(3) Solve for free displacements.

(4) Now solve for all loads using [total forces] = [K]*[all displacements]

It will be seen that this procedure is the core of both first and second order analysis.

4.2 SHEAR DISPLACEMENTS

Beams are structures whose length is much greater than its depth. If this is true, the displacements are functions only of axial and flexural stresses, as shown in section 4.1 above. However, when the length to depth ratio is on the order of ten (10) or less, deflections due to shearing stresses should be included, see reference 6.

In accounting for shearing stresses, the axial displacements are not affected, but the transverse and rotation displacements (v0, ϴ0, v1, ϴ1) are affected. Thus the kij multiplying these items in the basic stiffness matrix will be modified.

In the following, reference is made to the d.o.f. diagrams in section 4.1 above.

Let the general v = vf + vs where

v = total vertical displacement

vf = vertical displacement due to flexure

vs = vertical displacement due to shear

These displacements are independent of each other, with the governing equations being :

$$\frac{d^2 vf}{dx^2} = +\frac{M}{E*I}, \quad \Theta = \frac{dvf}{dx}, \quad \text{and} \quad \frac{dvs}{dx} = -\frac{\lambda*V}{G*A} \quad \text{where}$$

M = moment at some point x along the beam

ϴ = angular deflection at some point x along the beam

V = shear at some point x along the beam

Ashr = effective web area for wide flange beam

The procedure for finding k1j, k2j, k4j, and k5j, (corresponding to d.o.f.'s 1, 2, 4, and 5) with j = 1, 2, 4, and 5 is :

(1) Integrate the three equations above.
(2) Substitute the end conditions for the particular d.o.f. case into (1) above.
(3) Solve for k1j, k2j, k4j, k5j.

d.o.f.1

From the diagram for d.o.f.1,

d^2*vf/dx^2 = (-k21+(k21+k51)*x/L)/E*I , integrating

dvf/dx = Θ = (-k21*x + (k21+k51)*x^2/2)/E*I + c1 (1)

vf = (-k21*x^2/2 + (k21+k51)*x^3/6*L)/E*I+c1*x+ c2 (2)

dvs/dx = - λ*k11/G*A

vs = - λ*k11*x/G*A + c3 (3)

Combine (2) and (3) to find total displacement in y-direction,

v = (-k21*x^2/2 +(k21+k51)*x^3/6*L)/E*I+c1*x – λ*k11*x/G*A + c4 (c4 = c2 +c3) (4)

At x = 0, v = 1 and Θ = 0 so that c1 = 0 and c4 = 1

At x = L, v = 0 and Θ = 0 giving from (1)

0 = (-k21*L + (k21+k51)/2)/EI → k51 = k21 (5)

And from (4)

0 = (-k21*L^2/2 + (k21+k51)*L^2/6)/E*I - Λ*k11*L/G*A + 1 (6)
-

From equilibrium of moments, k11*L = k21 + k51 (7)

(5), (6), and (7) are three independent equations in three unknowns, which may be solved for k22, k21, and k51. From equilibrium of forces, k41 = -k11 and

$$k21 = \frac{6 E I}{L^2 (1+\Phi)} , \quad \Phi = \frac{12 \lambda E I}{G A L^2}$$

The remaining degrees of freedom (columns in the stiffness matrix) are solved with the same differential equations, the only changes being in the initial conditions. Collecting all terms, the reduced stiffness matrix (axial terms omitted for clarity) is shown as :

$$[\,kij\,] = \begin{bmatrix} k11 & k12 & k14 & k15 \\ k21 & k22 & k24 & k25 \\ k41 & k42 & k44 & k45 \\ k51 & k52 & k54 & k55 \end{bmatrix}$$

$$\begin{bmatrix} +\frac{12 E I}{L^3 (1+\Phi)} & +\frac{6 E I}{L^2 (1+\Phi)} & -\frac{12 E I}{L^3 (1+\Phi)} & +\frac{6 E I}{L^2 (1+\Phi)} \\ +\frac{6 E I}{L^2 (1+\Phi)} & +\frac{E I (4+\Phi)}{L (1+\Phi)} & -\frac{6 E I}{L^2 (1+\Phi)} & +\frac{E I (2-\Phi)}{L (1+\Phi)} \\ -\frac{12 E I}{L^3 (1+\Phi)} & -\frac{6 E I}{L^2 (1+\Phi)} & +\frac{12 E I}{L^3 (1+\Phi)} & -\frac{6 E I}{L^2 (1+\Phi)} \\ +\frac{6 E I}{L^2 (1+\Phi)} & +\frac{E I (2-\Phi)}{L (1+\Phi)} & -\frac{6 E I}{L^2 (1+\Phi)} & +\frac{E I (4+\Phi)}{L (1+\Phi)} \end{bmatrix}$$

Notice that if Φ =0, this matrix reduces the values of The stiffness matrix to the values of the basic stiffness matrix in 4.1 above.

A simple procedure to calculate the stiffness matrix whether or not shear displacements are considered is:

(1) Input data.
(2) If shear displacements are not considered, set Φ = 0, else calculate Φ.
(3) Find stiffness matrix and continue.

The impact of the Φ term is shown on the next page, using a three dimensional graph from 'maxima', reference 7.

The impact of the shear deformations can be seen by examining the 1+Φ term. If it is very close to 1, shear deformations need not be considered.

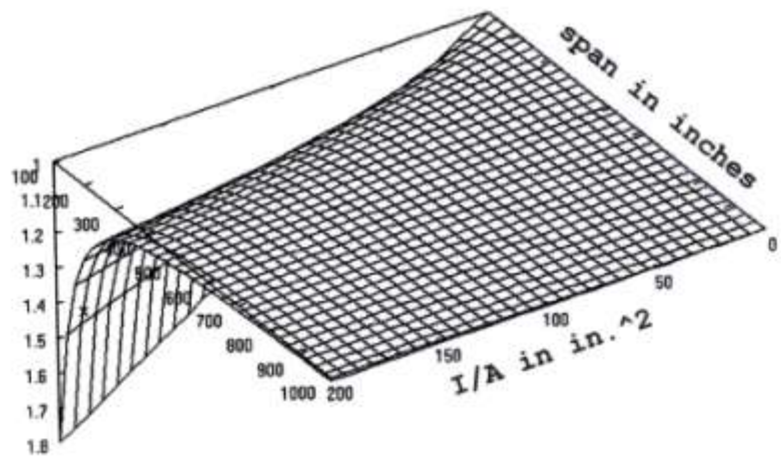
$$1+\Phi = 1 + \frac{12*\lambda*E*I}{G*A*L^2}$$

Let λ = 1

G = .3*E

Then :

$$1+\Phi = 1 + \frac{40*I}{A*L^2}$$, which is plotted below



4.3 SEMI-RIGID CONNECTIONS

The steel code recognizes three types of connections, simple, fully restrained (FR), and partially restrained (PR). Simple connections, also called pinned connections, transmit negligible moment between members. Typically, they have a rotational stiffness, M/Θ in kip-inches/radian, of less than or equal to 2*E*I/L. Fully restrained connections, also called rigid connections, transmit moment with negligible rotation between members, with a typical rotational stiffness greater than or equal to 20*E*I/L. Partially retrained connections, also called semi-rigid connections, transmit moment with non-negligible rotation between members, which must be included in the analysis,

and lie numerically between pinned and fixed connections.

Semi-rigid connections, represent the real behavior of a connection under load, as opposed to the ideal cases of perfectly pinned or perfectly rigid.
As an example, consider a gable frame. In this type of frame three major types of connection types are present, ridge, eave, and base. Semi-rigidity of the ridge connection increases the eave moment. Semi-rigidity of the eave connection increases the ridge moment. Semi-rigidity of the base connection couples moment into the foundation, for which it may not be designed.

In the following analysis, the force, stiffness, and displacement indices have been decreased by 1. This is because the program used assigns, to an array of n numbers, the labels 0 through n-1.

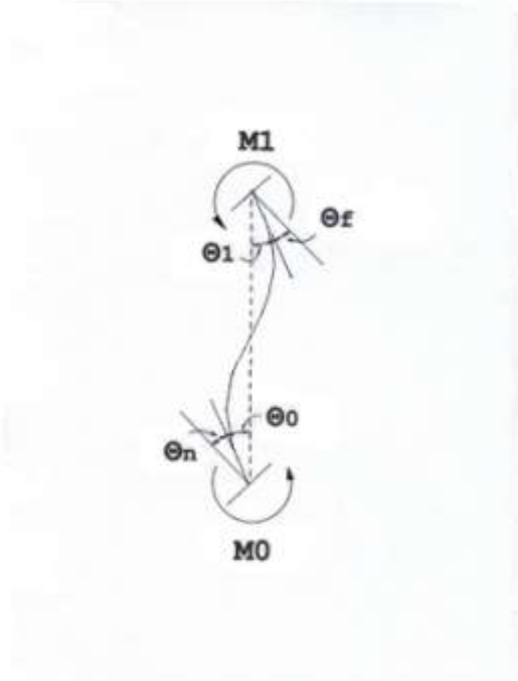
Consider the general two-dimensional stiffness matrix of a beam, be it prismatic or non-prismatic. The flexural terms are in red, and kij = kji by symmetry.

$$[K] = \begin{bmatrix} K00 & 0 & 0 & K03 & 0 & 0 \\ 0 & K11 & K12 & 0 & K14 & K15 \\ 0 & K12 & K22 & 0 & K24 & K25 \\ K03 & 0 & 0 & K33 & 0 & 0 \\ 0 & K14 & K24 & 0 & K44 & K45 \\ 0 & K15 & K25 & 0 & K45 & K55 \end{bmatrix}$$

The matrix equation expressing element forces in terms of element displacements is:

$$\begin{bmatrix} fx0 \\ fy0 \\ M0 \\ fx1 \\ fy1 \\ M1 \end{bmatrix} = [K]* \begin{bmatrix} x0 \\ y0 \\ \Theta 0 \\ x1 \\ y1 \\ \Theta 1 \end{bmatrix}$$

where the subscript 0 refers to the near end, and the subscript 1 to the far end.
The following diagram shows the relationships of the angles at the end of the beam.



$\Theta 0, \Theta 1$ = angular disp. due to beam stiffness
$\Theta n, \Theta f$ = angular disp. due to connection
$\Theta 0t$ = $\Theta 0 + \Theta n$ = total ang. disp., near end
$\Theta 1t$ = $\Theta 1 + \Theta f$ = total ang. disp., far end
M0 = $kn*\Theta n$ = near end moment, k″
M1 = $kf*\Theta f$ = far end moment, k″
kn = near end stiffness, k″/rad
kf = far end stiffness, k″/rad
cn = 1/kn = near end compliance, rad/k″
cf = 1/kf = far end compliance, rad/k″

In the equations above, the non-axial forces (fy0, M0, fy1, M1) are expressed in terms of y0, $\Theta 0$, y1, and $\Theta 1$. We wish, however, to have them expressed in terms of the total end displacements, y0, $\Theta 0t$, y1, and $\Theta 1t$. In other words, we need a new stiffness matrix which incorporates the semi-rigid end connections.

From the above, $\Theta 0 = \Theta 0t - cn*M0$
$\Theta 1 = \Theta 1t - cf*M1$

Substituting these values for $\Theta 0$ and $\Theta 1$ into the equations for the non-axial forces, we have :

$$fy0 + K12*cn*M0 + K15*cf*M1 = K11*y0 + K12*\Theta 0t + K14*y1 + K15*\Theta 1t$$

```
M0  + K22*cn*M0 + K25*cf*M1 =
      K12*y0 + K22*Θ0t + K24*y1 + K25*Θ1t
fy1 + K24*cn*M0 + K45*cf*M1 =
      K14*y0 + K24*Θ0t + K44*y1 + K45*Θ1t
M1  + K25*cn*M0 + K55*cf*M1 =
      K15*y0 + K25*Θ0t + K45*y1 + K55 *Θ1t
```

These kij terms may include only flexural terms, as in 4.1 above, or both flexural and shear terms, as in 4.2 above.

These equations are solved to give the new stiffness matrix as :

$$\begin{bmatrix} K00 & 0 & 0 & K03 & 0 & 0 \\ 0 & J11 & J12 & 0 & J14 & J15 \\ 0 & J12 & J22 & 0 & J24 & J25 \\ K03 & 0 & 0 & K33 & 0 & 0 \\ 0 & J14 & J24 & 0 & J44 & J45 \\ 0 & J15 & J25 & 0 & J45 & J55 \end{bmatrix}$$

See the function 'getksr' in Appendix 1 for a listing of these J-terms and the complete two-dimensional beam stiffness matrix, given the rigid stiffness matrix. Note that this is a linear matrix, that is, it does not change with load and, in general, is not symmetric. It also does not model a pinned connection directly, although this connection may be approached closely by increasing the compliance to a large value.

A typical nonlinear compliance for a general connection :

$$\text{compliance} = \frac{\Theta i}{Mi} = \frac{1}{(Rki*(1-(Mi/Mu)^n)^{(1/n)})} \quad \text{where}$$

i = 0,1
Rki = initial rotational stiffness
Mi = moment at point of interest
Mu = ultimate moment for connection
n = varies with connection type

Note the compliance gets large without limit as Mi → Mu. In the following work, compliances are taken as constant.

4.4 FIXED END FORCES – LOADS WITHIN NODES

It has been tacitly assumed up to this point that all loads are applied only at the element nodes. This approach can be used for loads internal to the element by subdivision into smaller elements. This process leads to an approximation for distributed loads, no matter how many subdivisions are used.

In a more general sense, the following equation relates external node forces, beam displacements, and internal forces, with superscript T indicating transpose :

$$[Fnodes]^T = [K]*[displacements]^T + [fef]^T \text{ , where}$$

$[Fnodes]^T$ **= forces on element nodes (external)**

$[K]$ **= member stiffness matrix**

$[disp.]^T$ **= member displacements at element ends (not at nodes for semi-rigid elements)**

$[fef]^T$ **= fixed end forces, i.e., forces at element ends required to equilibrate forces (and/or moments) internal to the beam, with all end displacements equal to zero. This requires solution of a structure indeterminant to the third degree.**

The case of semi-rigid end connections requires some more detail. In Section 4.3, the equations relating node lateral and end rotations may be written as :

$$\begin{bmatrix}1 & k12*c0 & 0 & k15*c1\\ 0 & 1+k22*c0 & 0 & k25*c1\\ 0 & k42*c0 & 1 & k45*c1\\ 0 & k52*c0 & 0 & 1+k55*c1\end{bmatrix} * \begin{bmatrix}V0\\ M0\\ V1\\ M1\end{bmatrix} = [Kbasic]*\begin{bmatrix}v0\\ \Theta 0t\\ v1\\ \Theta 1t\end{bmatrix} + \begin{bmatrix}Vf0\\ Mf0\\ Vf1\\ Mf1\end{bmatrix}$$

where [Kbasic] is the basic stiffness matrix in section 4.2 above, and [SR] is the matrix at left above. The node forces may be solved by multiplying each side of the equation above by the inverse of [SR], called [ISR].

$$[node\ forces]^T = [ISR]*[Kbasic]*[disp]^T + [ISR]*[fef]^T$$

Note that [ISR]*{Kbasic] is identically equal to the semi-rigid stiffness matrix found in section 4.3. This matrix [ISR], augmented for axial terms, is shown in Appendix 2.

This extra work is required so that when we combine elements to form a structure, all forces and displacements are compatible node values, whether or not each element has a semi-rigid connection.

Finding the fixed end forces requires solving a loaded beam fixed at both ends, indeterminant to the third degree. This process may be done by the force method, illustrated here in four steps.

(1) Establish the primary structure, which is the original structure with releases at the far end. In this case, axial displacements are not considered, so that the primary structure has vertical displacement and rotational displacement at the far end.

(2) Calculate displacements of the primary structure due to unit values of the redundant forces. This results in a 2x2 matrix

$$\begin{bmatrix} f11 & f12 \\ f21 & f22 \end{bmatrix} = [fij] \text{ where}$$

i = direction of displacement
j = direction of cause
Note that the fij are flexibility coefficients, with units displacement/force. They do not vary with internal loads and thus may be used for different internal loads, such as concentrated forces and linearly varying distributed loads.

(3) Find the displacements at the releases of the primary structure due to the loads to be figured for fixed-end forces, call this 2x1 matrix [Dp].

(4) Let [R] = 2x1 matrix of fixed end forces at the far end. Solve for [R] from [Dp] + {fij]*[R] = 0.
[R] = - [Ifij]*[Dp], where [Ifij] = inverse of [fij].

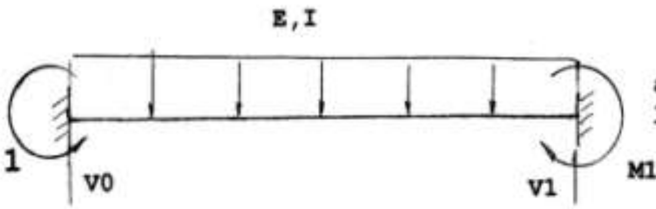
$$[Ifij] = \frac{1}{den}\begin{bmatrix} +f22 & -f12 \\ -f21 & +f11 \end{bmatrix}$$

$$den = f11*f22 - f12*f21$$

and [Ifij] is a stiffness matrix.

This process is illustrated below by a uniform distribution load, commonly used both in the literature and in practice.

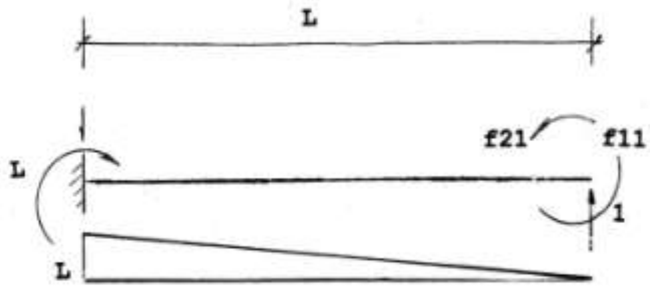
step 1



actual structure and loads with fixed ends

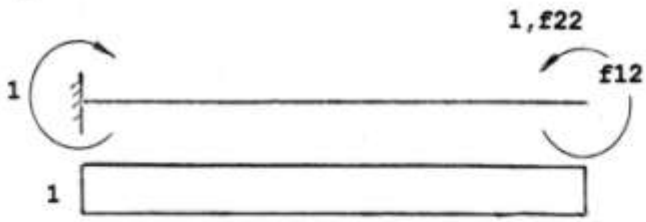
primary structure with 3 releases at far end

step 2



primary structure with unit force in +y direction at far end

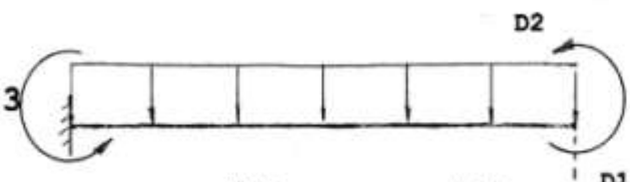
f21 = (1/2*E*I)*L^2
f11 = (1/3*E*I)*L^3



primary structure with unit moment at far end

f12 = (1/2*E*I)*L^2
f22 = (1/E*I)*L

step 3



primary structure with actual load

D1 = -w*L^2/(8*E*I)
D2 = -w*L^3/(6*E*I)

$$m = -\frac{w*L^2}{2} + w*L*x - \frac{w*x^2}{2}$$

step 4

$$[Ifij] = \begin{vmatrix} +12*E*I/L^3 & -6*E*I/L^2 \\ -6*E*I/L^2 & +4*E*I/L \end{vmatrix}$$

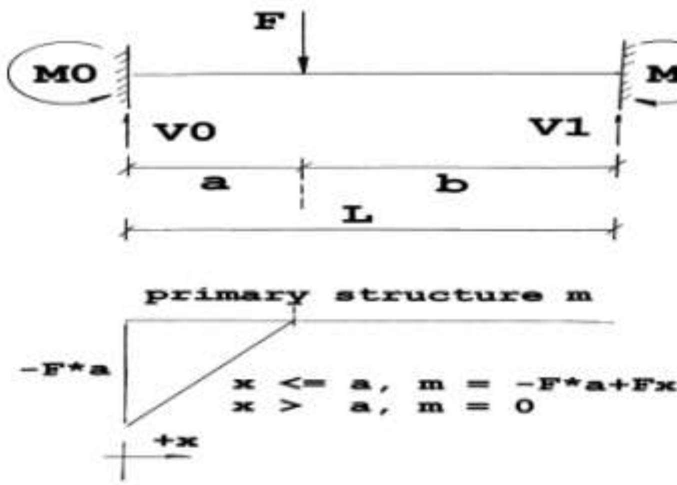
$$[\,V1\ M1\,]^T = -[Ifij]*[D1\ D2]^T$$

For equilibrium at left hand side,
Vertical force = *w*L/2
Moment = +w*L^2/12

fixed end force = [0 w*L/2 w*L^2/12 0 w*L/2 -w*L^2/12]

Example 2

Find the fixed end forces for a beam with a single concentrated load at and arbitrary position within the beam nodes.



Solution

The approach here is the same as that for the uniform distributed load above. The [fij] and [Ifij] are the same as above.

$$[fij] = \begin{bmatrix} L^3/3*E*I & L^2/2*E*I \\ L^2/2*E*I & L/E*I \end{bmatrix}$$

and

$$[Ifij] = \begin{bmatrix} 12*E*I/L^3 & -6*E*I/L^2 \\ -6*E*I/L^2 & 4*E*I/L \end{bmatrix}$$

Solving the primary structure for displacement at end 1 with real load,

$$[\ D1\ D2\ ]^T = [-F*a^2*(b+2*a/3) \quad -F*a^2/2*E*I]^T$$

As above, $[V1\ M1]^T = -[Ifij]*[D1\ D2]^T$

which yields

V1 = F*a^2*(a+3*b)/L^3 and M1 = -F*a^2*b/L^2

Using the equations of equilibrium,

V0 = F*b^2*(3*a+b)/L^3 and M0 = +F*a*b^2/L^2

Fixed end force vector = [0 V0 M0 0 V1 M1]

4.5 FIXED END FORCES - TEMPERATURE CHANGE

For a beam with end unrestrained against temperature changes, $\Delta L = \alpha*\Delta T*L$ where

ΔL = change in length, in.
α = temperature coefficient of resistance, in./in./°F
ΔT = change in temperature, °F
L = original length, in.

The force necessary to negate the change in length is :

F = $E*A*\Delta L/L$ where
F = force, kips
E = modulus of elasticity, ksi
A = cross-sectional area, in.^2

Combining these two equations, we have :

F = $E*A*\alpha*\Delta T$ and the fixed end force is:

$$[\ +E*A*\alpha*\Delta T \quad 0 \quad 0 \quad -E*A*\alpha*\Delta T \quad 0 \quad 0\ ]^T$$

Example 3

Consider a track rail system, where maximum displacement between rails cannot exceed one (1) inch. What is the maximum rail length which will not exceed this specification for a temperature range of -20°F to +100°F. Let α = .0000065 in./in/°F

Solution

The maximum ΔT = 100 -(-20) = 120 °F.
Solve $\Delta L = \alpha*\Delta T*L$ for L,

L = $\Delta L/\alpha*\Delta T$
L = 1/(.0000065*120)
L = 1282 in. = 106.8 ft

Note that standard rail length is 39 feet, which has a maximum ΔL = .0000065*120*39*12 = .365 in.

4.6 FRAME ANALYSIS

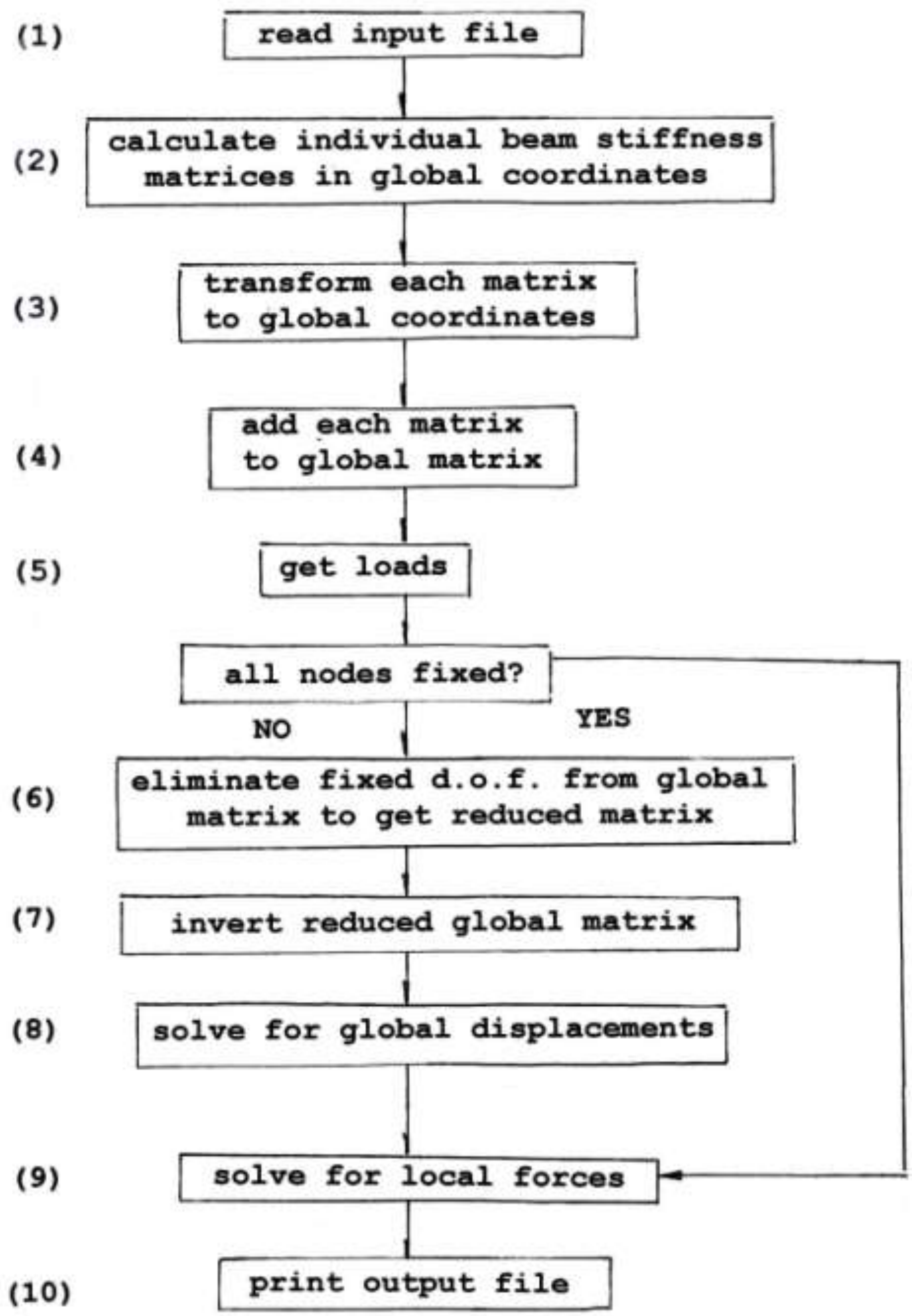
Section 4 up to this point has covered single beam elements. Most applications, however, consist of several members connected as a frame. The development of frame analysis in this section is done by description of a program 'frstx.c', a block diagram of which is given on the following page. The ten (10)parts of 'frstx.c' are :

Part (1) →
The input file lists:
Node coordinates, i.e. x and y global coordinates (2 x no.nodes)
Beam connection matrix – each entry has 1st node no., 2nd node no. and property no. (3 x no.beams)
Beam properties – A,Ashr,I,E,G,shear?,1st node compliance, and 2nd node compliance (8 x no beam properties). Note that a rigid connection has zero compliance.
Fixed degrees of freedom (no. fixed d.o.f.)
Node load degrees of freedom (no. node loads)
Node load values (no. node loads)
Total number of loads (no. loads)
Fixed end forces beam numbers (no. fef beams)
Fixed end forces matrix (6 x no. fef beams). This matrix does not include modifications due to semi-rigid connections as described in section 4.4 above. This correction is applied in part 5.

Part (2) →
For each beam element, find the stiffness matrix entries,
which may include shear effects and/or semi-rigid ends. These entries are in local coordinates.

Part (3) →
Transform each individual beam element to global coordinates, using
ϴ = tan^(-1)((y1-y0)/(x1-x0)),cϴ = cosϴ, sϴ = sinϴ

$$[TB] = \begin{bmatrix} c\Theta & s\Theta & 0 & 0 & 0 & 0 \\ -s\Theta & c\Theta & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & c\Theta & s\Theta & 0 \\ 0 & 0 & 0 & -s\Theta & c\Theta & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$


(1)
read input file
(2)
calculate individual beam stiffness matrices in global coordinates
(3)
transform each matrix to global coordinates
(4)
add each matrix to global matrix
(5)
get loads
all nodes fixed?
NO
YES
(6)
eliminate fixed d.o.f. from global matrix to get reduced matrix
(7)
invert reduced global matrix
(8)
solve for global displacements
(9)
solve for local forces
(10)
print output file

$$[ITB] = \begin{bmatrix} c\Theta & -s\Theta & 0 & 0 & 0 & 0 \\ s\Theta & c\Theta & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & c\Theta & -s\Theta & 0 \\ 0 & 0 & 0 & s\Theta & c\Theta & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

[Kglobal] = {ITB]*[Klocall}*[TB]

Part (4) →
Add each individual matrix to the global matrix, which has n x n entries, where n = 3 * no. nodes .

Part (5) →
The fixed end forces are first adjusted for semi-rigid terms as shown in section 4.4. The end result of this part is the column matrix with 'no. loads' entries.
If all the degrees of freedom are fixed, go to part 9.

Part (6) →
Eliminate fixed degrees of freedom from the complete global matrix. This results in a reduced m x m global matrix, where m = 3 X no. nodes – no. fixed.

Part (7) →
Here we invert the reduced m x m matrix by a series of row operations. The original matrix is augmented by an m x m identity matrix placed to the right of the matrix to be inverted. The three row operations are:
1. **Interchange of rows to obtain a nonzero pivot element.**
2. **Multiplication of all elements in a row by a constant to set pivot = 1.**
3. **Addition of elements of one row, each multiplied by the same constant, to the elements of a second row.**

The object is to transform the original matrix on left to the unit matrix by the row operations.

sample input:	2	4	3	2 |	1	0	0	0
	3	6	5	2 |	0	1	0	0
	2	5	2	-3 |	0	0	1	0
	4	5	14	14 |	0	0	0	1
with output :	1	0	0	0 |	-23	29	-12.8	-3.6
	0	1	0	0 |	10	-12	5.2	1.4
	0	0	1	0 |	1	-2	1.2	0.4
	0	0	0	1 |	2	-2	0.6	0.2
					<-inverted matrix->			

Part (8) →
Solve for global displacements of the free degrees of freedom.

$$[\text{global displacements}]^T = [\text{inverted global matrix}] * [\text{loads}]^T$$

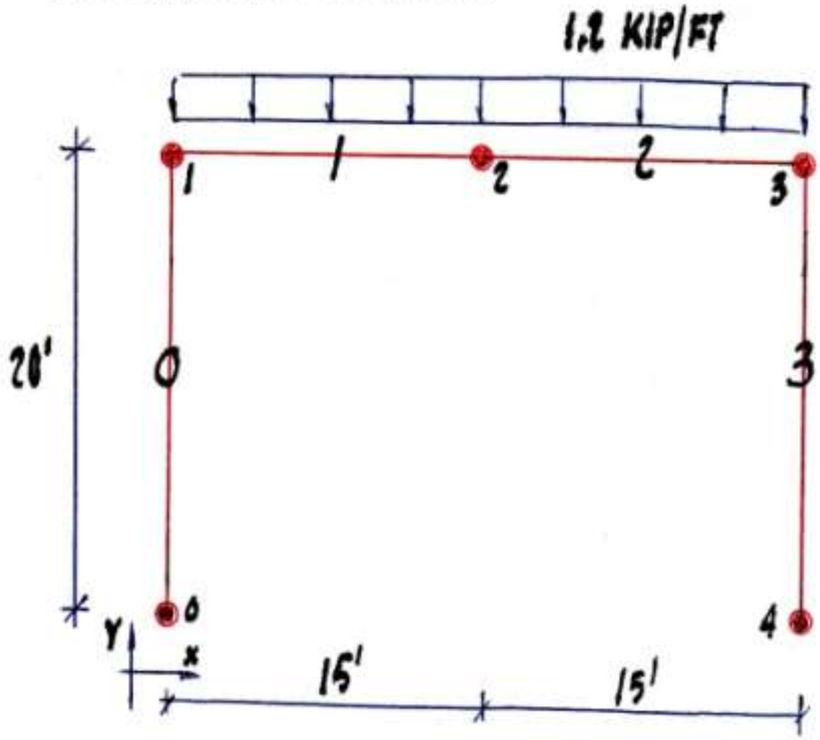
Part (9) →
Solve for local forces by first converting, for each individual beam, global displacements to local displacements. Then use :

$$[\text{Flocal}]^T = [\text{Klocal}] * [\text{local displacements}]^T$$

Part (10) →
Print global displacements and local forces into output file.

Example 4

The portal frame shown in outline below has W12x30 columns and rafter, with dimensions shown. Using 'frstx' program, find eave and center of rafter moments. Shear deflections need not be considered and both eave connections are rigid. Base connections are pinned.



Solution

The input file is shown,with explanation of entries in red. Consistent units are inches and kips, to get beam properties in familiar units. The fixed end forces are derived as shown in section 4.4.

Input file

----

5 4 1 4 0 9 2
Numbers of nodes, beams, beam properties, fixed d.o.f.'s, concentrated loads, total node loads, and beams with fixed end forces, respectively.

0 0 0 240 180 240 360 240 360
Global coordinates.

0 1 0 1 2 0 2 3 0 3 4 0
Beam connection matrix with material type number.

8.79 3.2084 238 29000 8700 0 0 0
Beam properties A, Aweb, I, E, G, consider shear?, compliance at first node, compliance at second node

0 1 12 13
Fixed degrees of freedom – this has the effect of making nodes 0 and 4 pinned.

1 2
Numbers of beams with fixed end forces.

0 9 270 0 9 -270
0 9 270 0 9 -270
Beams 1 and 2 fixed end forces

The output file gives eave moment as 747 kip-inches and moment at rafter center as 876 kip-inches.

Example 5

Redo example 4 with semi-rigid eave connections of stiffness 100 kip-inches/milliradian. Do this by changing beams 1 and 2 to have end compliances of .00001 rad/k"

Result

Eave moment = 591 k" and center of rafter moment 1029 k".

Example 6

Consider the frame of example 4 to be non-heated and erected at a temperature of 80 degrees Fahrenheit. During the winter the temperature drops to -20 degrees. Find the eave moments generated by this temperature change.

Solution

Using the procedure of section 4.5, each beam has a fixed end force vector of :

$$[\ -1656.915\ \ 0.0\ \ 0.0\ \ +1656.915\ \ 0.0\ \ 0.0\ ]^T$$

The result of the computer run is 129.4 kip-in. eave moment. The angular direction of this load on the connections is the same as that of the vertical loads, i.e., they are additive, which is a factor to consider as the vertical load may be maximum at the same time.

4.7 SUPPORT DISPLACEMENTS

Support displacements may also be handled using fixed end forces. One method of doing this is given without proof here:

(1) Using frstx.c or some other general purpose beam finite element program, solve for forces with all degrees of freedom fixed, except for the one where the displacement is desired. At this d.o.f., program a unit force.

(2) Calculate the quantity displacement desired divided by the displacement calculated in (1). Call the c1 = constant. Now find the quantity 1/c1, and use this factor for the force in the same program as in (1). The results of this run are the fixed end forces on the member connected to the support displacement.

(3) Adjust the program input file to the actual fixed degrees of freedom, and initial coordinates, using the fixed end forces from (2) to any other fixed end forces in the beam attached to the displacement.

Example 7

Consider the frame of example 4 to have a one inch displacement of node 0 in the x-direction. Solve for forces caused by this deflection.

Solution

----

(1) The input file should contain fourteen fixed degrees of freedom with with a unit load of 1 in the +x direction at node 0. The result of this computer run is that the displacement at d.o.f. 0 is +0.1669081 in. Therefore the force to be used in step (2) = 1/0.1669081 = 5.991321 kip.

(2) The only change from step (1) to step (2) is to change the load at d.o.f. 0 from 1 to 1/c1. The result of this run gives the fixed end forces of :
0.0 -5.991321 -718.9585 0.0 5.991321 -7.718.9585

(3) Using the fixed end forces from (2) into the actual structure obtains the output file giving base forces of 0.2303605 each side and eave moments each side of 55.28649 each side.

Note : Examples 6 and 7 are checked against reference 8.

Example 8

Consider the same frame as above, except that the bases are fixed instead of pinned.

Solution

Steps (1) and (2) are the same as in example 7. The only change in (3) is to change the bases from pinned to fixed, i.e., from fixed d.o.f.'s 0 1 12 13 to 0 1 2 12 13 14.

This results in:
Base force increases to 1.308180 kip per side
Eave moments increase to 89.70377 kip-in./side
Base moments are 224.2594 kip-in./side

4.8 INTERNAL FORCES

The outputs of finite element programs obtain the forces at the nodes only. Internal forces may cause higher moments within the beam element, which the node forces alone do not show. If the internal loads are concentrated types, this can most easily be handled by placing additional nodes at the points of the concentrated loads. If the load is uniformly distributed, however, another approach is needed. In Example 9 following, the rafter connections have a compliance of 0.00001 radians/kip.

Let the forces at the first node be P0, V0, and M0, which are positive in the x-direction, positive in the y-direction, and positive counterclockwise, respectively. The moment m(x) along the beam is:

m(x) = -M0 +V0*x -w*x^2/2, w = distributed load
The moment will be a maximum (or minimum) where dm/dx = 0.
dm/dx = V0 - w*x = 0 → x = V0/w for max(or min)moment.
Substituting this value for x into the equation for m(x),
m max = -M0 +V0^2/w - w*V0^2/w^2 = -M0 + V0^2/2*w

Example 9

Consider a portal pinned frame, W14x79 each member, oriented in the north-south plane, with a uniform load of 2.4 kip/ft, and a lateral load of +10 kip in the +x-direction at the north eave. Let span = 30 ft and height = 20 feet.

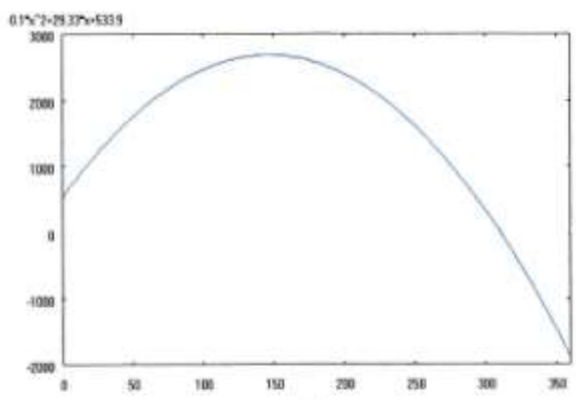
Solution

```
frstx.c gives :     north eave shear  = +29.33  kip
                    north eave moment = -533.9  kip-in.
                    south eave shear  = +42.67  kip
                    south eave moment = -1866.1 kip-in.
```

x at xmax = 29.33/(2.4/12) = 146.7 inches
Substituting this value into the equation for m max gives:
m max = +2685.0 kip-in., considerably greater than either eave moment!

The equation for rafter moment is:

M = +533.9 + 29.33*x - 0.1*x^2, x in inches

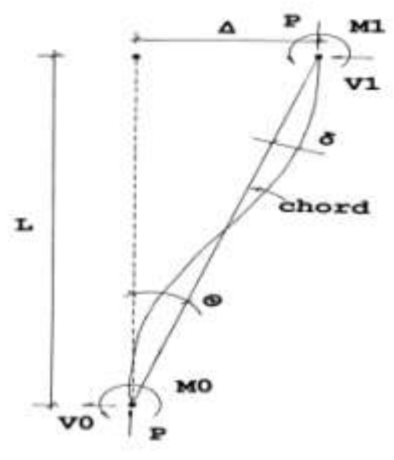


(

5. SECOND ORDER ELASTIC ANALYSIS

5.1 INTRODUCTION

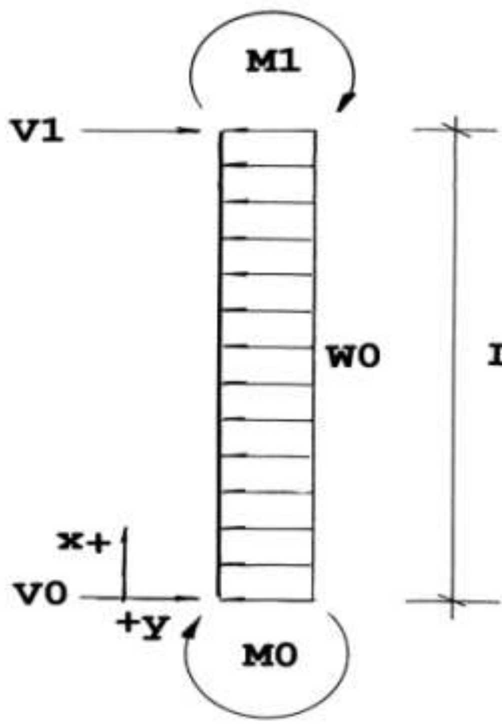
The first order elastic analysis covered above does not account for the two types of structural nonlinearities, namely material and geometric.
Material nonlinearities include changes in elastic modulus with load, member yielding with load, residual stresses, and decrease of semi-rigid connection stiffness with moment. Second order elastic analysis, as defined by the AISC, does not model these directly but provides approximations to these nonlinearities.
Geometric nonlinearities include connection mis-alignment, P-Δ, effects and P-δ effects. Connection mis-alignment may be treated either directly or indirectly by 'notional loads'. Members exhibiting P-Δ and P-δ effects are called beam-columns.
First order elastic analysis is based on the undeflected (at rest) geometry, rather than the actual geometry after deflections.This gives rise to the P-Δ effect, modification of the sidesway stiffness caused by moments induced by difference of alignment of axial forces. This is also called the chord rotation effect.
The P-δ effect is caused by the axial force changing the flexural stiffness of an individual member. The member curvature effect is another name for the P-δ effect. In a beam pinned at one end, vertically supported at the other, with an internal distributed or concentrated load and an axial load, the P-δ effect obtains, but not the P-Δ effect, because there is no change in node location from rest.



5.2 GENERAL BEAM-COLUMN

5.2.1 NO AXIAL LOAD

The developments here and in the following section are similar to those of reference 9, except for the definition of P0, change in algebraic signs, and inclusion of moments and deflections when shear displacements are considered.
It is useful to determine the forces and displacements of a beam without and with shear effects, a sketch of which is shown as :



Note that the forces are not independent as
V1 = w0*L - V0 and
M1 = -M0 -w0*L^2/2 +V0*L
This prismatic beam is then characterized by values of M0, V0, and w0.

By statics, and the fact that
E*I*y’’ = m = -M0 +V0*x - w0*x^2/2, then
y’’ = +M0/E*I - V0*x/E*I +w0*x^2/2*E*I
By integrating twice,
y = +M0*x^2/2*E*I - V0*x^3/6*E*I +w0*x^4/24*E*I +C0*x + C1

Boundary conditions are y = 0 at x = 0 and x = L, resulting C1 = 0 and C0 = M0*L/2*E*I – V0*L^2/6*E*I + w0*L^3/24*E*I.

These equations may be used to solve for displacements and moments at any point x.

To find maximum moment, set dm/dx = 0, giving xmax = V0/w0.

Example 10

Consider a steel beam with
E = 29000 ksi
I = 485 in.^4
w0 = 0.2 kip/ft
L = 28 ft
M0 = M1 = 0
Find maximum moment and displacement.

Solution

Here V0 = w0*L/2 and thus xmax = L/2
C0 is found to be –w0*L^3/24*E*I and maximum moment of +w0*L^@/8 and maximum displacement of -5*w0*w0*L^4/384*E*I.
Numerically, mmax = +235.2 kip-in.
ymax = -0.1967 in.

Next we consider the same problem, but with the addition of consideration of shear displacements.
From above, v = V0 – w0*x
For shear displacements, we have
y'' = -(1/G*As)*dv/dx = -(-w0/G*As) = w0/G*As
where G = shear modulus and As effective shear area.

Integrating twice, y = (w0/G*As)*x^2/2 + C2*x + C3
Using the boundary conditions of y = 0 at x = 0 and x = L,
y = -(w0*x/2*G*As)*(L-x)
As above, maximum deflections occurs at x = L/2.

Example 11

For the beam in Example 11, find the total deflection with G = 11000 ksi and As = 4.6886 in.^2 (d*tw for W14x48)
Solution

Shear deflection = (.2/12)*168^2/2*(11000)*4.6886= -0.0046"
Total deflection = -.1967 - .0046 = - 0.2013"

5.2.2 AXIAL COMPRESSIVE LOAD INCLUDED

The three diagrams on the following page illustrate the geometry and forces of the single beam column. It is used to find deflections and moments between the nodes of a second order finite element analysis. These forces and/or displacements may exceed those of the node values and thus should be checked. The development in this section follows the work of Dr. Munoz in reference 9.

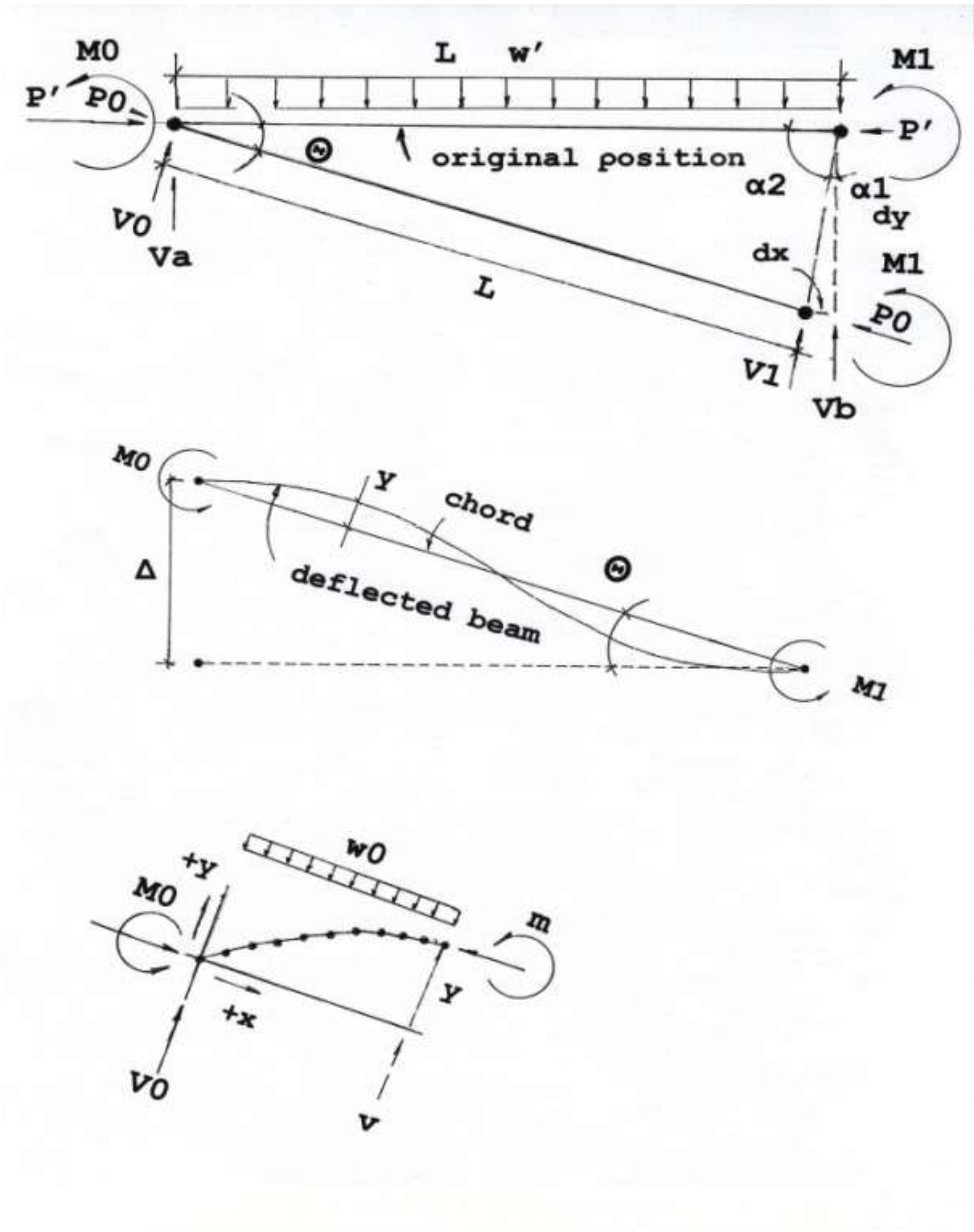
We assume that L is constant, i.e., the bowing effect, change in axial length with axial load, is neglected.

In the first diagram, dx and dy represent the net displacements of the beam after the analysis. Rigid body motion is eliminated by letting dx=dx1-dx0 and dy=dy1-dy0.

An isosceles triangle is formed with sides L, L, and (dx^2+dy^2)^(1/2), as shown. The angle Θ is formed by the trigonometry formula :
sin(A) = (2/(b*c))*(s*(s-a)*(s-b)*(s-c))^(1/2) where b and c are adjacent t0 angle A.
s = perimeter/2 = (a+b+c)/2
Substituting the values for a, b, c, and letting
sqrt(dx^2+dy^2) = dz,
s = L + dz/2
s-a = L-dz/2
s-b = s-c = z/2
Then sin(A) = dz*sqrt(L^2-dz^2)/L^2

Conversion of the original geometry forces and moments to the displaced chord is :
w0 = w'*cosΘ
P0 = P'*cosΘ –V'*sinΘ + (½)*w'*L*sinΘ
V0 = P'*sinΘ +V'*cosΘ

The second diagram shows the displacement, in an assumed configuration, of the beam with respect to the chord (the straight line from point 0 to point 1). The moment accompanying this displacement is called the P-δ effect, and is in addition to the P-Δ effect caused by frame movement under load.



The third diagram shows a free body diagram of an initial portion of the beam.This results in the following equation,

m= -M0+V0*x+w0*x^2/2-P0*y, where a CCW moment providing equilibrium for the beam segment is positive (w0 is neg.)

By the governing equation for moment, m = E*I*y'', we have

```
                                -M0 + V0*x + w0*x^2/2
y''+(P0/E*I)*y = f(x), f(x) = ---------------------
                                           EI
```

This equation is solved by summing the complementary solution (f(x) = 0) with a particular solution. Let the complementary solution be yc, the particular solution yp.

yc = A*sin(k*x) + B*cos(k*x), where k = (P/E*I)^(1/2)

Now let d( )/dx = D, so that the particular equation is :
(D^2 + K^2)*yp = f(x) → yp = (1/k^2 – D^2/k^4), when the fraction multiplying f(x) is divided out, and all forms above D^2 neglected. This results in:

yp = -M0/P0 + V0*x/P0 + w0*x^2/2*P0 - w0/k^2*P0

y = yc + yp

The values for A and B are solved by using the boundary conditions that m = M0 at x = 0, and m = M1 at x = L. Find m as the second derivative of y'' times +EI.

m = -k^2*E*I*A*sin(kx) - k^2*E*I*cos(kx) +w0/k^2

Making the substitution P0 = k^2*E*I,
m = -P0*A*sin(kx) - P0*B*cos(kx) + w0/k^2

Use the boundary conditions of m = M0 at x=0 and m = M1 at x = L,

B = -M0/P0 + w0/(k^2*P0)

```
    -M1+M0*cos(kL)    w0*(1-cos(kL))
A = ------------- +  ---------------
     P0*sin(k*L)     k^2*p0*sin(k*L)
```

```
                             1-cosα
Using the trig identity     ------ = tan(α/2)
                              sinα
```

```
     -M1+M0*cos(k*L)     w0*tan(k*L/2)
A =  --------------  +  -------------
       P0*sin(k*L)          k^2*P0
```

To find y, integrate y'' = +m/E*I twice.

```
y = A*sin(k*x) + B*cos(k*x) + w0*x^2/2*P0 + C0*x + C1
```

The two boundary conditions are that y = 0 at both x = 0 and x = L. This gives :

```
C1 = -B and
C0 = -A*sin(k*L)/L + B*(1-cos(k*L)/L – w0*L/2*P0
```

These equations for k,A,B,C0,C1,m,and y comprise the first part of program mark1.c, attached as Appendix 1.

The solutions to this point are based on y''(total) = y''(flexure), only. To consider shear, use the approach by Drs. Timoshenko and Gere found in reference 10.

The curvature due to flexure alone is derived above as

```
yf'' = -M0/E*I + w0*x^2/2*E*I + V0*x/E*I – P0*y/E*I
```

The additional curvature due to shear is P0*yt''/G*Ashr where yt'' = y'' = total curvature. Thus,

```
y''*(1-P0/G*Ashr) + P0*y = -M0/E*I +W0*x^2/2*E*I + V0*x/E*I
```

This differential equation is solved as above, combining complementary and particular solutions,

```
y=A*sin(k*x)+B*cos(k*x)–M0/P0+w0*x^2/2*P0+V0*x/P0-w0/k^2*P0
```

except that now k = sqrt(P0/ζ*E*I) and ζ = 1 – P0/G*Ashr, where G = the shear modulus.

Similarly, we solve B = (-M0+w0/k^2)/ζ*P0

```
       ζ*(M1-M0*cos(k*L))      ζ*E*I*w0*tan(k*L/2)
A = - ------------------  +  -------------------
          P0*sin(k*L)               P0*k^32
```

We find m by twice differentiating y,and using m = -E*I*y''

```
m = -(P0*A*sin(k*x) - P0*B*cos(k*x))/ζ + E*I*w0/P0
```

Now again use double integration to find y, obtaining

y = A*sin(k*x) + B*cos(k*x) +w0*x^2/2 + C0*x + C1

Using the initial conditions that y = 0 and x = 0 and at x = L, C1 = -B and
C0 = -A*sin(k*L)/L + B*(1-cos(k*L))/L – w0*L/2*P0

Now consider the additional displacements caused by shear alone.
y′′s = -(dV/dx)/g*Ashr where dV/dx = η = slope of shear equation. In general, for M0 and V0 at node 0, M1 and V1 at node 1, and with uniform distributed load of w0,

Of the five variables, V0,M0,V1,M1, and w0

The shear diagram is a straight line with coordinates (0,V0) and(L,-V1). This may be solved to yield
V(x) = η*x + V0 where

η = (M0+M1)/L^2+w0/2-V0/L (w0 is negative)

Now ys′′= - (1/G*Ashr)*dV/dx = -η/G*Ashr
Integrating twice, and using the boundary conditions of y = 0 at both x = 0 and x = L,
ys = -η*x*(x-L)/2*G*Ashr
which is the additional shear deflection.
These shear equations comprise the second part of mark1.c

Notes on equations for y and m above :

(1) Neglecting shear displacement is equivalent to letting G*As approach infinity (same as 1/G*As → 0).
(2) The equations are indeterminant if P = 0. In this case use the methods of section 5.2.1. If P is negative, indicating tension, the equations above are not valid, instead solve the equation y′′ – k^2*y = f(x). This approach yields hyperbolic instead of trig functions.
(3) Only uniformly distributed loads are valid. For other continuous loads, use the basic functions in f(x) above. If concentrated loads are internal, it is most convenient to add nodes at these places.
(4) The moments above, representing the equilibrating moment at the end of the beam segment taken at any point along the beam, are positive CCW. The deflections are positive in the +y direction.

Example 12

Using the beam described in example 10, find moments and displacements for axial loads of = 0, 150, 300, and 450 kip. Compare the results with AISC Benchmark Problem Case 1. Compute both without and with shear displacements. See Example 12 in section 5.2.1 for P =0.

Solution

The following results are found :

w/0 shear displacements :

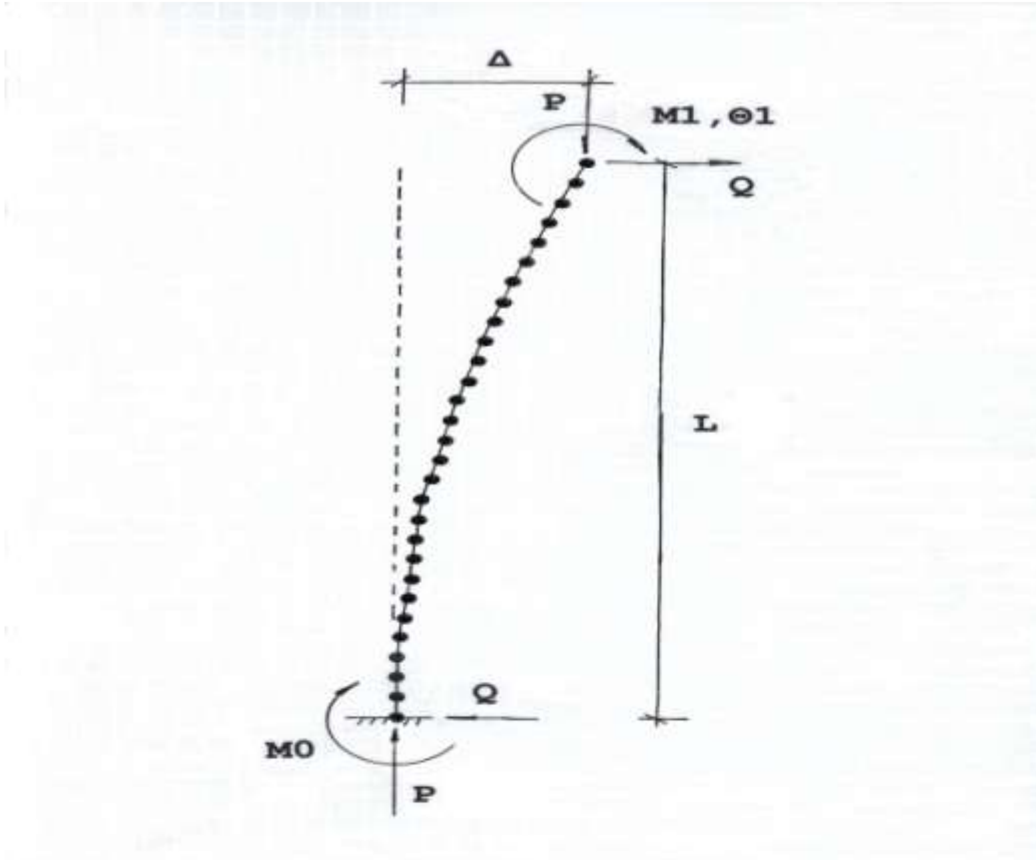
P(kip)	AISC M(k″)/Δ(in.)	mark1.c M(k″)/Δ(in.)	difference
0	235/.197	235/.197	0/0
150	269/.224	269/.224	0/0
300	313/.261	313/.260	0/0.38%
450	375/.311	375/.311	0/0

With shear displacements :

P(kip)	AISC M(k″)/Δ(in.)	mark1.c M(k″)/Δ(in.)	difference
0	235/.202	235/.201	0/0.50%
150	270/.230	270/.229	0/0.43%
300	316/.269	316/.267	0/0.74%
450	380/.322	380/.319	0/0.93%

5.3 CANTILEVER BEAM-COLUMN

Examples of cantilever beam-columns are large elevated off-highway signs and construction cranes. Reference 10 by Dr. Aristizabal-Ochoa presents a powerful method for analyzing these structures, using the slope-deflection equations. His paper includes both semi-rigid end conditions and tensile loads, both of which are not shown here. We consider a fixed base, prismatic beam, with a concentrated transverse load at the tip, vertical compression, and optional consideration of shear displacements and end moments.



The basic slope-deflection equations are:

M0 = S11*(E*I/L)*(Θ0-Δ/L) + S12*(E*I/L)*(Θ1-Δ/L)
M1 = S21*(E*I/L)*(Θ0-Δ/L) + S22*(E*I/L)*(Θ1-Δ/L) and:

S11 = S22 = (1-β*Φ/tanΦ)/den
S12 = S21 = (β*Φ/sinΦ-1)/den
den = L*sqrt(P/β*E*I)
If shear displacements included, β = 1/(1+P/G*Ashr)
If not, β = 1

For a cantilever, Θ0 = 0 , obtaining

M0 = S11*(E*I/L)*(-Δ/L) + S12*(E*I/L)*(Θ1-Δ/L)
M1 = S12*(E*I/L)*(-Δ/L) + S11*(E*I/L)*(Θ1-Δ/L)

Taking moment about node 1,
M0 + M1 + P*Δ + Q*L = 0

This gives three equations in three unknowns, M0,Δ, and Θ1
In matrix form we have

```
┌                                           ┐ ┌    ┐   ┌        ┐
│ 1  +E*I*(S11+S12)/L^2   -E*I*S12/L │ │ M0 │   │   0    │
│ 0  -E*I*(S11+S12)/L^2   +E*I*S11/L │*│ Δ  │ = │  +M1   │
│ 1         P                  0       │ │ Θ1 │   │-M1-Q*L│
└                                           ┘ └    ┘   └        ┘

┌    ┐                ┌        ┐
│ M0 │                │ 0      │
│ Δ  │ = [Kc]^(-1)*│ +M1    │ → [unknowns]=[Kc]'*[knowns]
│ Θ1 │                │-M1-Q*L│
└    ┘                └        ┘
```

These equations are solved by the program mark2.c, attached as Appendix 2.

Example 13

Given a cantilever beam-column with the following values:

As = 4.6886 in.^2
E = 29000 ksi
G = 11000 ksi
L = 336 in.
Q = 1 kip

Use compressive loads of 100, 150, and 200 kips to find base moments and tip deflections, with shear displacements, by program mark2.c. Moments and deflections for the P=0 case as M = P*L, Δ = P*L^3/3*E*I. Compare these results with Benchmark Case 2 in Reference 2 (2010 AISC Steel Code).

Solution

Shear displacements not used :

P(kip)	AISC M(k-")/Δ(in.)	mark2.c M(k-in.)/Δ(in.)	Difference
0	336 / .901	336 / .899	0/0
100	469 / 1.33	469 / 1.33	0/0
150	598 / 1.75	597 / 1.74	0.17%/0.57%
200	848 / 2.56	844 / 2.54	0.47%/0.78%

Shear displacements used :

P(kip)	AISC M(k-")/Δ(in.)	mark2.c M(k-in.)/Δ(in.)	Difference
0	336 / .907	336 / .906	0/0
100	470 / 1.34	470 / 1.34	0/0
150	601 / 1.77	600 / 1.76	0.17%/0.56%
200	856 / 2.60	853 / 2.59	0.35%/0.38%

5.4 SOLUTION BY REPEATED APPROXIMATIONS

To this point, all equations have been solved explicitly. However, when calculating a complete structure by the finite element method which includes P-Δ and P-δ effects, a series of approximations are used which converge on the correct solution. Generally these approximations are used in an iterative process, finally obtaining a result with error less than some chosen number. Two methods of doing this are the Newton-Raphson method and repeated substitution of the updated variable in the defining equation or equations.

The Newton-Raphson method uses the following algorithm :
x(n+1) = x(n) – f(x(n))/f′(x(n)) where
f(x) = 0 at balance
x(n) = nth approximation
x(n+1) = (n+1)th approximation
f′(x) = df(x)/dx

As an example of this method, use x = sqrt(1+x+cos(x))
f(x) = x – sqrt(1+x+cos(x))
f′(x) = 1 – (1-sin(x))/2*sqrt(1+x+cos(x))
with the following results, the criterion being the iteration stops when the value, to six decimal places, of x(n) does not change.

n	x(n)	f(x(n)	f′(x(n))
1	0.000000	-1.414214	0.646447
2	2.187673	0.572376	0.942948
3	1.580666	-0.022705	0.999835
4	1.603371	-0.000001	0.999835
5	1.603372	-0.000000	0.999835
6	1.603372	0.000000	0.999835

The second method is that of repeated substitution. In this case, the algorithm is
x(n+1) = sqrt(1+x(n)+sin(x(n))), with the following results

n	x(n)
1	0.000000
2	1.414214
3	1.603171
4	1.603372
5	1.603372

This does not mean that the second method is superior, or that it always converges faster than the Newton-Raphson method, but does show that it is probably competitive. It has the strong advantage that no differentiation is required, especially important when matrix equations are involved.

5.5 FINITE ELEMENT ANALYSIS OF FRAMES

The procedures given in sections 5.2 and 5.3 above are useful for analysis of single beams, and for determining maximum intermediate moment in the case of section 5.2. For analysis of a larger structure, perhaps containing many members, a finite element matrix which contain terms combining axial load effects, shear displacements, semi-rigid end connections, and fixed-end forces. Such a matrix is available in Reference 11, by Drs. Gorgun, Yilmaz, and Karacan. The reference contains terms for all three conditions of P, namely compression, no axial force, and tension. In general terms the two-dimensional stiffness matrix is:

$$\begin{bmatrix} k00 & k01 & k02 & k03 & k04 & k05 \\ & k11 & k12 & k13 & k14 & k15 \\ & & k22 & k23 & k24 & k25 \\ & & & k33 & k34 & k35 \\ \text{symmetric} & & & & k44 & k45 \\ & & & & & k55 \end{bmatrix}$$

and the individual terms for the compressive case are the rather formidable equations :

```
k00 = k33 = +E*A/L
k03 = k30 = -E*A/L
k11 = k44 = +E*I*Ψ^3*δ^2*((1-Ψ^2*β1*β2)*sinΨ+
             Ψ*(β1+β2)cosΨ)/L^3*Ω
k14 = k41 = -E*I*Ψ^3*δ^2*((1-Ψ^2*β1*β2)*sinΨ+
             Ψ*(β1+β2)cosΨ)/L^3*Ω
k12 = k21 = +E*I*Ψ^2*δ*(Ψ*β2*sinΨ-cosΨ+1))/L^2*Ω
k24 = k42 = -E*I*Ψ^2*δ*(Ψ*β2*sinΨ-cosΨ+1))/L^2*Ω
k15 = k51 = +E*I*Ψ^2*δ*(Ψ*β1*sinΨ-cosΨ+1))/L^2*Ω
k45 = k54 = -E*I*Ψ^2*δ*(Ψ*β1*sinΨ-cosΨ+1))/L^2*Ω
k22       = +E*I*Ψ*((1+Ψ^2*δ*β2)*sinΨ-Ψ*δ*cosΨ)
k25 = k52 = +E*I*Ψ*(Ψ*δ-sinΨ)
k55       = +E*I*Ψ*((1+Ψ^2*δ*β1)*sinΨ-Ψ*δ*cosΨ)
Ω         = Ψ*(δ*(Ψ^2*β1*β2-1)+β1+β2)*sinΨ
            -(2+Ψ^2*δ*(β1+β2))*cosΨ+2
```

where

```
β    = E*I/L^2*G*Ashr, shear displacement factor
β1   = c0*E*I/L, semi-rigid factor at node 0
β2   = c1*E*I/L, semi-rigid factor at node 1
Ψ    = L*sqrt((P/E*I)/(1-P/G*Ashr))
δ    = 1-P/G*Ashr
```

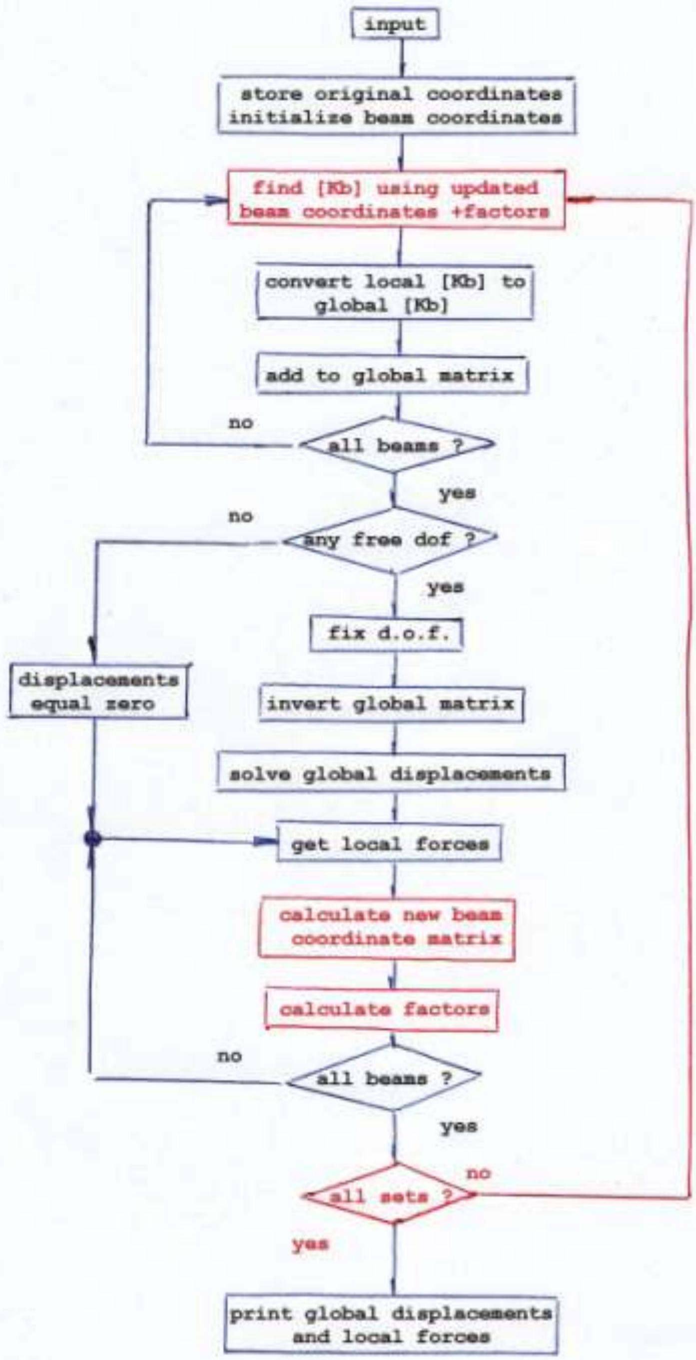
and the fixed-end moments for a uniform distributed load :

```
m0   = +ω*L*L*(1+Ψ^2*β)*(Ψ*(4+Ψ^2*(β+β2))*sinΨ
       +(4-Ψ^2*(1-4*β-2*β2))*cosΨ
       -(4+Ψ^2*(1+4*β*2*β2)))/2*Ψ^2*Ω
m1   = -ω*L*L*(1+Ψ^2*β)*(Ψ*(4+Ψ^2*(β+β1))*sinΨ
       +(4-Ψ^2*(1-4*β-2*β1))*cosΨ
       -(4+Ψ^2*(1+4*β*2*β1)))/2*Ψ^2*Ω
```

Reference 12, by Drs. Ekhande, Selvappalam, and Madugula provides a more detailed entry for the axial terms in the stiffness matrix, i.e. k00, k33, k03, and k30. This term now becomes s1*E*A/L instead of E*A/L.

```
s1   = 1/(1+E*A*Hz/(4.0+P^3*L^2))
Hz   = Ψ*(M0^2+M1^2)*(1/tanΨ+Ψ/(sinΨ)^2)-2*(M0+M1)^2
       +2*Ψ*M0*M1*(1/sinΨ)*(1+Ψ/tanΨ)
```

The equations above in this section are incorporated into an iterative finite element program, frmx.c. The logic diagram of this program is now shown on the next page, with the actions corresponding to a second order analysis, but not to a first order analysis, shown in red.
The front end and main function of frmx.c are shown as appendix 5. As shown there, the input file was slightly changed to accommodate only uniform distributed load fixed end forces. The new term, number of sets, refers to the number of iterations, i.e., complete system calculations. The number of sets required for closure will vary from structure to structure. If the number of sets is made equal to one, 1st order analysis results.


input
store original coordinates
initialize beam coordinates
find [Kb] using updated
beam coordinates +factors
convert local [Kb] to
global [Kb]
add to global matrix
no
all beams ?
yes
no
any free dof ?
yes
fix d.o.f.
displacements
equal zero
invert global matrix
solve global displacements
get local forces
calculate new beam
coordinate matrix
calculate factors
no
all beams ?
yes
no
all sets ?
yes
print global displacements
and local forces

Notes on logic diagram :

(1) all beams = preceding operation done for all beams.

(2) After each beam stiffness matrix [Kb] and fixed end force vector calculated, it is stored in a master matrix for all beams.

(3) all sets = number of iterations accomplished. One set programmed = first order analysis.

Example 14

Compare frmx.c with AISC displacements and moments.

Solution

Shear displacements not used :

P(kip)	Benchmark 1 AISC	Benchmark 1 frmx.c	P(kip)	Benchmark 2 AISC	Benchmark 2 frmx.c
0	235/.197	235/.197	0	336/.901	336/.899
150	269/.224	269/.224	100	469/1.33	469/1.33
300	313/.261	313/.260	150	598/1.75	597/1.74
450	375/.311	374/.309	200	848/2.56	844/2.54

Shear displacements used :

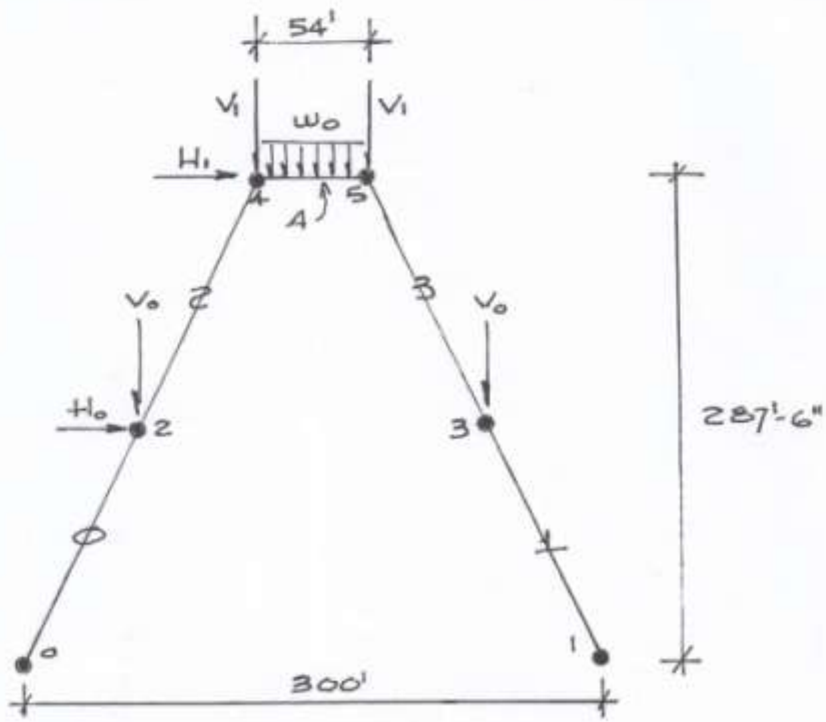
P(kip)	Benchmark 1 AISC	Benchmark 1 frmx.c	P(kip)	Benchmark 2 AISC	Benchmark 2 frmx.c
0	235/.202	235/.201	0	336/.907	336/.906
150	270/.230	270/.230	100	470/1.34	470/1.34
300	316/.269	315/.268	150	601/1.77	600/1.76
450	380/.322	379/.321	200	856/2.60	853/2.59

As a final example, consider the structure shown on the next page. It is in a coal mine in Dortmund, Germany, and has been retired since 1986, but preserved as a historical structure. In operation it lifted 70 tons of payload from a shaft 3300 deep (see reference 8). The structure consists of two trapezoidal frames, braced together for lateral stability.

Assumed parameters :

Ix = 219219 in.^4 A = 231 in.^2 Ashr = 123 in.^2
E = 29000 ksi G = 11000 ksi
V0 = 149 kip V1 = 120 kip
H0 = 10 kip H1 = 10 kip w0 = 1 kip/ft

Use first and second order analyses, with pinned supports and shear effects for comparison.


54'
V_1
V_1
w_o
H_1
4
5
A
2
3
V_o
V_o
H_o
2
3
287'-6"
0
1
300'

In the second order analysis, it took four (4) total iterations for two successive calculations sets to have identical stiffness matrix factors to seven (7) decimal places and five (5) total iterations for two successive calculation sets to have identical displacements to seven (7) decimal places.

Beam	Axial (kip)	1st order M0/M1 (k")/(k")	2nd order M0/M1 (k")/(k")	difference
0	+286	0/+45691	0/+46946	0/+2.75%
1	+321	0/-32409	0/-32900	0/+1.52%
2	+161	-45961/-29187	-46946/-30143	+2.75%/+3.28%
3	+192	+32409/+38502	+32900/+39715	+1.52%/+3.15%
4	+111	+29187/-38502	+30143/-39715	+3.28%/+3.15%

It is seen that, for this case, an average increase of approximately 3% is obtained between first and second order analysis.

It should be noted :

(1) Different ratios of 1st order to 2nd order results obtain with different structures and loadings.
(2) Each load combination must be investigated separately.
(3) See the next section for modification of E, G, and additional notional loads to simulate changes in modulus with load and for structure imperfections

6. FURTHER CONSIDERATIONS

6.1 Geometric Imperfections

These are either modeled directly or with the use of fictional "notional" lateral loads as :

Ni = 0.002*Yi where

Ni = lateral load at the ith level, applied in the most destabilizing direction

Yi = LRFD gravity load at the ith level

6.2 Stiffness Adjustments

For flexural stiffness, use 0.80*τb*E (and G) where

τb = 1.0 if Pr/Py <= 0.5, else 4*(Pr/Py)*(1-Pr/Py)

For other stiffness multiply use 0.80*E.

6.3 Another Benchmark

Figure C-C2.4 on page 277 of the 2010 steel code presents a W10x60 cantilever beam, major axis bending, fully braced out-of-plane, with

L = 180 in. I = 341 in.^4 A = 17.6 in.^2

Ashr = d*t =4.2924 in.^2 (assumed)

E = 0.8*29000 = 23200 ksi

G = 8920 ksi

with results for a rigorous P-Δ and P-δ analysis.

program	m	delta	difference
AISC	1394 k"	2.22"	not applicable
mark2.c	1400 k"	2.23"	+0.43%/+0.45%
frmx.c	1392 k"	2.21"	-0.14%/-0.45%

6.4 Calculation of Available Strengths

AISC Chapter	Subject
B	Design requirements
D	Tension members
E	Compression members
F	Flexural members
G	Design of members for shear
H	Combined forces and torsion
J	Connections
K	HSS and box members
L	Serviceability

REFERENCES

1. "International Building Code", 2012 Edition http://publicecodes.citation.com
2. "Specification for Structural Steel Buildings", ANSI/AISC 360-10, www.aisc.org
3. "Structures Design Guidelines for Load and Resistance Factor Design", Topic No. 625-020-150-c, Florida DOT, www.dot.state.fl.us
4. "Detailing Manual for Load and Resistance Factor Design", Topic No. 625-020-200-d, Florida DOT, www.dot.state.fl.us
5. "Concepts and Applications of Finite Element Analysis", R.D. Cook, etal., John Wiley+Sons, 1989, 3rd Edition, ISBN 0471847887, www.barnesandnoble.com
6. "Matrix Analysis of Structures", A. Kassimali, 2nd Edition, CL-Engineering Company, 2012, ISBN 978-1-1114-2622-4, www.barnesandnoble.com
7. Maxima, a Computer Algebra System, www.sourceforge.net
8. "Frames and Arches", Valerian Leontovich, McGraw-Hill, 1959, LC 58-8848, www.amazon.com
9. "Elastic Second-Order Computer Analysis of Beam-Columns and Frames", H.R. Munoz, Master's Thesis, University of Texas at Austin, 1991, www.dtic.mil/dtic
10. "Theory of Elastic Stability", S. Timoshenko and J. Gere, ISBN 0-07-064749-6, www.barnesandnoble.com
11. "Slope-Deflection Equations for Stability and Second-Order Analysis of Timoshenko Beam-Column Structures with Semi-Rigid Conections", J.D. Aristizabal-Ochoa, ScienceDirect Engineering Structures 30(2008)-2517-2527, www.sciencedirect.com
12. "Nonlinear analysis of frames composed of flexibly connected members with rigid end sections accounting for shear deformations", Gorgun, Yilmaz, and Karacan, 19Jul2012, http://www.academicjournals.org/SRE
13. "Stability Functions for Three-Dimensional Beam-Columns", Ekhande, Selvappalam, and Madugula, Journal of Structural Engineering, ASCE, February 1989, pp.467-479, www.asce.org

APPENDIX 1

```
void getksr(double cn,double cf,double mtrx[6][6])
{
        int i,j;
        double den,t00,t01,t03,t11,t12,t13;
        double J[6][6],K[6][6];
        for(i=0;i<=5;i++)
        {
                for(j=0;j<=5;j++)
                {
                        J[i][j] =       mtrx[i][j];
                        K[i][j] =       mtrx[i][j];
                }
        }

        den     =       (1.0+K[2][2]*cn)*(1.0+K[5][5]*cf)-
                         K[2][5]*K[2][5]*cn*cf;

        t00     =       (1.0+K[2][2]*cn)*(1.0+K[5][5]*cf)-
                         K[2][5]*K[2][5]*cn*cf;
        t01     =       K[1][5]*K[2][5]*cn*cf-(1.0+K[5][5]*cf)*K[1][2]*cn;
        t03     =       K[1][2]*K[2][5]*cn*cf-(1.0+K[2][2]*cn)*K[1][5]*cf;
        t11     =       K[2][5]*K[4][5]*cn*cf-(1.0+K[5][5]*cf)*K[2][4]*cn;
        t12     =       (1.0+K[2][2]*cn)*(1.0+K[5][5]*cf)-
                         K[2][5]*K[2][5]*cn*cf;
        t13     =       K[2][4]*K[2][5]*cn*cf-(1.0+K[2][2]*cn)*K[4][5]*cf;

        J[1][1] =       (K[1][1]*t00+K[1][2]*t01+K[1][5]*t03)/den;
        J[1][2] =       (K[1][2]*t00+K[2][2]*t01+K[2][5]*t03)/den;
        J[1][4] =       (K[1][4]*t00+K[4][2]*t01+K[4][5]*t03)/den;
        J[1][5] =       (K[1][5]*t00+K[5][2]*t01+K[5][5]*t03)/den;

        J[2][1] =       (K[1][2]*(1.0+K[5][5]*cf)-K[1][5]*K[2][5]*cf)/den;
        J[2][2] =       (K[2][2]*(1.0+K[5][5]*cf)-K[2][5]*K[2][5]*cf)/den;
        J[2][4] =       (K[4][2]*(1.0+K[5][5]*cf)-K[4][5]*K[2][5]*cf)/den;
        J[2][5] =       (K[5][2]*(1.0+K[5][5]*cf)-K[5][5]*K[2][5]*cf)/den;

        J[4][1] =       (K[1][2]*t11+K[1][4]*t12+K[1][5]*t13)/den;
        J[4][2] =       (K[2][2]*t11+K[2][4]*t12+K[2][5]*t13)/den;
        J[4][4] =       (K[4][2]*t11+K[4][4]*t12+K[4][5]*t13)/den;
        J[4][5] =       (K[5][2]*t11+K[5][4]*t12+K[5][5]*t13)/den;

        J[5][1] =       (K[1][5]*(1.0+K[2][2]*cn)-K[2][5]*K[1][2]*cn)/den;
        J[5][2] =       (K[2][5]*(1.0+K[2][2]*cn)-K[2][5]*K[2][2]*cn)/den;
        J[5][4] =       (K[4][5]*(1.0+K[2][2]*cn)-K[2][5]*K[4][2]*cn)/den;
        J[5][5] =       (K[5][5]*(1.0+K[2][2]*cn)-K[2][5]*K[5][2]*cn)/den;

        for(i=0;i<=5;i++)
        {
                for(j=0;j<=5;j++)
                {
                        mtrx[i][j]      =       J[i][j];
                }
        }
}
```

APPENDIX 2 – [ISR] MATRIX

DEN	**0**	**0**	**0**	**0**	**0**
0	**DEN**	**k52*k15*c0*c1-k12*c0*(1+k55*c1)**	**0**	**0**	**k12*k45*c0*c1-k15*c1*(1+k22*c0)**
0	**0**	**1+k55*c1**	**0**	**0**	**-k25*c1**
0	**0**	**0**	**DEN**	**0**	**0**
0	**0**	**k52*k45*c0*c1-k42*c0*(1+k55*c1)**	**0**	**DEN**	**k42*k25*c0*c1-k45*c1*(1+k22*c0)**
0	**0**	**-k52*c0**	**0**	**0**	**1+k22*c0**

All terms divided by DEN where :

DEN = (1+k22*c0)*(1+k55*c1)-k25*k52*c0*c1

APPENDIX 3

```
/*************************************************************************
 *                                                                        *
 * mark1.c : 05-24-13 : ml : general purpose beam-column program given :  *
 *                                                                        *
 * shear        =       1 for inclusion of shear displacement             *
 * As           =       effective shear area, taken as d*tw, in.^2        *       *
 * E            =       modulus of elasticity, kip/in.^2                  *
 * I            =       moment of inertia, in.^4                          *
 * L            =       beam-column length, in.                           *
 * M0           =       moment at end 0, kip-in.                          *
 * M1           =       moment at end 1, kip-in.                          *
 * Pp           =       axial force, original coordinates, kip            *
 * Vp           =       shear at end 0, original coordinates, kip         *
 * wp           =       unifrom distributed load, orginal coordinates,kip/ft *
 * xa,xb,ya,yb  =       displacements from original coordinates, in.      *
 *                                                                        *
 * Reference    =       "Elastic Second-Order Computer Analysis of Beam-  *
 *                      Columns and Frames", H.R. Munoz, Master's Thesis, *
 *                      University of Texas at Austin, 1991,              *
 *                      www.dtic.mil/dtic                                 *
 *                                                                        *
 *************************************************************************/

#include<math.h>
#include<stdio.h>
#include<stdlib.h>

int main(void)
{
        int shear;                                          // first input
        double As,E,G,I,L,M0,M1,Pp,Vp,wp,xa,xb,ya,yb;       // 14 inputs
        double A,B,C0,C1,delta,deltasquared,k,m,nu,P0,
               s,theta,V0,w0,x,y,z;
        FILE *inn;
        FILE *out;
        inn     =       fopen("mark1.in","r");
        out     =       fopen("mark1.out","w+");
        fscanf(inn,"%i %lf %lf %lf %lf %lf %lf %lf",
                    &shear,&As,&E,&G,&I,&L,&M0,&M1);
        fscanf(inn,"%lf %lf %lf %lf %lf %lf %lf",
                    &Pp,&Vp,&wp,&xa,&xb,&ya,&yb);
        fclose(inn);
        wp      =       wp/12.0;

        deltasquared   =       (xb-xa)*(xb-xa)+(yb-ya)*(yb-ya);
        if(deltasquared>0)
        {
                delta   =       sqrt(deltasquared);
                s       =       L+delta/2.0;
                theta   =       asin(2.0/(L*L))*
                                sqrt(s*(s-L)*(s-L)*(s-delta));
                w0      =       wp*cos(theta);
                P0      =       Pp*cos(theta)-Vp*sin(theta)+
                                (w0/2.0)*sin(theta);
                V0      =       Pp*sin(theta)+Vp*cos(theta);
        }
        else
        {
                w0      =       wp;
                P0      =       Pp;
                V0      =       Vp;
        }

        fprintf(out,"   x(in.)       m(kip-in.)    y(in.)\n");
        fprintf(out,"  -------    --------------   -------\n");

        for(x=0.0;x<=L+0.01;x+=L/20.0)
        {
```

```
                if(shear!=1)
                {
                        k       =       +sqrt(P0/(E*I));
                        A       =       -(M1-M0*cos(k*L))/(P0*sin(k*L))
                                        +w0*tan(k*L/2.0)/(P0*k*k);
                        B       =       +(-M0+w0/(k*k))/P0;
                        C0      =       -A*sin(k*L)/L+B*(1.0-cos(k*L))/L
                                        -w0*L/(2*P0);
                        m       =       -P0*A*sin(k*x)-P0*B*cos(k*x)
                                        +w0/(k*k);
                        y       =       A*sin(k*x)+B*cos(k*x)
                                        +w0*x*x/(2.0*P0)+C0*x-B;
                }
                else
                {
                        z       =       +1.0-P0/(G*As);
                        k       =       +sqrt(P0/(z*E*I));
                        nu      =       (M0+M1)/(L*L)+w0/2.0-V0/L;
                        A       =       -z*(M1-M0*cos(k*L))/(P0*sin(k*L))
                                        +z*E*I*w0*tan(k*L/2.0)/(P0*P0);
                        B       =       +z*(-M0+w0*E*I/P0)/P0;
                        C1      =       -B;
                        C0      =       -A*sin(k*L)/L+B*(1.0-cos(k*L))/L
                                        -w0*L/(2*P0);
                        m       =       -P0*A*sin(k*x)/z-P0*B*cos(k*x)/z
                                        +E*I*w0/P0;
                        y       =       A*sin(k*x)+B*cos(k*x)+w0*x*x/(2*P0)
                                        +C0*x+C1;
                        y       =       y-nu*x*(x-L)/(2.0*G*As);
                }
                fprintf(out,"%9.3f %16.6e %9.3f\n",x,m,y);
        }

        fclose(out);
        return 0;
}
```

APPENDIX 4

```
/**********************************************************************
 *                                                                    *
 *  mark2.c : 2nd order cantilever with optional shear displacements  *
 *            and end point transverse and moment loads: 5-24-13 : ml *
 *                                                                    *
 *  Reference : "Slope-deflection equations for stability and second- *
 *               order analysis of Timoshenko beam-column strutures   *
 *               witn semi-rigid connections", J.D. Aristizabal-Ochoa,*
 *               2008,www.elsevier.com/locate/engstruct               *
 *                                                                    *
 **********************************************************************/

 #include<math.h>
 #include<stdio.h>
 #include<stdlib.h>
 #include"hdrinv3.h"

 int main(void)
 {
        int gi,gj,shear_no;
        double Ashr,c0,c1,E,G,Iz,L,M1,P,Q,shear_num;    // inputs
        double dlt,gden,gwrkf,M0,theta1,s11,s12,s21,s22;
        double knowns[3],TM[3][3];
        double *ginput_file,*gSij;
        void invert3x(double [3][3]);
        void getSij(double *,double *);
        FILE *inn;
        FILE *out;

        ginput_file     =       calloc(11,sizeof(double));
        gSij            =       calloc(4,sizeof(double));
        inn             =       fopen("mark2.in","r");
        out             =       fopen("mark2.out","w+");

        for(gi=0;gi<=12;gi++)
        {
                fscanf(inn,"%lf",&gwrkf);
                *(ginput_file+gi) =       gwrkf;
        }
        fclose(inn);

        E               =       *(ginput_file+ 3);
        Iz              =       *(ginput_file+ 5);
        L               =       *(ginput_file+ 6);
        M1              =       *(ginput_file+ 7);
        P               =       *(ginput_file+ 8);
        Q               =       *(ginput_file+ 9);

        getSij(ginput_file,gSij);

        s11        =       *(gSij+0);
        s12        =       *(gSij+1);
        s21        =       *(gSij+2);
        s22        =       *(gSij+3);

        TM[0][0]   =      +1.0;
        TM[0][1]   =      +E*Iz*(s11+s12)/(L*L);
        TM[0][2]   =      -E*Iz*s12/L;
        TM[1][0]   =       0.0;
        TM[1][1]   =      -E*Iz*(s21+s22)/(L*L);
        TM[1][2]   =      +E*Iz*s22/L;
        TM[2][0]   =      +1.0;
        TM[2][1]   =      +P;
        TM[2][2]   =       0.0;

        knowns[0]  =       0.0;
        knowns[1]  =       +M1;
        knowns[2]  =       -M1-Q*L;
```

```
        invert3x(TM);

        M0=dlt=theta1=0.0;

        for(gi=0;gi<=2;gi++)
        {
                M0     +=      TM[0][gi]*knowns[gi];
                dlt    +=      TM[1][gi]*knowns[gi];
                theta1 +=      TM[2][gi]*knowns[gi];
        }

        fprintf(out,"M0     = ");fprintf(out,"%16.6e",M0);
        fprintf(out,"  kip-in.\n");
        fprintf(out,"dlt    = ");fprintf(out,"%16.6e",dlt);
        fprintf(out,"  in.\n");
        fprintf(out,"theta1 = ");fprintf(out,"%16.6e",theta1);
        fprintf(out,"  radians\n");

        fclose(out);
        return 0;
}

void getSij(double *input_file,double *Sij)
{
        int shear_no;
        double Ashr,c0,c1,E,G,Iz,L,P,shear_num;
        double beta,den,psi,r0,r1;

        Ashr            =       *(input_file+0);
        c0              =       *(input_file+1);
        c1              =       *(input_file+2);
        E               =       *(input_file+3);
        G               =       *(input_file+4);
        Iz              =       *(input_file+5);
        L               =       *(input_file+6);
        P               =       *(input_file+8);
        shear_num       =       *(input_file+10);

        shear_no        =       shear_num;

        if(shear_no!=0)
        {
                beta    =       1.0/(1.0+P/(G*Ashr));
                psi     =       L*sqrt(P/(beta*E*Iz));
        }
        else
        {
                beta    =       1.0;
                psi     =       L*sqrt(P/(E*Iz));
        }

        r0              =       1.0/(1.0+3.0*c0*E*Iz/L);
        r1              =       1.0/(1.0+3.0*c1*E*Iz/L);

        den             =       (1.0-r0)*(1.0-r1)*beta*psi*psi+
                                3.0*(r0+r1-2.0*r0*r1)*(1.0-beta*psi/tan(psi))+
                                9.0*r0*r1*(tan(psi/2.0)/(psi/2.0)-beta);
        *(Sij+0)        =       (3.0*r0*(1.0-r1)*beta*psi*psi+
                                 9.0*r0*r1*(1.0-beta*psi/tan(psi)))/den;
        *(Sij+1)        =       9.0*r0*r1*(beta*psi/sin(psi)-1.0)/den;
        *(Sij+2)        =       9.0*r0*r1*(beta*psi/sin(psi)-1.0)/den;
        *(Sij+3)        =       (3.0*r1*(1.0-r0)*beta*psi*psi+
                                 9.0*r0*r1*(1.0-beta*psi/tan(psi)))/den;
}
```

APPENDIX 5

```
/*************************************************************************
 *                                                                       *
 * frmx.c: second order beam analysis with stability factors,semi-rigid, *
 * shear terms, fixed end forces and uniform loads : ml : 05-24-13       *
 *                                                                       *
 * input file = frmx.in                                                  *
 *              no nodes,beams,beamprops,fixed dof,nodeloads,sets (6 it.) *
 *              2d node matrix (2 x number nodes)                        *
 *              beam connection matrix (3 x number beams)                *
 *              beamprop matrix(A,Ashr,Iz,E,G,shear_no,c0,c1,w0 (9 items) *
 *              fixed d.o.f vector (1 x number fixed d.o.f.)             *
 *              node load d.o.f. (1 x number node loads)                 *
 *              node load values (1 x number node loads)                 *
 *                                                                       *
 * output files = frmx.out                                               *
 *                node number  - x disp. - y disp. - rotation            *
 *                               (global coordinates)                    *
 *                beam number  - near node - fx fy M (local coordinates) *
 *                             - far node  - fx fy M (  "          "    ) *
 *                                                                       *
 * screen output = displacements each set for finding convergence        *
 *                                                                       *
 * Ref.1     = "Nonlinear analysis of frames composed of flexibly        *
 *              connected members with rigid end sections accounting for *
 *              shear deformations", Gorgun, Yilmaz, and Karacan,        *
 *              19Jul2012,http://www.academicjournals.org/SRE            *
 * Ref.2    =  "Stability Functions for Three-Dimensional Beam-Columns", *
 *              Ekhande, Selvappalam, Madugula, Journal of Structural    *
 *              Engineering (ASCE), February 1989, pp. 467-479           *
 *                                                                       *
 * Note : hdri6.h must be in same directory as frmx.exe                  *
 *                                                                       *
 *************************************************************************/


#include<math.h>
#include<stdio.h>
#include<stdlib.h>
#include"hdri6.h"
int main(void)
{
                        /* SCALAR DECLARATIONS */

        int gi,gj,gk,grank;
        int gno_nodes;
        int gno_beams;
        int gno_beamprops;
        int gno_fixed;
        int gno_nodeloads;
        int gno_sets;

                        /* VECTOR DECLARATIONS */

        int *gfixed_vek;
        int *gnodeload_dof;
        double *gnodeload_val;
        double *gloads;

                        /* MATRIX DECLARATIONS */

        int *gbeamconn_mtrx;
        double gKBlocal[6][6];
        double gTB[6][6];
        double gITB[6][6];
        double gKBglobal[6][6];
        double *gcoord_mtrx;
        double *gbeamprop_mtrx;
```

```
double *gglob_mtrx;
double *gred_mtrx;
double *gdisp_mtrx;
double *gbeamforce_mtrx;
double *gtkb;           // all local beam k after shear + sr
double *gtotalTB;       // all global to local
double *gtotalITB;      // all local to global
double *gfactor_mtrx;   // all beam stability factors
double *gbcoor_mtrx;    // revised coordinates for K calculation
double *gfff_val;       // all fixed end forces in local
                        // coordinates

                /* FUNCTION PROTYPES */

void input1(int *,int *,int *,int *,int *,int *);
void input2(int,int,int,int,int,int,double *,int *,double *,
            int *,int *,double *);
void findinitialmatrices(int,double*,int *,double *,double *,
                        double *);
void findKBlocal(int,double [6][6],double *,double *);
void findTBmatrix(int,double *,int *,double [6][6],
                  double [6][6],double *,double*);
void findKBglobal(double [6][6],double [6][6],double [6][6],
                  double [6][6]);
void add_bk(int,int,int *,double [6][6],double *);
void getloads(int,int,int,int *,int *,double *,double *,double *,
              double *);
void fix_dof(double *,int *,double *,int,int);
void invert(int,double*);
void solve_global_displacements(int,int,int *,double *,double *,
                               double*);
void getlocalforces(int,int *,double *,double *,double *,
                    double *,double *);
void revko(int,int,double *,double *,double *);
void findfactors(int,double *,int *,double *,double *,double *,
                 double *);
void prn_out(double *,double *,int,int);

                /* MAINLINE PROGRAM */

input1(&gno_nodes,&gno_beams,&gno_beamprops,&gno_fixed,
       &gno_nodeloads,&gno_sets);

if(gno_sets<1)
{
        printf("\n");
        printf("sets less than 1\n");
        exit(0);
}
else
{
        ;
}

grank           = 3*gno_nodes-gno_fixed;
gcoord_mtrx     = calloc(gno_nodes*2,sizeof(double));
gbeamconn_mtrx  = calloc(gno_beams*3,sizeof(int));
gbeamprop_mtrx  = calloc(gno_beamprops*9,sizeof(double));
gfixed_vek      = calloc(gno_fixed,sizeof(int));
gnodeload_dof   = calloc(gno_nodeloads,sizeof(int));
gnodeload_val   = calloc(gno_nodeloads,sizeof(double));
gglob_mtrx      = calloc((3*gno_nodes)*(3*gno_nodes),
                        sizeof(double));
gred_mtrx       = calloc(grank*grank,sizeof(double));
gdisp_mtrx      = calloc(3*gno_nodes,sizeof(double));
gbeamforce_mtrx = calloc(6*gno_beams,sizeof(double));
gtkb            = calloc(36*gno_beams,sizeof(double));
gtotalTB        = calloc(36*gno_beams,sizeof(double));
gtotalITB       = calloc(36*gno_beams,sizeof(double));
gloads          = calloc(3*gno_nodes,sizeof(double));
gfactor_mtrx    = calloc(7*gno_beams,sizeof(double));
```

```
gbcoor_mtrx     = calloc(2*gno_nodes,sizeof(double));
gfff_val        = calloc(6*gno_beams,sizeof(double));

input2(gno_nodes,gno_beams,gno_beamprops,gno_fixed,
       gno_nodeloads,gno_sets,gcoord_mtrx,gbeamconn_mtrx,
       gbeamprop_mtrx,gfixed_vek,gnodeload_dof,gnodeload_val);

for(gi=0;gi<=gno_beams-1;gi++)
{
        findinitialmatrices(gi,gcoord_mtrx,gbeamconn_mtrx,
                            gbeamprop_mtrx,gfactor_mtrx,gfff_val);
}

for(gi=0;gi<=2*gno_nodes-1;gi++)
{
        *(gbcoor_mtrx+gi)       =       *(gcoord_mtrx+gi);
}

for(gk=1;gk<=gno_sets;gk++)
{

        for(gi=0;gi<=3*gno_nodes-1;gi++)             // all rows
        {
                for(gj=0;gj<=3*gno_nodes-1;gj++)     // all cols
                {
                        *(gglob_mtrx+3*gno_nodes*gi+gj)=0.0;
                }
        }

        for(gi=0;gi<=6*gno_beams-1;gi++)
        {
                *(gbeamforce_mtrx+gi)   =       0.0;
        }

        for(gi=0;gi<=gno_beams-1;gi++)
        {
                findKBlocal(gi,gKBlocal,gtkb,gfactor_mtrx);

                findTBmatrix(gi,gcoord_mtrx,gbeamconn_mtrx,gTB,gITB,
                             gtotalTB,gtotalITB);
                findKBglobal(gKBlocal,gTB,gKBglobal,gITB);

                add_bk(gi,gno_nodes,gbeamconn_mtrx,gKBglobal,
                       gglob_mtrx);
        }

        getloads(gno_nodes,gno_beams,gno_nodeloads,gbeamconn_mtrx,
                 gnodeload_dof,gnodeload_val,gtotalITB,gloads,
                 gfff_val);

        if(grank!=0)
        {
                fix_dof(gred_mtrx,gfixed_vek,gglob_mtrx,gno_nodes,
                        gno_fixed);

                invert(grank,gred_mtrx);

                solve_global_displacements(gno_nodes,gno_fixed,
                                           gfixed_vek,gloads,
                                           gred_mtrx,gdisp_mtrx);
        }
        else
        {
                for(gi=0;gi<=3*gno_nodes-1;gi++)
                {
                        *(gdisp_mtrx+gi)        =       0.0;
                }
        }

        for(gi=0;gi<=gno_beams-1;gi++)
        {
```

```
                        getlocalforces(gi,gbeamconn_mtrx,gtkb,gtotalTB,
                                       gfff_val,gdisp_mtrx,gbeamforce_mtrx);
                }

                revko(gk,gno_nodes,gcoord_mtrx,gdisp_mtrx,gbcoor_mtrx);

                for(gi=0;gi<=gno_beams-1;gi++)
                {
                        findfactors(gi,gbcoor_mtrx,gbeamconn_mtrx,
                                    gbeamprop_mtrx,gbeamforce_mtrx,
                                    gfactor_mtrx,gfff_val);
                }
        }

        prn_out(gdisp_mtrx,gbeamforce_mtrx,gno_nodes,gno_beams);

        return 0;
}
```