Finite Element Analysis (Book Draft)

Working Paper · March 2021

DOI: 10.13140/RG.2.2.32391.70560

CITATION
1

READS
6,311

1 author:

Mohammad Tawfik
Academy of Knowledge
83 PUBLICATIONS **654** CITATIONS

SEE PROFILE

Finite Element Analysis

Mohammad Tawfik

Academy of Knowledge

http://academyofknowledge.org

DOI: 10.13140/RG.2.2.32391.70560

Videos explaining these lecture notes are available on http://FEM.AcademyOfKnowledge.org)

Draft printed on: 17. March 2021

Table of Contents

1. Numerical Solution of Differential Equations

(Videos explaining this topic are available on http://FEM.AcademyOfKnowledge.org)

1.1 Classification of Numerical Solutions of Differential Equations

Two main families of approximate methods could be identified in the literature. The discrete coordinate methods and the distributed coordinate methods.

Discrete coordinate methods depend on solving the differential relations at pre-specified points in the domain. When those points are determined, the differential equation may be approximately presented in the form of a difference equation. The difference equation presents a relation, based of the differential equation, between the values of the dependent variables at different values of the independents variables. When those equations are solved, the values of the dependent variables are determined at those points giving an approximation of the distribution of the solution. Examples of the discrete coordinate methods are finite difference methods and the Runge-Kutta methods. Discrete coordinate methods are widely used in fluid dynamics and in the solution of initial value problems.

The other family of approximate methods is the distributed coordinate methods. These methods, generally, are based on approximating the solution of the differential equation using a summation of functions that satisfy some or all the boundary conditions. Each of the proposed functions is multiplied by a coefficient, generalized coordinate, that is then evaluated by a certain technique that identifies different methods from one another. After the solution of the problem, you will obtain a function that represents, approximately, the solution of the problem at any point in the domain.

Stationary functional methods are part of the distributed coordinate methods family. These methods depend on minimizing/maximizing the value of a functional that describes a certain property of the solution, for example, the total energy of the system. Using the stationary functional approach, the finite element model of a problem may be obtained. It is usually much easier to present the relations of different variables using a functional, especially when the relations are complex as in the case of fluid structure interaction problems or structure dynamics involving control mechanisms.

The weighted residual methods, on the other hand, work directly on the differential equations. As the approximate solution is introduced, the differential equation is no more balanced. Thus, a residue, a form of error, is introduced to the differential equation. The different weighted residual methods handle the residue in different ways to obtain the values of the generalized coordinates that satisfy a certain criterion.

1.2 Weighted Residual Methods

1.2.1 Basic Concepts

We may right a general boundary value problem in the form:

$$L(f(x))=g(x)$$

Where *f(x)* is the function of interest, *g(x)* is some arbitrary input to the domain (excitation function), while *L(.)* is a differential operator that may be expanded in the form:

$$L(.)=a_0(x)+a_1(x)\frac{d}{dx}+a_2(x)\frac{d^2}{dx^2}+\ldots a_n(x)\frac{d^n}{dx^n}$$

Which is a general linear nth order differential equation with coefficients that may depend on the independent variable *x*. The above equation has to have certain boundary conditions that render the solution unique. If we select different functions, Ψ_i (which we may call *trial functions*), that satisfy all boundary conditions but do not necessarily satisfy the differential equation, we may write the approximate solution of *f(x)* in the form of:

$$f(x)=\sum_{i=1}^{n} a_i\psi_i(x)$$

Where a_i are the generalized coordinates, or the unknown coefficients (Note the similarity with the interpolation using polynomials). Applying the differential operator on the approximate solution, you get:

$$L(f(x))-g(x)=L\left(\sum_{i=1}^{n} a_i\psi_i\right)-g(x)=\sum_{i=1}^{n} a_i L(\psi_i)-g(x)\neq 0$$

Note that the right hand side of the above equation is not equal to zero. The non-zero value of the right hand side is called the *residue*.

$$\sum_{i=1}^{n} a_i L(\psi_i)-g(x)=R(x)$$

Note:

The *residue* is NOT the *error*.

The error, as you may recall, is the difference between the exact solution and the approximate one. Meanwhile, the residue is the imbalance created in the differential equation due to the use of the approximate solution, this maybe expressed in the form:

$$L\left(\sum_{i=1}^{n} a_i\psi_i-f(x)\right)=R(x)$$

1.2.2 General Weighted Residual Method

When we were solving interpolation problems, the problem of finding the generalized coordinates was a problem of forcing the function to pass by the given points of data. When we were solving the regression problems, our problem was minimizing the error created because we ket the function move *freely* away from the data points. However, in both problems, we had a reference to measure from, namely, *the data points*. In the current case, we have a direct evaluation of the residue, which is related to the error. If we could *do something about the residue*, we may affect the error in a desirable way.

The criterion used in Weighted Residual methods is based on one simple assumption, that is, if we integrate the residue over the domain and force it to equal to zero, then the error will be reduced. Or:

$$\int_{Doomain} R(x)\,dx = 0$$

But is we substitute the expression for the residue, we get:

$$\int_{Doomain} \left(\sum_{i=1}^{n} a_i L(\psi_i(x)) - g(x) \right) dx = 0$$

Which is a single equation in in n unknown generalized coordinates a_i. To find a solution for a linear system of equations in n unknown variables, we need to have n linearly independent equations. In order to generate such a set of equations, we may multiply the integrand in the above equation by n linearly independent functions and perform the integral with each of the functions to generate an equation. These linearly independent functions are called *weighing functions*. Thus we may write:

$$\int_{Doomain} \left(\sum_{i=1}^{n} a_i L(\psi_i(x)) - g(x) \right) w_j(x)\,dx = 0$$

Hence, we may write the jth equation in the form:

$$\int_{Doomain} \left(\sum_{i=1}^{n} a_i L(\psi_i(x)) w_j(x) \right) dx = \int_{Doomain} g(x) w_j(x)\,dx$$

In matrix form, the equations will be:

$$[k_{ij}]\{a_i\} = \{q_j\}$$

Where

$$k_{ij} = \int_{Domain} L(\psi_i(x)) w_j(x)\, dx$$
$$q_j = \int_{Domain} g(x) w_j(x)\, dx$$

1.2.3 Example of a bar problem

The bar tensile problem is a classical problem that describes the relation between the axially distributed loads and the displacement of a bar. Let's consider the bar in Figure 1.2.1 with constant modulus of elasticity and cross section area.

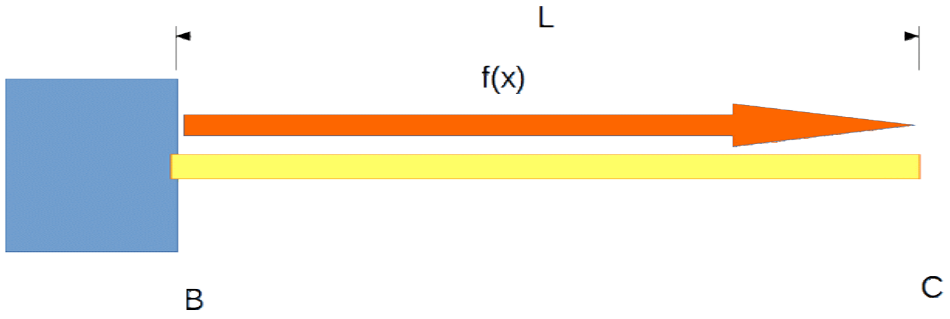

Figure 1.2.1: Simple fixed-free bar with distributed load

The force displacement relation is given by:

$$EA\frac{d^2u}{dx^2} + f(x) = 0$$

Subject to the boundary conditions

$$u(0) = 0$$
$$\frac{du(L)}{dx} = 0$$

Now, lets use the approximate solution

$$u(x) = \sum_{i=1}^{n} a_i \psi_i(x)$$

Substituting it into the differential equation, we get

$$EA\sum_{i=1}^{n} a_i \frac{d^2\psi_i}{dx^2} + f(x) = R(x)$$

Selecting weighting functions, w_j, and applying the method, we get:

$$EA\sum_{i=1}^{n} a_i \int_0^L \left(\frac{d^2\psi_i}{dx^2} w_j \right) dx = -\int_0^L f(x) w_j dx$$

For the boundary conditions to be satisfied, we need a function that has zero value at x=0 and has a slope equal to zero at the free end. Sinusoidal functions are appropriate for this, hence, we may write:

$$u(x) = \sum_{i=1}^{n} a_i \sin\left(\frac{(2i-1)\pi x}{2L} \right)$$

For the purpose of illustration, let's use the first term only (*n=1*). For the weighting function, we may use a polynomial term. The simplest term would be 1, thus:

$$EA\, a \int_0^L \left(\frac{-\pi^2}{4L^2} \sin\left(\frac{\pi x}{2L} \right) \right) dx = -\int_0^L f(x) dx$$

For a constant forcing function, $f(x)=f_o$, performing the integration, we get:

$$\frac{EA\,\pi}{2L} a = f_o L$$

Or:

$$a = \frac{2 f_o L^2}{EA\,\pi} \approx 0.637 \frac{f_o L^2}{EA}$$

Then, the approximate solution for this problem becomes

$$u(x) \approx 0.637 \frac{f_o L^2}{EA} \sin\left(\frac{\pi x}{2L} \right)$$

Now we may compare the obtained solution with the exact one that may be obtained from solving the differential equation. The maximum displacement and the maximum strain may be compared with the exact solution. The maximum displacement is

$$u(L) = 0.637 \frac{f_o L^2}{EA} \left(\text{Exact= } 0.5 \frac{f_o L^2}{EA} \right)$$

And maximum strain is:

$$\frac{du(0)}{dx} = \frac{f_o L}{EA} \left(\text{Exact= } \frac{f_o L}{EA} \right)$$

1.2.4 Collocation Method

The collocation method may be seen as one of the weighted residual family when the weighting function becomes the delta function. The delta function is one that may be described by:

$$\lim_{\epsilon \to 0} \int_{x_0-\epsilon}^{x_0+\epsilon} \delta(x-x_0)\,dx = 1$$

Or, we may write:

$$\begin{aligned}\delta(x-x_0) &= \lim_{\epsilon \to 0} \frac{1}{2\epsilon} (x = x_0) \\ &= 0 (x \neq x_0)\end{aligned}$$

Which leads to the identity:

$$\int_{-\infty}^{\infty} \delta(x-x_0) g(x)\,dx = g(x_0)$$

Now, if we select a set of points x_j inside the domain of the problem, we may write down the integral of the residue, multiplied by the delta functions, as follows:

$$\int_{Domain} R(x)\delta(x-x_j) = R(x_j) = \sum_{i=1}^{n} a_i L(\psi_i(x_j)) - g(x_j) = 0$$

Which gives the equations in the form:

$$[k_{ij}]\{a_i\} = \{q_j\}$$

Where

$$\begin{aligned}k_{ij} &= L(\psi_i(x_j)) \\ q_j &= g(x_j)\end{aligned}$$

1.2.5 Example the bar problem using the collocation method

The bar tensile problem is a classical problem that describes the relation between the axially distributed loads and the displacement of a bar. Let’s consider the bar in Figure 1.2.2 with constant modulus of elasticity and cross section area.

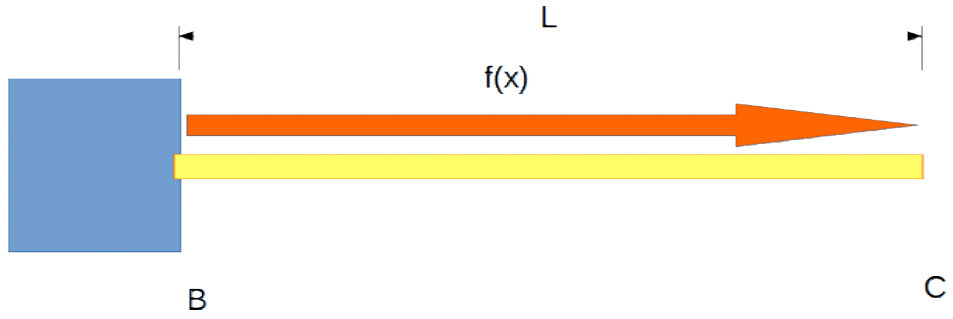

Figure 1.2.2: Simple fixed-free bar with distributed load

The force displacement relation is given by:

$$EA\frac{d^2u}{dx^2}+f(x)=0$$

Subject to the boundary conditions

$$u(0)=0$$
$$\frac{du(L)}{dx}=0$$

Now, lets use the approximate solution

$$u(x)=\sum_{i=1}^{n} a_i\psi_i(x)$$

Substituting it into the differential equation, we get

$$EA\sum_{i=1}^{n} a_i\frac{d^2\psi_i}{dx^2}+f(x)=R(x)$$

Selecting weighting functions to be the Dirac delta function and applying the method, we get:

$$EA\sum_{i=1}^{n} a_i\left(\frac{d^2\psi_i(x_j)}{dx^2}\right)=-f(x_j)$$

For the boundary conditions to be satisfied, we need a function that has zero value at x=0 and has a slope equal to zero at the free end. Sinusoidal functions are appropriate for this, hence, we may write:

$$u(x)=\sum_{i=1}^{n} a_i\sin\left(\frac{(2i-1)\pi x}{2L}\right)$$

For the purpose of illustration, let's use the first term only (*n=1*). The most logical choice for the collocation point would be $x_j=L/2$, thus:

$$EA\,a\left(-\frac{\pi^2}{4L^2}\sin\left(\frac{\pi(L/2)}{2L}\right)\right)=-f(L/2)$$

For a constant forcing function, $f(x)=f_o$, performing the integration, we get:

$$\frac{EA\,\pi^2}{4\sqrt{2}L^2}a=f_o$$

Or:

$$a=\frac{4\sqrt{2}f_oL^2}{EA\,\pi^2}\approx 0.57\frac{f_oL^2}{EA}$$

Then, the approximate solution for this problem becomes

$$u(x)\approx 0.57\frac{f_oL^2}{EA}\sin\left(\frac{\pi x}{2L}\right)$$

Now we may compare the obtained solution with the exact one that may be obtained from solving the differential equation. The maximum displacement and the maximum strain may be compared with the exact solution. The maximum displacement is

$$u(L)=0.57\frac{f_oL^2}{EA}\left(\text{Exact= }0.5\frac{f_oL^2}{EA}\right)$$

And maximum strain is:

$$\frac{du(0)}{dx}=0.9\frac{f_oL}{EA}\left(\text{Exact= }\frac{f_oL}{EA}\right)$$

1.2.6 Subdomain Method

The idea behind the subdomain method is to force the integral of the residue to be equal to zero on a sub-interval of the domain. The method may be also seen as using the unit step functions as weighting functions. The unit step function may be described by:

$$S(x-x_0)=\int_{-\infty}^{\infty}\delta(x-x_0)\,dx \begin{matrix}=0(x<x_0)\\=1(x>x_0)\end{matrix}$$

Which leads to the identity:

$$S(x-x_j)-S(x-x_{j-1}) \begin{matrix}=1(x_j<x<x_{j+1})\\=0(x<x_j \text{ or } x>x_{j+1})\end{matrix}$$

Now, if we select a set of points x_j inside the domain of the problem, we may write down the integral of the residue, multiplied by the identity derived above, as follows:

$$\int_{Domain} R(x)(S(x-x_j)-S(x-x_{j+1}))\,dx=\int_{x_j}^{x_{j+1}} R(x)\,dx=\sum_{i=1}^{n} a_i \int_{x_j}^{x_{j+1}} (L(\psi_i(x))-g(x))\,dx=0$$

Which gives the equations in the form:

$$[k_{ij}]\{a_i\}=\{q_j\}$$

Where

$$k_{ij}=\int_{x_j}^{x_{j+1}} L(\psi_i(x))\,dx$$

$$q_j=\int_{x_j}^{x_{j+1}} g(x)\,dx$$

1.2.7 Example the bar problem using the subdomain method

The bar tensile problem is a classical problem that describes the relation between the axially distributed loads and the displacement of a bar. Let’s consider the bar in Figure 1.2.3 with constant modulus of elasticity and cross section area.

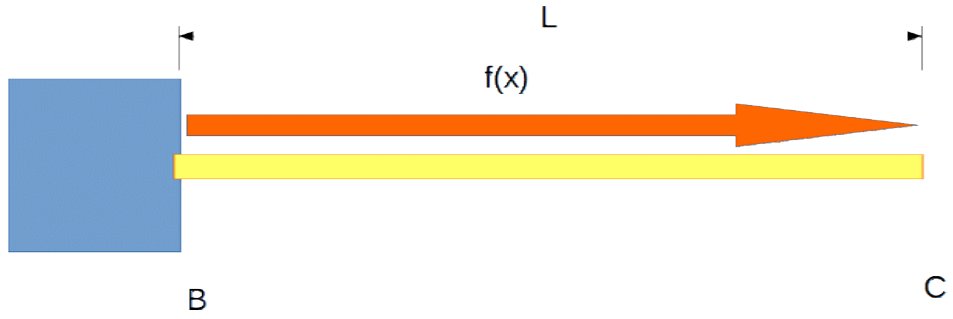

Figure 1.2.3: Simple fixed-free bar with distributed load

The force displacement relation is given by:

$$EA\frac{d^2u}{dx^2}+f(x)=0$$

Subject to the boundary conditions

$$u(0)=0$$
$$\frac{du(L)}{dx}=0$$

Now, lets use the approximate solution

$$u(x)=\sum_{i=1}^{n} a_i\psi_i(x)$$

Substituting it into the differential equation, we get

$$EA\sum_{i=1}^{n} a_i\frac{d^2\psi_i}{dx^2}+f(x)=R(x)$$

Applying the subdomain method, we get:

$$EA\sum_{i=1}^{n} a_i\int_{x_j}^{x_{j+1}}\left(\frac{d^2\psi_i}{dx^2}\right)dx=-\int_{x_j}^{x_{j+1}} f(x)\,dx$$

For the boundary conditions to be satisfied, we need a function that has zero value at x=0 and has a slope equal to zero at the free end. Sinusoidal functions are appropriate for this, hence, we may write:

$$u(x)=\sum_{i=1}^{n} a_i\sin\left(\frac{(2i-1)\pi x}{2L}\right)$$

For the purpose of illustration, let's use the first term only ($n=1$). The most logical choice for the subdomain would be $x_j=0$ & $x_{j+1}=L$, thus:

$$EA\,a\int_0^L\left(\frac{-\pi^2}{4L^2}\sin\left(\frac{\pi x}{2L}\right)\right)dx=-\int_0^L f(x)dx$$

For a constant forcing function, $f(x)=f_o$, performing the integration, we get:

$$\frac{EA\,\pi}{2L}a=f_o L$$

Or:

$$a=\frac{2f_o L^2}{EA\,\pi}\approx 0.637\frac{f_o L^2}{EA}$$

Then, the approximate solution for this problem becomes

$$u(x)\approx 0.637\frac{f_o L^2}{EA}\sin\left(\frac{\pi x}{2L}\right)$$

Now we may compare the obtained solution with the exact one that may be obtained from solving the differential equation. The maximum displacement and the maximum strain may be compared with the exact solution. The maximum displacement is

$$u(L)=0.637\frac{f_o L^2}{EA}\left(\text{Exact}=\ 0.5\frac{f_o L^2}{EA}\right)$$

And maximum strain is:

$$\frac{du(0)}{dx}=\frac{f_o L}{EA}\left(\text{Exact}=\ \frac{f_o L}{EA}\right)$$

1.2.8 The Galerkin Method

The Galerkin method uses the proposed solution functions as the weighting functions. Thus the solution procedure will require the selection of one set of functions. That method has proven very efficient and accurate as a weighted residual method. Many numerical solution methods are derived from the Galerkin method. The Galerkin method may be presented by the following integral:

$$\int_{Domain} R(x)\psi_j(x)dx = \sum_{i=1}^{n} a_i \int_{Domain} (L(\psi_i(x)) - g(x))\psi_j(x)dx = 0$$

Which gives the equations in the form:

$$[k_{ij}]\{a_i\} = \{q_j\}$$

Where

$$k_{ij} = \int_{Domain} L(\psi_i(x))\psi_j(x)dx$$

$$q_j = \int_{Domain} g(x)\psi_j(x)dx$$

1.2.9 Example the bar problem using the Galerkin method

The bar tensile problem is a classical problem that describes the relation between the axially distributed loads and the displacement of a bar. Let's consider the bar in Figure 1.2.4 with constant modulus of elasticity and cross section area.

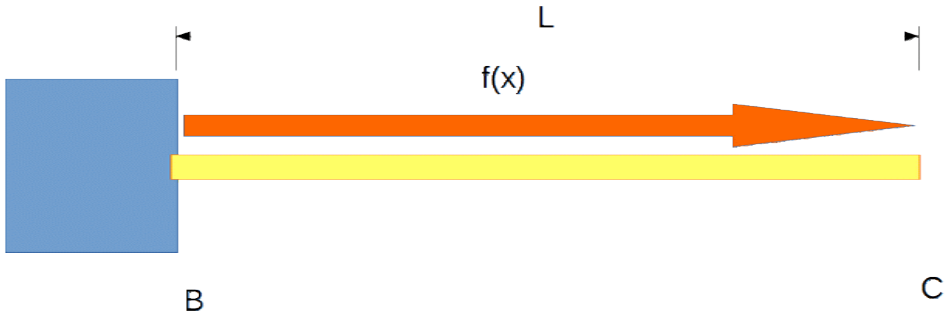

Figure 1.2.4: Simple fixed-free bar with distributed load

The force displacement relation is given by:

$$EA\frac{d^2u}{dx^2} + f(x) = 0$$

Subject to the boundary conditions

$$u(0)=0$$
$$\frac{du(L)}{dx}=0$$

Now, lets use the approximate solution

$$u(x)=\sum_{i=1}^{n} a_i \psi_i(x)$$

Substituting it into the differential equation, we get

$$EA\sum_{i=1}^{n} a_i \frac{d^2\psi_i}{dx^2}+f(x)=R(x)$$

Applying the Galerkin method, we get:

$$EA\sum_{i=1}^{n} a_i \int_0^L \left(\frac{d^2\psi_i}{dx^2}\right)\psi_j(x)\,dx=-\int_0^L f(x)\psi_j(x)\,dx$$

For the boundary conditions to be satisfied, we need a function that has zero value at x=0 and has a slope equal to zero at the free end. Sinusoidal functions are appropriate for this, hence, we may write:

$$u(x)=\sum_{i=1}^{n} a_i \sin\left(\frac{(2i-1)\pi x}{2L}\right)$$

For the purpose of illustration, let's use the first term only (*n=1*), thus:

$$EA\,a\int_0^L \left(\frac{-\pi^2}{4L^2}\sin^2\left(\frac{\pi x}{2L}\right)\right)dx=-\int_0^L f(x)\sin\left(\frac{\pi x}{2L}\right)dx$$

For a constant forcing function, $f(x)=f_o$, performing the integration, we get:

$$\frac{EA\,\pi^2}{4L^2}\frac{L}{2}a=f_o\frac{2L}{\pi}$$

Or:

$$a=\frac{16 f_o L^2}{EA\,\pi^3}\approx 0.52\frac{f_o L^2}{EA}$$

Then, the approximate solution for this problem becomes

$$u(x)\approx 0.52\frac{f_o L^2}{EA}\sin\left(\frac{\pi x}{2L}\right)$$

Now we may compare the obtained solution with the exact one that may be obtained from solving the differential equation. The maximum displacement and the maximum strain may be compared with the exact solution. The maximum displacement is

$$u(L)=0.52\frac{f_o L^2}{EA}\left(\text{Exact= } 0.5\frac{f_o L^2}{EA}\right)$$

And maximum strain is:

$$\frac{du(0)}{dx}=0.82\frac{f_o L}{EA}\left(\text{Exact= } \frac{f_o L}{EA}\right)$$

In most structure mechanics problems, the differential equation involves second derivative or higher for the displacement function. When Galerkin method is applied for such problems, you get the proposed function multiplied by itself or by one of its function family. This suggests the use of integration by parts. Let's examine this for the previous example. Substituting with the approximate solution: (Int. by Parts)

$$\int_0^L \frac{d^2\psi_i}{dx^2}\psi_j dx=\frac{d\psi_i}{dx}\psi_j\Big|_0^L-\int_0^L \frac{d\psi_i}{dx}\frac{d\psi_j}{dx}dx$$

But, for this specific problem, the boundary integrals are equal to zero since the functions were already chosen to satisfy the boundary conditions. Evaluating the integrals will give you the same results. So, what did we gain by performing the integration by parts?

- The functions are required to be less differentiable
- Not all boundary conditions need to be satisfied
- The matrix became symmetric!

The above gains suggested that the Galerkin method is the best candidate for the derivation of the finite element, finite volume, and boundary element models as a weighted residual method.

2. The Finite Element Method as a Weighted Residual Method

(Videos explaining this topic are available on http://AcademyOfKnowledge.org under "Introduction to the Finite Element Method: FEM-Basics")

One of the very popular numerical methods used for solving boundary value problems is the finite element method. A lot may be said and written about the method, but, here we are going to focus on the method as one that belongs to the weighted residual methods. To be more specific. The finite element method may be viewed as one of the Galerkin methods applied on a subdomain. The basic idea in this approach is that the differential equation may apply to any part, subdomain, of the whole domain, and we may apply the Galerkin method to that subdomain, called element. Further, the finite element method will use a function series that have generalized coordinates that have direct physical meaning, much like what we learner in the Lagrange interpolation method.

2.1 The Grid – Elements and Nodes

The first step in the finite element method is dividing the domain into elements (subdomains) that are connected to neighboring elements at nodes. The grid used by the finite element method is called *unstructured grid* because does not need to have any specific order of numbering for the elements or the nodes (Figure 2.1.1 presents an example of an unstructured grid creaded for aerodynamic analysis of an airfoil while Figure 2.1.2 presents a structured one). However, if we can impose some order to the numbering, that will result in *better looking* set of equations that may have some special methods of solving in a faster manner. In this work, we will not focus on ordering the elements and nodes in a certain manner, though, the problem we will introduce will be all having straight-forward numbering schemes.

For a one-dimensional domain. The grid generation is much simpler than what we see in the figures above. The domain is divided into intervals, elements, and at the end of each element the points are named *nodes*. The element length, thus, is the difference between the x-values of the end nodes. (see Figure 2.1.3). Now that we have all what we need to know about the grid, we may start creating the interpolation functions.

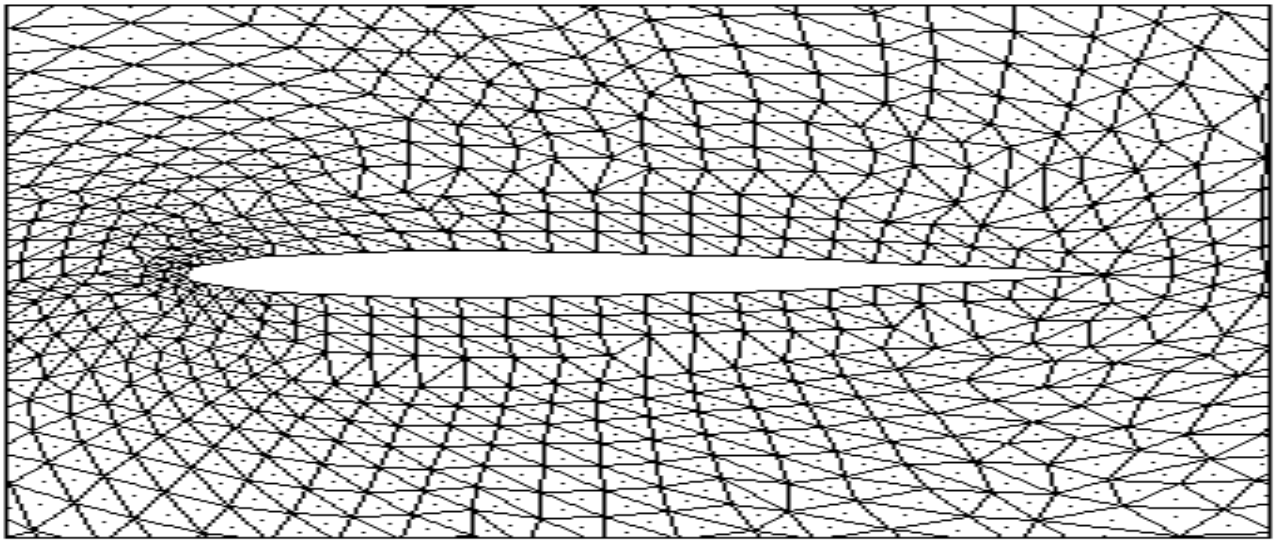

Figure 2.1.1: Example of 2-D unstructured grid near the surface of an airfoil

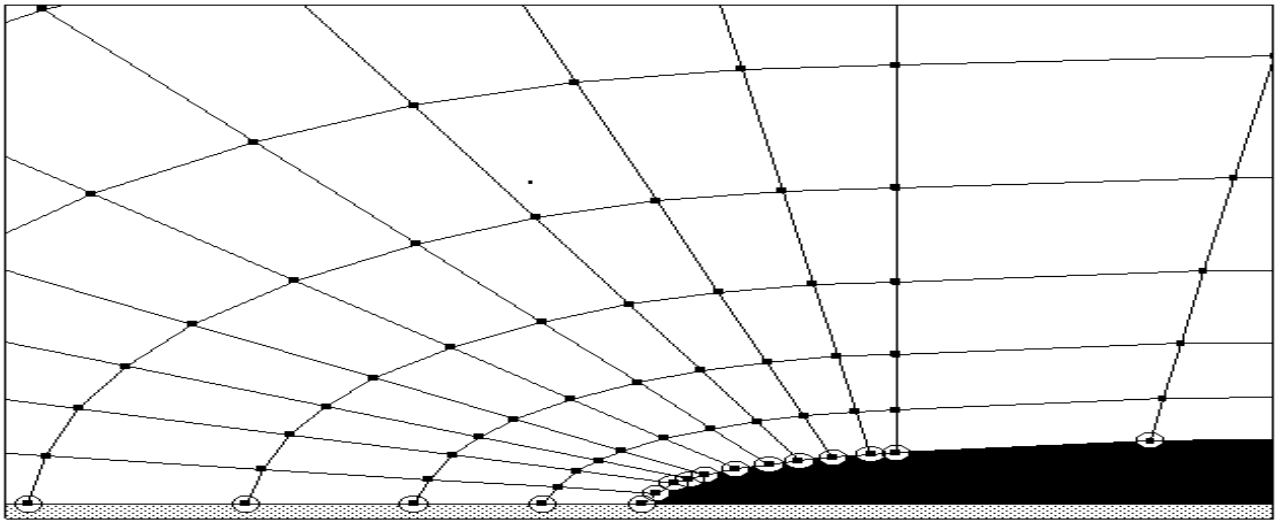

Figure 2.1.2: An example of a structured grid near the leading edge of an airfoil

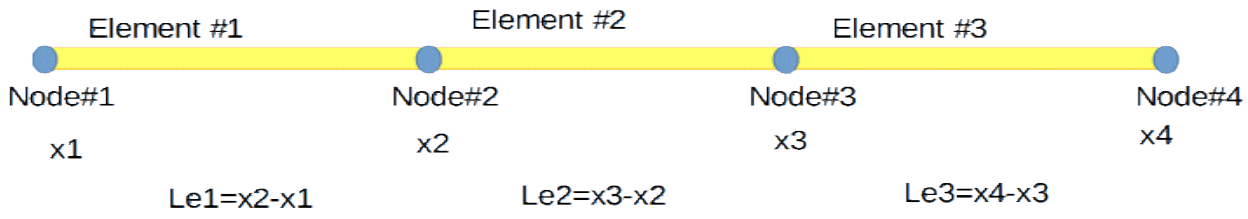

Figure 2.1.3: A grid in one-dimensional domain

2.2 Interpolation function

The interpolation function for the element needs to satisfy the essential boundary conditions as we intend to be using the Galerkin method. If the essential boundary conditions are the values of the function at each end of the element, we need to satisfy two conditions, which leads to the need of a first order polynomial. The polynomial may, generally, be written as:

$$u(x)=a_0+a_1 x$$

Where u(x) may present a displacement of a bar, the temperature in a conductive material, the flow potential in a fluid, or any other function that may be described by the problem. In vector form, we may write:

$$u(x)=\lfloor 1 \quad x \rfloor \begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix} = \lfloor H(x) \rfloor \{a\}$$

The boundary conditions that need to be satisfied by this polynomial are the values of the function at each of the end of the elements presented in Figure 2.2.1.

Figure 2.2.1. One-Dimensional element with end function values and concentrated excitations.

Using the boundary conditions, we may get:

$$u(0)=u_1=\lfloor H(0) \rfloor \{a\}$$
$$u(L)=u_2=\lfloor H(L) \rfloor \{a\}$$

In matrix form:

$$\begin{Bmatrix} u_1 \\ u_2 \end{Bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & L \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \end{Bmatrix}$$

$$\{u\}=[T_m]\{a\}$$

Which may, readily, be solved to get:

$$\{a\}=\begin{bmatrix} 1 & 0 \\ \frac{-1}{L} & \frac{1}{L} \end{bmatrix}\{u\}$$

$$\{a\}=[T_m]^{-1}\{u\}$$

From which we may write:

$$u(x)=\lfloor H(x)\rfloor[T_m]^{-1}\{u\}=\lfloor N(x)\rfloor\{u\}$$

where:

$$\lfloor N(x)\rfloor^T=\{N(x)\}=\begin{Bmatrix} 1-\frac{x}{L} \\ \frac{x}{L} \end{Bmatrix}$$

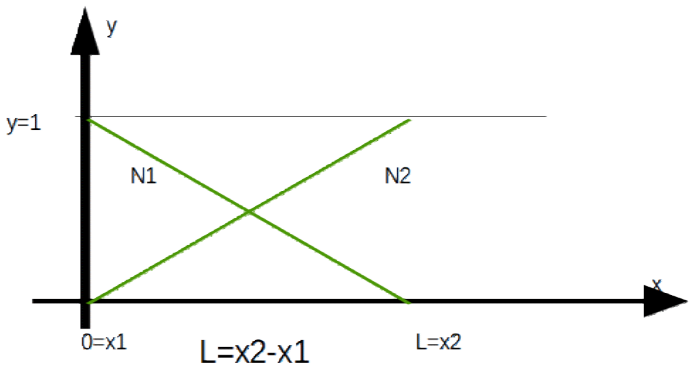

Figure 2.2.2. A sketch of $N_1(x)$ and $N_2(x)$.

Note that the functions presented in the above vector are exactly those we would have got using the Lagrange interpolation method. If we plot the two functions, we will get a graph similar to the sketch presented in Figure 2.2.2. Let's now keep focusing on this simple problem to illustrate the method. From the point of view of the weighted residual methods, we may view the interpolation function as:

$$u(x)=N_1(x)u_1+N_2(x)u_2=\sum_{i=1}^{2}u_i N_i(x)$$

Where the proposed solution functions are $N_i(x)$ while the generalized coordinates are u_i. The next step would be creating the element equations using the Galerkin method.

2.3 Element Equations

Recall from section 1.2.8 that the Galerkin method may be written as:

$$\int_{Domain} R(x)N_j(x)dx=\sum_{i=1}^{n} u_i \int_{Domain} (L(N_i(x))-g(x))N_j(x)dx=0$$

Which gives the equations in the form:

$$[k_{ij}]\{u_i\}=\{q_j\}$$

Where

$$k_{ij}=\int_{Domain} L(N_i(x))N_j(x)dx$$
$$q_j=\int_{Domain} g(x)N_j(x)dx$$

If we use the bar problem as an example, we may rewrite the governing equation of the bar in the form:

$$\frac{d}{dx}\left(EA\frac{du}{dx}\right)+f(x)=0$$

Where E is the modulus of elasticity, A is the cross-sectional area, u is the longitudinal displacement of each point of the bar, and $f(x)$ is a longitudinal distributed loading that may occur due to gravitational field, other kind of fields, or friction applies on the surface of the bar. (see Figure 2.3.1)

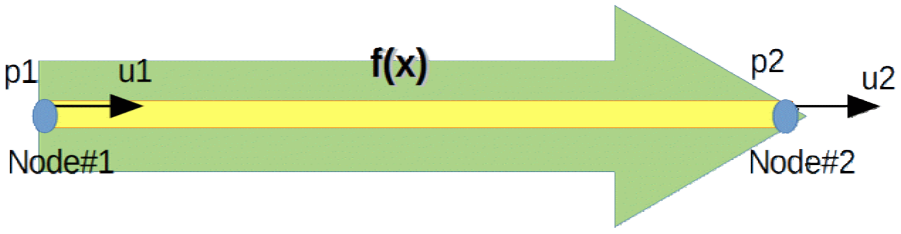

Figure 2.3.1. External distributed load on a bar element.

If we use the proposed solution into the differential equation, we get:

$$\sum_{i=1}^{2} \frac{d}{dx}\left(EA\frac{dN_i}{dx}u_i\right)+f(x)=R(x)$$

Applying the Galerkin method, we get:

$$\int_{x_1}^{x_2} N_j R(x)\,dx=\sum_{i=1}^{2}\int_{x_1}^{x_2} N_j \frac{d}{dx}\left(EA\frac{dN_i}{dx}u_i\right)dx+\int_{x_1}^{x_2} N_j f(x)\,dx=0$$

Where x_1 and x_2 are the boundaries of the element we are concerned with. However, since all the integrations we are performing are bound by x_1 and x_2, we may transfer the coordinates such that we perform all the integrals from 0 to L where L is the element length. Applying the integration by parts, the above equation may be rewritten as:

$$\sum_{i=1}^{2}\left(N_j EA\frac{dN_i}{dx}u_i\Big|_0^L-u_i\int_0^L EA\frac{dN_j}{dx}\frac{dN_i}{dx}dx\right)+\int_0^L N_j f(x)\,dx=0$$

Separating the boundary evaluations, we get:

$$\sum_{i=1}^{2}\left(u_i\int_0^L EA\frac{dN_j}{dx}\frac{dN_i}{dx}dx\right)=\int_0^L N_j f(x)\,dx+N_j\sum_{i=1}^{2} EA\frac{dN_i}{dx}u_i\Big|_0^L$$

From the interpolation functions we obtained in section 2.2, can find that:

$$N_1(0)=1\ ,\ N_1(L)=0\ ,\ N_2(0)=0\ ,\ N_2(L)=1$$
$$\frac{dN_1}{dx}=-\frac{1}{L}\ ,\ \frac{dN_2}{dx}=\frac{1}{L}$$

Using those relations into the equation above, we get:

$$\left(-u_1\int_0^L EA\frac{dN_j}{dx}\frac{1}{L}dx\right)+\left(u_2\int_0^L EA\frac{dN_j}{dx}\frac{1}{L}dx\right)=\int_0^L N_j f(x)\,dx+N_j\left(EA\frac{du}{dx}\right)\Big|_0^L$$

Note that du/dx is the strain of the bar element, when the strain is multiplied by the modulus of elasticity and the area, the result becomes the applied load. For j=1, we get the first equation as:

$$\left(u_1\int_0^L EA\frac{1}{L^2}dx\right)-\left(u_2\int_0^L EA\frac{1}{L^2}dx\right)=\int_0^L (1-\frac{x}{L})f(x)dx+p_1$$

For j=2, we get the second equation as:

$$\left(-u_1\int_0^L EA\frac{1}{L^2}dx\right)+\left(u_2\int_0^L EA\frac{1}{L^2}dx\right)=\int_0^L (\frac{x}{L})f(x)dx+p_2$$

If we assume EA to be constant over the element, we get the element equations as:

$$\frac{EA}{L}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}\begin{Bmatrix}u_1\\ u_2\end{Bmatrix}=\begin{Bmatrix}f_1\\ f_2\end{Bmatrix}+\begin{Bmatrix}p_1\\ p_2\end{Bmatrix}$$

Or

$$[K]\{u\}=\{q\}$$

Where [K] is called the stiffness matrix and {q} is called the generalized force vector. Note that the generalized force vector includes the effect of the distributed loads as well as the concentrated ones. Also note that we can not write an explicit expression for the generalized force without knowing the force function. However, we may write:

$$\int_0^L f(x)\{N\} = \begin{Bmatrix} \int_0^L f(x) N_1 dx \\ \int_0^L f(x) N_2 dx \end{Bmatrix} = \begin{Bmatrix} f_1 \\ f_2 \end{Bmatrix}$$

Let's examine what have we got here. On the left hand side, we have a matrix that is multiplied by a vector of the displacements at both ends of the bar element. On the right hand side we have two vectors. The first vector is the result of integrating the distributed force over the element, while the second vector represents the externally applied loads at the nodes of the elements. The above equation is *the element equation*.

2.4 Assembling the Structure Equations

Let's consider a simple bar structure that is divided into two elements as in Figure 2.4.1. We should consider two things:

1. The displacement at the second node of the first element is exactly the same as that at the first node of the second element. This should apply to ensure the continuity of the displacement from one element to the neighboring one.
2. The concentrated loads maybe divided among the elements at the connecting node, but, the summation of the divided parts should add up to the total externally applied concentrated load.

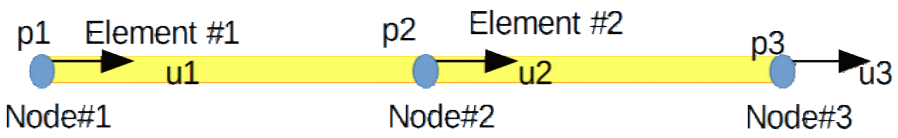

Figure 2.4.1. Two-Element bar structures.

We may write the element equation for each element to be:

$$[K^{(1)}]\{u\}^{(1)}=\{q\}^{(1)}$$

$$[K^{(2)}]\{u\}^{(2)}=\{q\}^{(2)}$$

Each of the above equations consists of two scalar equations. Now, if we want to model the whole structure, we will need to add the generalized forces to obtain the total external concentrated loads.

$$\frac{EA}{L}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}\begin{Bmatrix}u_1\\ u_2\end{Bmatrix}=\begin{Bmatrix}f_1^{(1)}\\ f_2^{(1)}\end{Bmatrix}+\begin{Bmatrix}p_1^{(1)}\\ p_2^{(1)}\end{Bmatrix}$$

$$\frac{EA}{L}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}\begin{Bmatrix}u_2\\ u_3\end{Bmatrix}=\begin{Bmatrix}f_2^{(2)}\\ f_3^{(2)}\end{Bmatrix}+\begin{Bmatrix}p_2^{(2)}\\ p_3^{(2)}\end{Bmatrix}$$

Where the superscripts (1) and (2) denote the information from the first and second element matrices respectively. If we expand the above to sets of equations, we get four equations:

$$\begin{aligned}
\frac{EA}{L}u_1-\frac{EA}{L}u_2&=f_1^{(1)}+p_1^{(1)}\\
-\frac{EA}{L}u_1+\frac{EA}{L}u_2&=f_2^{(1)}+p_2^{(1)}\\
\frac{EA}{L}u_2-\frac{EA}{L}u_3&=f_2^{(2)}+p_2^{(2)}\\
-\frac{EA}{L}u_2+\frac{EA}{L}u_3&=f_3^{(2)}+p_3^{(2)}
\end{aligned}$$

If we add the second and the third equations, we get:

$$\begin{aligned}
\frac{EA}{L}u_1-\frac{EA}{L}u_2&=f_1^{(1)}+p_1^{(1)}\\
-\frac{EA}{L}u_1+2\frac{EA}{L}u_2-\frac{EA}{L}u_3&=f_2^{(1)}+f_2^{(2)}+p_2^{(2)}+p_2^{(1)}\\
-\frac{EA}{L}u_2+\frac{EA}{L}u_3&=f_3^{(2)}+p_3^{(2)}
\end{aligned}$$

Keeping in mind that the displacements at the connecting node is the same, we can get the equations for the assembled structure to be:

$$\frac{EA}{L}\begin{bmatrix}1 & -1 & 0\\ -1 & 2 & -1\\ 0 & -1 & 1\end{bmatrix}\begin{Bmatrix}u_1\\ u_2\\ u_3\end{Bmatrix}=\begin{Bmatrix}f_1^{(1)}\\ f_2^{(1)}+f_1^{(2)}\\ f_2^{(2)}\end{Bmatrix}+\begin{Bmatrix}p_1\\ p_2\\ p_3\end{Bmatrix}$$

Where we assumed that the element lengths, modulus of elasticity, and cross-section area are constant for both elements.

Thus, the assembly could be reduced to a simple procedure:

1. Create an empty square matrix (global matrix) with the size of the number of nodes (degrees of freedom)
2. *Add* each element matrix to the corresponding location in the global matrix. For the case of the bar, the element matrices will lie on the diagonal of the global matrix.

2.5 Applying the boundary conditions

Up to this moment, all the procedure described for the finite element model may apply to any bar with any number of elements. Differences will appear when different bars are subjected to different boundary conditions. For illustration purposes, we will consider two cases here.

2.5.1 Case of a fixed-free bar:

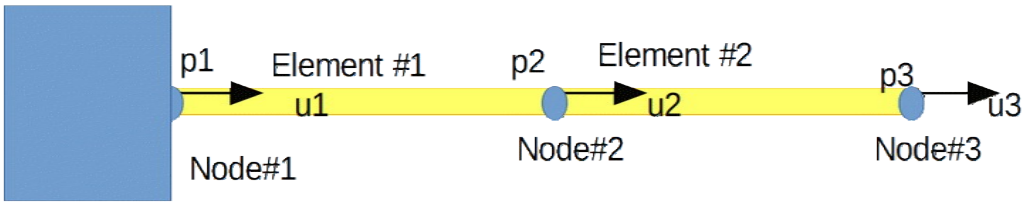

Figure 2.5.1. Fixed-Free Bar.

For a fixed-free bar fixed from the left side, like the one demonstrated in Figure 2.5.1, we have:

$$u_1 = 0$$

Thus, the equation we obtained may be written as:

$$\frac{EA}{L}\begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix}\begin{Bmatrix} 0 \\ u_2 \\ u_3 \end{Bmatrix} = \begin{Bmatrix} f_1^{(1)} \\ f_2^{(1)} + f_1^{(2)} \\ f_2^{(2)} \end{Bmatrix} + \begin{Bmatrix} p_1 \\ p_2 \\ p_3 \end{Bmatrix}$$

Which may be separated into two distinct equations given as:

$$\frac{EA}{L}\begin{bmatrix}2 & -1\\ -1 & 1\end{bmatrix}\begin{Bmatrix}u_2\\ u_3\end{Bmatrix}=\begin{Bmatrix}f_2^{(1)}+f_1^{(2)}\\ f_2^{(2)}\end{Bmatrix}+\begin{Bmatrix}p_2\\ p_3\end{Bmatrix}$$

Which is called the *primary equations*. The primary equations could be readily solved for the unknown displacements by inverting the matrix on the left hand side and multiplying it by the right hand side vectors. The second equation becomes:

$$\frac{EA}{L}\begin{bmatrix}-1 & 0\end{bmatrix}\begin{Bmatrix}u_2\\ u_3\end{Bmatrix}=\{f_1^{(1)}\}+\{p_1\}$$

Which is called the *secondary equation* or the *auxiliary equation*. The secondary equation has the unknown variables as the concentrated force at the fixed end, thus, it is the support reaction which is found by substituting the solution obtained from the primary equations directly.

2.5.2 Case of a fixed-fixed bar:

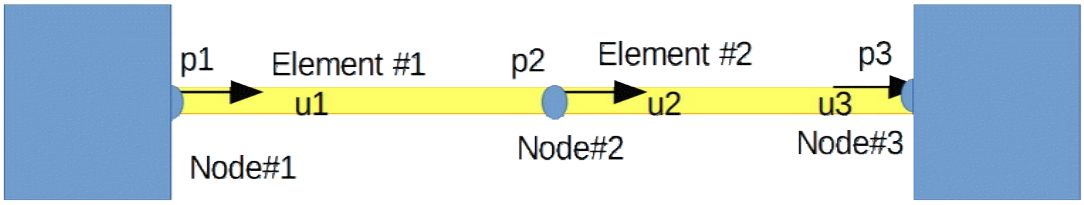

Figure 2.5.2. Fixed-Fixed Bar.

For a fixed-fixed bar, like the one demonstrated in Figure 2.5.2, we have:

$u_1=0$ and $u_3=0$

Thus, the equation we obtained may be written as:

$$\frac{EA}{L}\begin{bmatrix}1 & -1 & 0\\ -1 & 2 & -1\\ 0 & -1 & 1\end{bmatrix}\begin{Bmatrix}0\\ u_2\\ 0\end{Bmatrix}=\begin{Bmatrix}f_1^{(1)}\\ f_2^{(1)}+f_1^{(2)}\\ f_2^{(2)}\end{Bmatrix}+\begin{Bmatrix}p_1\\ p_2\\ p_3\end{Bmatrix}$$

which gives the primary equation as:

$$\frac{EA}{L}[2]\{u_2\}=\{f_2^{(1)}+f_1^{(2)}\}+\{p_2\}$$

And the secondary equations as:

$$\frac{EA}{L}\begin{bmatrix}-1\\-1\end{bmatrix}\{u_2\}=\begin{Bmatrix}f_1^{(1)}\\f_2^{(2)}\end{Bmatrix}+\begin{Bmatrix}p_1\\p_3\end{Bmatrix}$$

Which may be used to evaluate the support reactions at both ends.

2.6 Summary of the Finite Element Procedure

In this section, the general procedure of the finite element method was demonstrated using the differential equation governing the deflection of a bar. The procedure may be summarized as:

1. Create the grid. Identify all elements in the domain and the nodes that connect different elements.
2. Select the interpolation function that you will be using for the elements
3. Define the element matrices through writing the element equation
4. Assemble the element matrices to create the whole domain equations
5. Apply boundary conditions and separate the secondary equations
6. Solve for the primary variables that represent the solution of the differential equation at the nodes
7. If needed, evaluate the secondary variables using the secondary equations.

We need to notice that it is common, when investigating new problems using the finite element method, to perform *convergence analysis*. To test weather the solution has reached the best we may obtain using the model, we change the mesh size several times by increasing the number of elements and see weather the solution is changed much. When changing the number of elements does not affect the final solution, we claim that the solution has converged and that we have is the best possible solution.

In the following sections we will use the finite element method, as derived through the Galerkin method, to solve different problems and demonstrate how they may be programmed using the open source package Octave© which is so similar to Matlab©. We will also demonstrate how to obtain the element matrices using the package Mathematica© and the open source package wxMaxima® which facilitate symbolic analysis.

2.7 Examples

2.7.1 Compound bar with multiple loads

A compound bar is one that is composed of more than one bar bonded to one another at their ends. For the bar below, calculate the reaction in the support B, the stress, strain, and elongation of parts 1 & 2, the deflection of points D & C.

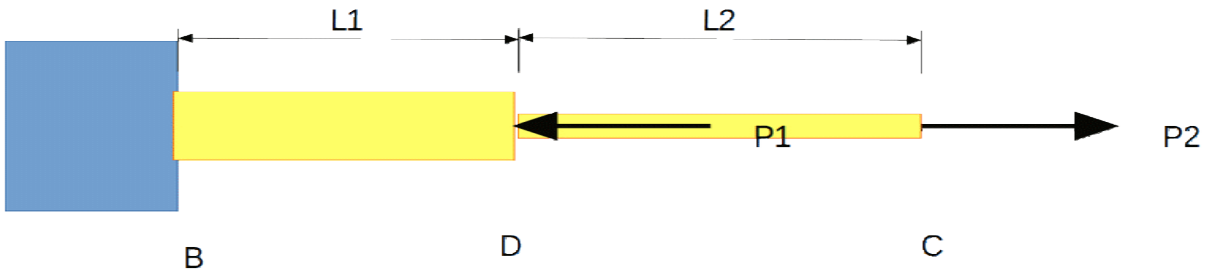

Figure 2.7.1: Compound bar with multiple loads

Solution:

Dividing the structure into two elements BD and DC, we get the elements' equations as:

$$\frac{E_1 A_1}{L_1}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\begin{Bmatrix} u_B \\ u_D \end{Bmatrix}=\begin{Bmatrix} R \\ -P_1^1 \end{Bmatrix}$$

$$\frac{E_2 A_2}{L_2}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\begin{Bmatrix} u_D \\ u_C \end{Bmatrix}=\begin{Bmatrix} -P_1^2 \\ P_2 \end{Bmatrix}$$

Assembling the structure matrix, we get:

$$\begin{bmatrix} \frac{E_1 A_1}{L_1} & \frac{-E_1 A_1}{L_1} & 0 \\ \frac{-E_1 A_1}{L_1} & \frac{E_1 A_1}{L_1}+\frac{E_2 A_2}{L_2} & \frac{-E_2 A_2}{L_2} \\ 0 & \frac{-E_2 A_2}{L_2} & \frac{E_2 A_2}{L_2} \end{bmatrix}\begin{Bmatrix} u_B \\ u_D \\ u_C \end{Bmatrix}=\begin{Bmatrix} R \\ -P_1 \\ P_2 \end{Bmatrix}$$

The boundary conditions of this problem are that the displacement u_B is equal to zero, which gives the main equations and the auxiliary equation as:

$$\begin{bmatrix} \frac{E_1 A_1}{L_1}+\frac{E_2 A_2}{L_2} & \frac{-E_2 A_2}{L_2} \\ \frac{-E_2 A_2}{L_2} & \frac{E_2 A_2}{L_2} \end{bmatrix}\begin{Bmatrix} u_D \\ u_C \end{Bmatrix}=\begin{Bmatrix} -P_1 \\ P_2 \end{Bmatrix}$$

$$\begin{bmatrix} \frac{E_1 A_1}{L_1} & -\frac{E_1 A_1}{L_1} & 0 \end{bmatrix} \begin{Bmatrix} 0 \\ u_D \\ u_C \end{Bmatrix} = \{R\}$$

Solving the main equations for the displacements, we get:

$$u_D = \frac{(P_2 - P_1) L_1}{E_1 A_1}$$

$$u_C = \frac{(P_2 - P_1) L_1}{E_1 A_1} + \frac{P_2 L_2}{E_2 A_2}$$

Which when substituted in the auxiliary equation, gives the reaction force as:

$$R = P_1 - P_2$$

Note: you may check the results using the classical mechanics of material solution procedure and you will find that the Finite Element model we used was able to return the exact solution.

2.7.2 Bar under its own weight

The bar shown below is hanging vertically. Find the displacement and stress distributions in the bar.

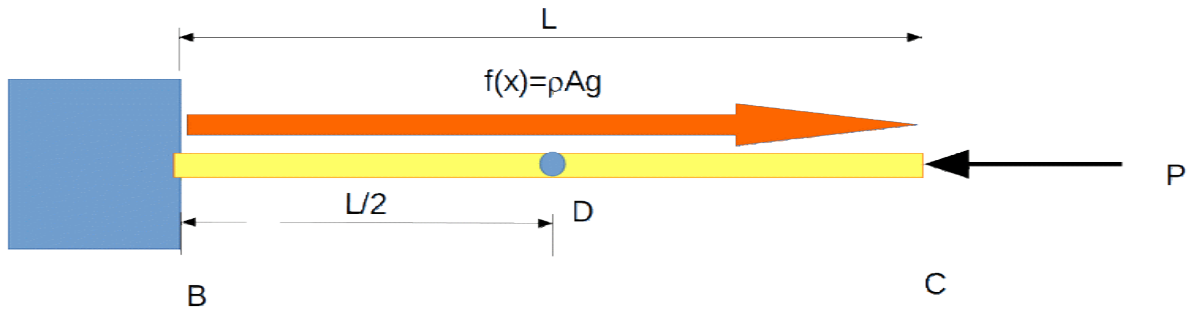

Figure 2.7.2: Fixed-free bar subject to end and distributed loads

Solution:

Dividing the structure into two elements, we may write the elements' equations as:

$$\frac{E A}{L/2} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{Bmatrix} u_B \\ u_D \end{Bmatrix} = \rho A g \frac{L}{2} \begin{Bmatrix} \frac{1}{2} \\ \frac{1}{2} \end{Bmatrix} + \begin{Bmatrix} R \\ 0 \end{Bmatrix}$$

$$\frac{EA}{L/2}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}\begin{Bmatrix}u_D\\ u_C\end{Bmatrix}=\rho A g\frac{L}{2}\begin{Bmatrix}\frac{1}{2}\\ \frac{1}{2}\end{Bmatrix}+\begin{Bmatrix}0\\ -P\end{Bmatrix}$$

Assembling the structure matrix, we get:

$$\frac{EA}{L/2}\begin{bmatrix}1 & -1 & 0\\ -1 & 2 & -1\\ 0 & -1 & 1\end{bmatrix}\begin{Bmatrix}u_B\\ u_D\\ u_C\end{Bmatrix}=\frac{\rho A g L}{4}\begin{Bmatrix}1\\ 2\\ 1\end{Bmatrix}+\begin{Bmatrix}R\\ 0\\ -P\end{Bmatrix}$$

The boundary conditions of this problem are that the displacement u_B is equal to zero, which gives the main equations and the auxiliary equation as:

$$\frac{EA}{L/2}\begin{bmatrix}2 & -1\\ -1 & 1\end{bmatrix}\begin{Bmatrix}u_D\\ u_C\end{Bmatrix}=\frac{\rho A g L}{4}\begin{Bmatrix}2\\ 1\end{Bmatrix}+\begin{Bmatrix}0\\ -P\end{Bmatrix}$$

$$\frac{EA}{L/2}\begin{bmatrix}1 & -1 & 0\end{bmatrix}\begin{Bmatrix}0\\ u_D\\ u_C\end{Bmatrix}=\frac{\rho A g L}{4}+R$$

Solving the main equations for the displacements, we get:

$$\begin{Bmatrix}u_D\\ u_C\end{Bmatrix}=\frac{\rho g L^2}{8E}\begin{Bmatrix}3\\ 4\end{Bmatrix}-\frac{PL}{2EA}\begin{Bmatrix}1\\ 2\end{Bmatrix}$$

Or:

$$u_D=\frac{3\rho g L^2}{8E}-\frac{PL}{2EA}$$

$$u_C=\frac{\rho g L^2}{2E}-\frac{PL}{EA}$$

Which is equal to the exact solution. When substituting the solution into the auxiliary equation, gives the reaction force as:

$R=P-\rho AgL$ (which is the exact solution as well)

For the sake of illustration, let's repeat the above problem using **four elements** instead of two. Also, let's use node numbering instead of letters (Figure 2.7.3). For this case, the Element length will be one quarter of the total length, which will only affect the element equation:

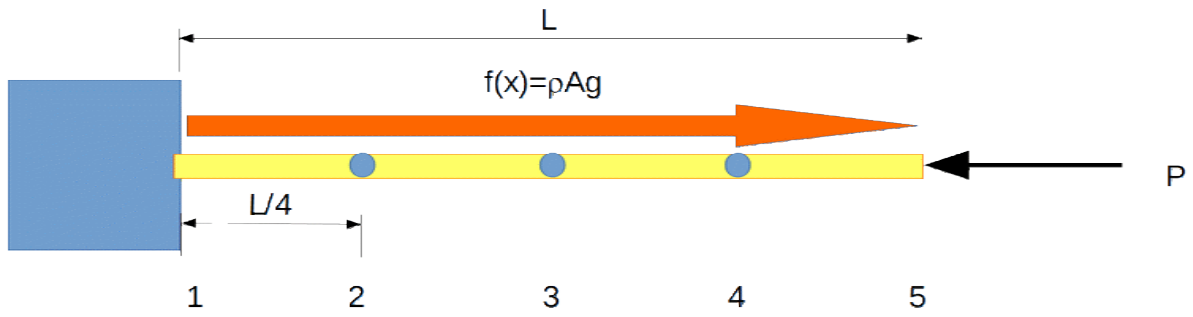

Figure 2.7.3: Four-Element Example

$$\frac{EA}{L/4}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}\begin{Bmatrix}u_i\\ u_{i+1}\end{Bmatrix}=\rho A g\frac{L}{4}\begin{Bmatrix}\frac{1}{2}\\ \frac{1}{2}\end{Bmatrix}+\begin{Bmatrix}p_1^{(i)}\\ p_2^{(i)}\end{Bmatrix}$$

Assembling the four elements, we get:

$$\frac{EA}{L/4}\begin{bmatrix}1 & -1 & 0 & 0 & 0\\ -1 & 2 & -1 & 0 & 0\\ 0 & -1 & 2 & -1 & 0\\ 0 & 0 & -1 & 2 & -1\\ 0 & 0 & 0 & -1 & 1\end{bmatrix}\begin{Bmatrix}u_1\\ u_2\\ u_3\\ u_4\\ u_5\end{Bmatrix}=\frac{\rho A g L}{8}\begin{Bmatrix}1\\ 2\\ 2\\ 2\\ 1\end{Bmatrix}+\begin{Bmatrix}R\\ 0\\ 0\\ 0\\ -P\end{Bmatrix}$$

Applying the boundary conditions, we get the primary equations:

$$\frac{EA}{L/4}\begin{bmatrix}2 & -1 & 0 & 0\\ -1 & 2 & -1 & 0\\ 0 & -1 & 2 & -1\\ 0 & 0 & -1 & 1\end{bmatrix}\begin{Bmatrix}u_2\\ u_3\\ u_4\\ u_5\end{Bmatrix}=\frac{\rho A g L}{8}\begin{Bmatrix}2\\ 2\\ 2\\ 1\end{Bmatrix}+\begin{Bmatrix}0\\ 0\\ 0\\ -P\end{Bmatrix}$$

And the secondary:

$$\frac{EA}{L/4}\begin{bmatrix}1 & -1 & 0 & 0 & 0\end{bmatrix}\begin{Bmatrix}u_1\\ u_2\\ u_3\\ u_4\\ u_5\end{Bmatrix}=\frac{\rho A g L}{8}+R$$

Solving the primary equations, you get:

$$\begin{Bmatrix} u_2 \\ u_3 \\ u_4 \\ u_5 \end{Bmatrix} = \frac{\rho g L^2}{32 E} \begin{Bmatrix} 7 \\ 12 \\ 15 \\ 16 \end{Bmatrix} - \frac{PL}{4 EA} \begin{Bmatrix} 1 \\ 2 \\ 3 \\ 4 \end{Bmatrix}$$

And you may check the substitution into the auxiliary equation yourself

2.7.3 Compound bar fixed from both sides

For the bar shown below, determine the reaction forces in supports B & C and the displacement of point D.

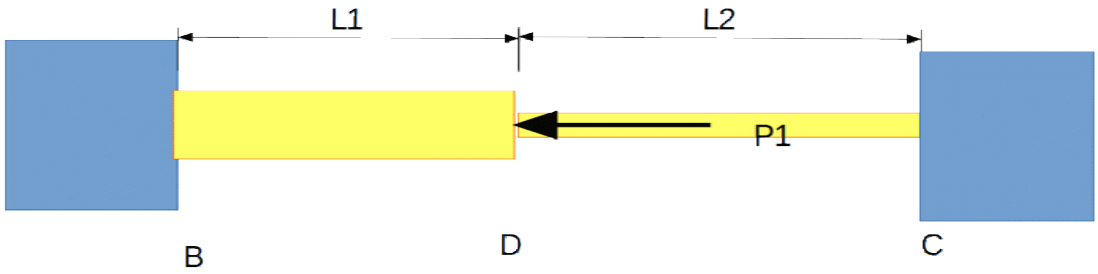

Figure 2.7.4: Compund bar fixed from both sides

Solution:

Dividing the structure into two elements BD and DC, we get the elements' equations as:

$$\frac{E_1 A_1}{L_1} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{Bmatrix} u_B \\ u_D \end{Bmatrix} = \begin{Bmatrix} R_B \\ -P_1^1 \end{Bmatrix}$$

$$\frac{E_2 A_2}{L_2} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{Bmatrix} u_D \\ u_C \end{Bmatrix} = \begin{Bmatrix} -P_1^2 \\ R_C \end{Bmatrix}$$

Assembling the structure matrix, we get:

$$\begin{bmatrix} \frac{E_1 A_1}{L_1} & \frac{-E_1 A_1}{L_1} & 0 \\ \frac{-E_1 A_1}{L_1} & \frac{E_1 A_1}{L_1} + \frac{E_2 A_2}{L_2} & \frac{-E_2 A_2}{L_2} \\ 0 & \frac{-E_2 A_2}{L_2} & \frac{E_2 A_2}{L_2} \end{bmatrix} \begin{Bmatrix} u_B \\ u_D \\ u_C \end{Bmatrix} = \begin{Bmatrix} R_B \\ -P_1 \\ R_C \end{Bmatrix}$$

The boundary conditions of this problem are that the displacements u_B and u_C are equal to zero, which gives the main equation and the auxiliary equations as:

$$\left(\frac{E_1 A_1}{L_1}+\frac{E_2 A_2}{L_2}\right)u_D=-P_1$$

$$\begin{Bmatrix}\frac{-E_1 A_1}{L_1}\\ \frac{-E_2 A_2}{L_2}\end{Bmatrix}u_D=\begin{Bmatrix}R_B\\ R_C\end{Bmatrix}$$

Solving the main equation for the displacement, we get:

$$u_D=\frac{-P_1}{\frac{E_1 A_1}{L_1}+\frac{E_2 A_2}{L_2}}$$

Which when substituted in the auxiliary equations, gives the reaction forces as:

$$\frac{P_1}{\frac{E_1 A_1}{L_1}+\frac{E_2 A_2}{L_2}}\begin{Bmatrix}\frac{E_1 A_1}{L_1}\\ \frac{E_2 A_2}{L_2}\end{Bmatrix}=\begin{Bmatrix}R_B\\ R_C\end{Bmatrix}$$

2.7.4 Bar with non-constant cross-section

Let us consider the case when the cross section of the bar is not constant. In this case, we should expect the element equation to look a bit different from the one with constant cross sections. The are several proposed ways to solve this problem. The easiest, but computationally expensive, is to divide the element into several smaller elements each with a constant cross section, then use the original element equation for each. This solution should yield the exact solution as the number of elements becomes very large, nevertheless, this does not seem practical.

Another way of solving this problem is to use an average of the cross section properties into the original element equation, which is quite practical and should not be hard to compute, however, you need to be aware of the fact that you have introduced an approximation into the model, which may be good in terms of the expected error.

Finally, I recommend getting back to the element equation and perform the integrations using the distribution function, if known, to obtain the exact integrals in order to be able to capture the full physics into the model. For illustration, let us consider the case in Figure 2.7.5 where the area is changing along the bar with a given function.

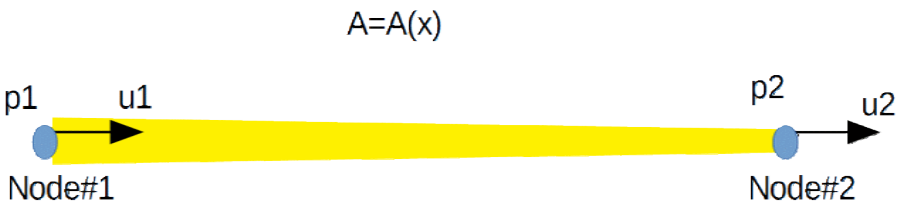

Figure 2.7.5. Bar with non-constant cross-section.

We may write the element equations using Galerkin method:

$$\int_{x_1}^{x_2} EA(x)\{N'(x)\}\{N'(x)\}^T dx = \int_{x_1}^{x_2} f(x)\{N(x)\}dx + \{N(x)\}\big|_{x=x_1} p_1 + \{N(x)\}\big|_{x=x_2} p_2$$

For the above equation, the right-hand-side will not change from what we obtained before yielding the excitation vectors due to distributed and concentrated loads simultaneously. To obtain the left hand side matrix, we will need to include the change in the area in the integration. We may use a symbolic manipulator to perform the integration. Below, you may find the wxMaxima® code for performing that.

```
H:matrix([1,x]);
Hx:diff(H,x);
A:A1*(L-x)/L + A2*(x/L);
x:0;
Tb1:H,numer;
x:L;
Tb2:H,numer;
TT:matrix(Tb1[1],Tb2[1]);
T1:invert(TT);
kill(x);

Ke:E.transpose(T1).integrate(transpose(Hx).Hx*A,x,0,L).T1;
```

In the above code, we used a linear function for the distribution of the area to simplify the problem. However, the resulting stiffness matrix was exactly the same as the original one with the area replaced by the average-area:

$$K = \frac{E A_{average}}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

$$A_{average} = \frac{A_1 + A_2}{2}$$

Which is a special case for the linear distribution. Now, you have the tool necessary to generate any stiffness matrix with any distribution of area and/or modulus of elasticity.

3. Bars and Trusses

(Videos explaining this topic are available on http://AcademyOfKnowledge.org under "Introduction to the Finite Element Method: Truss Elements")

3.1 The Element Equation using Vectors

In the previous sections, we presented the derivation of the element equation through the weighted residual method. However, we presented the derivation through series terms. In this section, we will be presenting the derivation of the equation using vectors (still using weighted residual approach) in order to be able to generalize the procedure later in our work.

Recall that obtained the integration of the residue using Galerkin method as:

$$\sum_{i=1}^{2}\left(u_i\int_0^L EA\frac{dN_j}{dx}\frac{dN_i}{dx}dx\right)=\int_0^L N_j f(x)dx+N_j\sum_{i=1}^{2}EA\frac{dN_i}{dx}u_i\Big|_0^L$$

If we use the presentation:

$$\lfloor N(x)\rfloor=\lfloor N_1(x)\quad N_2(x)\rfloor$$

We may present the above equation in the form:

$$\left(\int_0^L EA\{N_x(x)\}\lfloor N_x(x)\rfloor dx\right)\{u\}=\int_0^L\{N_x(x)\}f(x)dx+\begin{Bmatrix}p_1\\p_2\end{Bmatrix}$$

Thus, essentially, when obtaining the element matrix, we are performing the integration:

$$[K]=\int_0^L EA\{N_x(x)\}\lfloor N_x(x)\rfloor dx$$

We may also use the relation:

$$\lfloor N(x)\rfloor=\lfloor H(x)\rfloor[T_m]^{-1}$$

To rewrite the integration in the form:

$$[K]=[T^{-1}]^T\int_0^L EA\{H_x(x)\}\lfloor H_x(x)\rfloor dx[T^{-1}]$$

Which utilized the fact that the transformation matrix T is constant. This last form us particularly useful when we are going to utilize numerical integration to obtain the element matrices.

3.2 Using wxMaxima to Obtain the Interpolation Function and The Element Matrices

(You may download and install wxMaxima for free from http://andrejv.github.io/wxmaxima/)

We will use a very general approach to derive the element matrices using wxMaxima. This approach will be used as it introduces great flexibility for using in many other models (Beams, plates, etc …). The first step will be defining the *H* vector as the basic building block for the interpolation and the element matrices. The subscript *u* will be used to identify that the problem is solving for the axial deflections.

```
H:matrix([1,x]);
```

Now, wxMaxima identifies the vector as a 1x2 vector. To get the derivative, we type the line:

```
Hx:diff(H,x);
```

Which defines the slope vector for wxMaxima. The subscript *x* indicates that the variable presents the derivative with respect to x. Now, we will build the transformation matrix and get its inverse:

```
x:0;
Tb1:H,numer;
x:L;
Tb2:H,numer;
TT:matrix(Tb1[1],Tb2[1]);
T1:invert(TT);
```

The above code displays the matrix:

$$T^{-1}=\begin{bmatrix} 1 & 0 \\ \frac{-1}{L} & \frac{1}{L} \end{bmatrix}$$

To obtain the element stiffness matrix *K*, we will present it in three steps as follows:

```
Ke:E*transpose(T1).integrate(transpose(Hx).Hx*A,x,0,L).T1;
```

Which will display the matrix as:

$$K=\begin{bmatrix} \frac{EA}{L} & \frac{-EA}{L} \\ \frac{-EA}{L} & \frac{EA}{L} \end{bmatrix}$$

Which is the stiffness matrix we obtained before in section 2.3.

Note:
In the above integration, we used *H(x)* instead of *N(x)*. The relation between them is the transformation matrix *T* which is constant, thus, by extracting *T* out of the integration, it becomes much simpler since the integration will only involve simple terms. This will be ***extremely useful*** when we will use numerical integration for obtaining the element equations.

To demonstrate the flexibility of the above code, we will attempt obtaining the element matrix for an element with three nodes (adding one in the middle). See Figure 3.2.1.

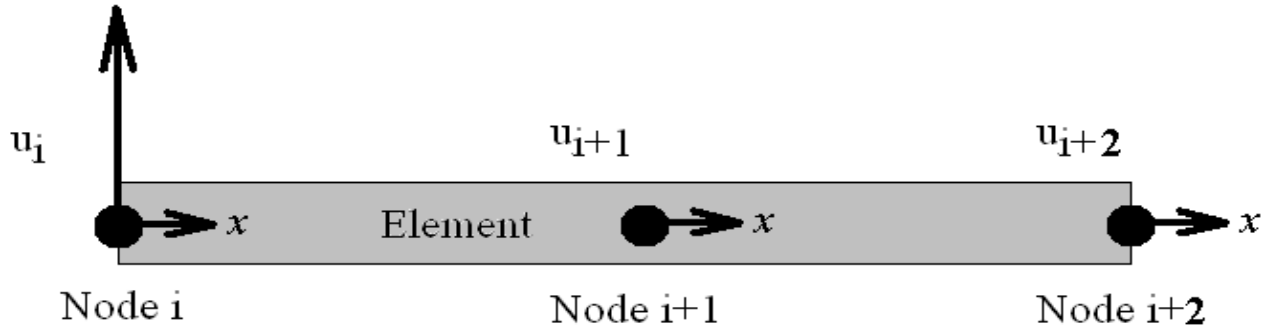

Figure 3.2.1. 3-Node Bar Element

Three nodes will require us to interpolate a parabola over the element, thus:

$$u(x)=a_0+a_1x+a_2x^2$$

From which:

$$\lfloor H(x)\rfloor=\lfloor 1 \quad x \quad x^2\rfloor$$

Incorporating that into the wxMaxima program, we get:

```
H:matrix([1,x,x*x]);
```

To obtain the transformation matrix, we need to add an extra line:

```
x:0;
Tb1:H,numer;
x:L/2;
Tb2:H,numer;
x:L;
Tb3:H,numer;
TT:matrix(Tb1[1],Tb2[1],Tb3[1]);
T1:invert(TT);
```

The above code displays the matrix:

$$T^{-1}=\begin{bmatrix} 1 & 0 & 0 \\ \frac{-3}{L} & \frac{4}{L} & \frac{-1}{L} \\ \frac{2}{L^2} & \frac{-4}{L^2} & \frac{2}{L^2} \end{bmatrix}$$

To obtain the stiffness matrix, no code need to be changed:

```
Ke:E*transpose(T1).integrate(transpose(Hx).Hx*A,x,0,L).T1;
```

And we obtain the stiffness matrix as:

$$K=\frac{EA}{3L}\begin{bmatrix} 7 & -8 & 1 \\ -8 & 16 & -8 \\ 1 & -8 & 7 \end{bmatrix}$$

Further extension of the program to create higher order elements is possible in a straight forward manner.

NOTE:

The extension of the above procedure to obtain higher order elements will be limited with the condition of the transfer matrix *T* as it will become ill-conditioned when the number of nodes becomes high [1]

3.3 Using Octave to Create the Assembled Equations and Get the Results

In the following, you will find a listing of the Octave program (which will readily run on Matlab and FreeMat packages) with remarks describing each step (remarks are preceded by %).

```
%Clearing the memory and display
clear all
clc
%Problem Data
NE=2; %number of elements
Length=2.0; %bar length
Width=0.01; %bar width
Thickness=0.01; %bar thickness
Modulus=71e9; %Aluminum modulus of elasticity
%Cross-section area
Area=Width*Thickness;
Le=Length/NE; %Element Length
%Element stiffness matrix
Ke=Modulus*Area*[1 -1; -1 1]/Le;
%Global stiffness and mass matrix assembly
%Initializing an empty matrix
KGlobal=zeros(NE+1,NE+1);
%Assembling the global matrix
for ii=1:NE
    KGlobal(ii:ii+1,ii:ii+1)= KGlobal(ii:ii+1,ii:ii+1)+Ke;
end
%For a fixed-free bar the first degree of freedom is zero
%Obtaining the auxiliary equations
KAux=KGlobal(1, 2:NE+1);
%Applying the boundary conditions
KGlobal(1,:)=[];
KGlobal(:,1)=[];
```

```
%force Vector
FGlobal=zeros(NE,1); %This is the empty force fector
FGlobal(NE)=1000; %Adding a single point load at the tip of 1000N
%Obtainning the solution displacement field
WW=inv(KGlobal)*FGlobal
%Obtaining the support reaction
Reactions= Kaux*WW
```

If we want to modify the above program so that it uses a 3-node element, the following lines will be changed:

```
%Element stiffness matrix
Ke=Modulus*Area*[7 -8 1; -8 16 -8; 1 -8 7]/Le/3;
%Global stiffness and mass matrix assembly
%Initializing an empty matrix
KGlobal=zeros(2*NE+1,2*NE+1);
%Assembling the global matrix
for ii=1:NE
    KGlobal(2*ii-1:2*ii+1,2*ii-1:2*ii+1)= ...
                  KGlobal(2*ii-1:2*ii+1,2*ii-1:2*ii+1)+Ke;
end
%For a fixed-free bar the first degree of freedom is zero
%Obtaining the auxiliary equations
KAux=KGlobal(1, 2:2*NE+1);
%Applying the boundary conditions
KGlobal(1,:)=[];
KGlobal(:,1)=[];
%force Vector
FGlobal=zeros(2*NE,1); %This is the empty force fector
FGlobal(2*NE)=1000; %Adding a single point load at the tip of
1000N
```

Note that the results obtained from the 2-node and 3-node elements will be exactly the same as they satisfy the exact solution of the bar. Nevertheless, the above demonstrates the solution procedure in both cases.

3.4 Trusses

The problem of a truss uses the same fundamental elements of the bar, however, it has two extra problems added to it:

1. Rotation of Axes. The truss elements are generally oriented in in the plane (or space), thus, the axes of all the elements need to be rotated to a common global system of axes.

2. Logistics. The truss elements may be connected to any node, and any given node may be connected to one or more elements, thus, we need to be able to keep track of which is which.

3.4.1 Rotation of Axes

For a general bar element that is oriented in the plane with its primary axis inclined to the global x-axis with an angle θ, we may draw Figure 3.4.1. We may write the global displacements in terms of the local ones using the rotation matrix as follows:

$$\begin{Bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \end{Bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & \cos(\theta) & -\sin(\theta) \\ 0 & 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{Bmatrix} u_{e1} \\ v_{e1} \\ u_{e2} \\ v_{e2} \end{Bmatrix}$$

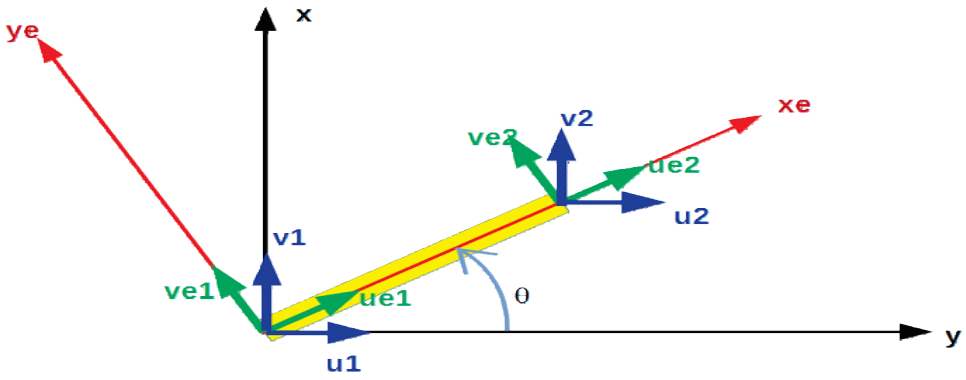

Figure 3.4.1. A generally oriented bar in the x-y plane

Similarly, we may write the equation for the rotation of the nodal forces as:

$$\begin{Bmatrix} qx_1 \\ qy_1 \\ qx_2 \\ qy_2 \end{Bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & \cos(\theta) & -\sin(\theta) \\ 0 & 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{Bmatrix} qx_{e1} \\ qy_{e1} \\ qx_{e2} \\ qy_{e2} \end{Bmatrix}$$

If we draw our attention to the element equation in local coordinates, we get:

$$\frac{EA}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{Bmatrix} u_{e1} \\ u_{e2} \end{Bmatrix} = \begin{Bmatrix} qx_{e1} \\ qx_{e2} \end{Bmatrix}$$

If we introduced the lateral deflection of the bar into the equations, the result will be two extra trivial equations and the element equation will be:

$$\frac{EA}{L}\begin{bmatrix}1&0&-1&0\\0&0&0&0\\-1&0&1&0\\0&0&0&0\end{bmatrix}\begin{Bmatrix}u_{e1}\\v_{e1}\\u_{e2}\\v_{e2}\end{Bmatrix}=\begin{Bmatrix}qx_{e1}\\0\\qx_{e2}\\0\end{Bmatrix}$$

Using the transformations created above, we get:

$$\frac{EA}{L}\begin{bmatrix}1&0&-1&0\\0&0&0&0\\-1&0&1&0\\0&0&0&0\end{bmatrix}\begin{bmatrix}\cos(\theta)&\sin(\theta)&0&0\\-\sin(\theta)&\cos(\theta)&0&0\\0&0&\cos(\theta)&\sin(\theta)\\0&0&-\sin(\theta)&\cos(\theta)\end{bmatrix}\begin{Bmatrix}u_1\\v_1\\u_2\\v_2\end{Bmatrix}$$

$$=\begin{bmatrix}\cos(\theta)&\sin(\theta)&0&0\\-\sin(\theta)&\cos(\theta)&0&0\\0&0&\cos(\theta)&\sin(\theta)\\0&0&-\sin(\theta)&\cos(\theta)\end{bmatrix}\begin{Bmatrix}qx_1\\qy_1\\qx_2\\qy_2\end{Bmatrix}$$

Multiplying both sides of the equation by the inverse of the rotation matrix from the left, we get:

$$\frac{EA}{L}\begin{bmatrix}\cos(\theta)&-\sin(\theta)&0&0\\\sin(\theta)&\cos(\theta)&0&0\\0&0&\cos(\theta)&-\sin(\theta)\\0&0&\sin(\theta)&\cos(\theta)\end{bmatrix}\begin{bmatrix}1&0&-1&0\\0&0&0&0\\-1&0&1&0\\0&0&0&0\end{bmatrix}\begin{bmatrix}\cos(\theta)&\sin(\theta)&0&0\\-\sin(\theta)&\cos(\theta)&0&0\\0&0&\cos(\theta)&\sin(\theta)\\0&0&-\sin(\theta)&\cos(\theta)\end{bmatrix}\begin{Bmatrix}u_1\\v_1\\u_2\\v_2\end{Bmatrix}$$

$$=\begin{Bmatrix}qx_1\\qy_1\\qx_2\\qy_2\end{Bmatrix}$$

Which indicates that the stiffness matrix of the element in global coordinates may be written as:

$$[K]=\frac{EA}{L}$$

$$\begin{bmatrix}\cos(\theta)&-\sin(\theta)&0&0\\\sin(\theta)&\cos(\theta)&0&0\\0&0&\cos(\theta)&-\sin(\theta)\\0&0&\sin(\theta)&\cos(\theta)\end{bmatrix}\begin{bmatrix}1&0&-1&0\\0&0&0&0\\-1&0&1&0\\0&0&0&0\end{bmatrix}\begin{bmatrix}\cos(\theta)&\sin(\theta)&0&0\\-\sin(\theta)&\cos(\theta)&0&0\\0&0&\cos(\theta)&\sin(\theta)\\0&0&-\sin(\theta)&\cos(\theta)\end{bmatrix}$$

For which we may, easily, get an explicit expression by multiplying the three matrices, however, we will leave that for the computer to evaluate.

3.4.2 The Logistic Problem

To demonstrate the logistic problem, let us consider the truss of Figure 3.4.2. That truss consists of equilateral triangles created by similar bar elements of length *L*. The truss contains 9 elements connected at 6 nodes. As you may recognize, this truss is statically determinate and we may solve it for the forces directly. However, the deflection of the nodes may turn out to be a bit tricky. To record the information of the nodes and the elements, we will select the x-y coordinated to have the origin located at node number 1. In the figure, the nodes are identified by circles around the number, while the elements are identified by squares.

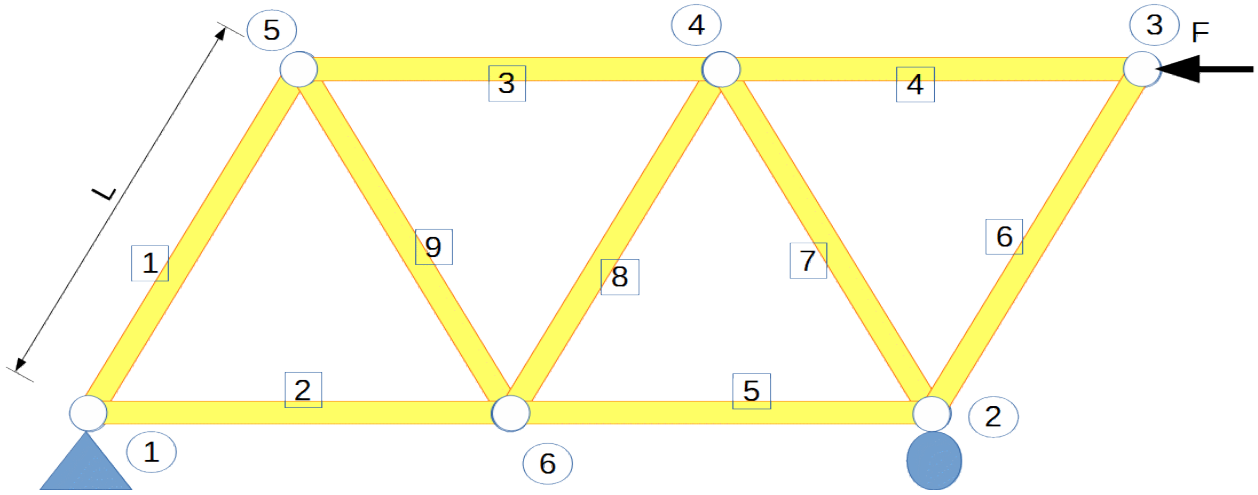

Figure 3.4.2. 6-node 9-element truss illustration

The first step would be to identify the coordinates of each node as well as the degrees of freedom it is associated with. The following table summarizes the data of the nodes:

Table 3.4.1. Data of the Nodes

Node #	X	Y	Fx	Fy	u	v
1	0	0	0	0	1	2
2	2 L	0	0	0	3	4
3	2.5 L	√3/2 L	-F	0	5	6
4	1.5 L	√3/2 L	0	0	7	8
5	0.5 L	√3/2 L	0	0	9	10
6	L	0	0	0	11	12

In the above table, we chose to number the horizontal and vertical displacements of each node in an orderly fashion, However, this is not necessary, rather, we chose to do that for convenience.

The second set of information that we need to keep track of is the element connectivity and material properties. The following table summarizes the data:

Table 3.4.2. Data of the Elements

Element #	Node#1	Node#2	E	A
1	1	5	E	A
2	1	6	E	A
3	5	4	E	A
4	4	3	E	A
5	6	2	E	A
6	2	3	E	A
7	2	4	E	A
8	6	4	E	A
9	6	5	E	A

The code that solves the above truss problem is listed in section 13.1. The last section of the program may be neglected if you are not interested in evaluating the element forces.

4. Beams

(Videos explaining this topic are available on http://AcademyOfKnowledge.org under "Introduction to the Finite Element Method: Beam Elements")

4.1 Governing Equation and Boundary Conditions

The Euler-Bernoulli beam equation has the main assumption that the lines normal to the elastic axis before bending stay normal after bending. However, it is not uncommon to assume that the material properties of the cross section are constant. Both the above above assumption result in the differential equation relating the applied forces to the deflection to be written as:

$$\frac{d^2}{dx^2}\left(EI\frac{d^2v}{dx^2}\right)=f(x)$$

The differential equation above is a fourth-order one that needs four boundary conditions in order to produce a unique solution. First type of boundary conditions would be knowing the displacement of the beam at one or more points. This may be written as:

$$v(x_1)=\delta$$

Where δ is the known value of the deflection at that point. Note here that the *usual* value is zero, nevertheless, is may have any non-zero value as we will see in some examples. The value of the displacement may be determined in the cases of pinned, roller, or fixed supports.

The second type of boundary condition may be in the form:

$$\frac{dv}{dx}=\theta$$

At such boundaries, the slope of the beam is determined at the point of the boundary condition. For small slopes, the angle of the tangent is equal to the value of the slope. As in the case of the displacement boundary condition, the usual value of the slope fixation is zero, but it can definitely take any value according to the fixation condition. Slopes of the beam may be set using fixed supports as well as sliding supports.

The third type of boundary condition is the bending moment boundary condition which may be written as:

$$EI_z\frac{d^2v}{dx^2}=M$$

For this boundary condition, the bending moment of the beam is determined at a given point. Free ends, rollers, and pinned supports are all examples of where we may be able to determine the bending moment as a boundary condition. It usually takes the value of zero unless there is a concentrated moment located at that point.

The forth, and last, type of boundary condition that we may set to the beam is the shear boundary condition.

$$EI_z \frac{d^3 v}{d x^3} = S$$

This kind of boundary conditions occurs when you know the applied lateral force at a given point. This usually happens at free ends and sliding supports.

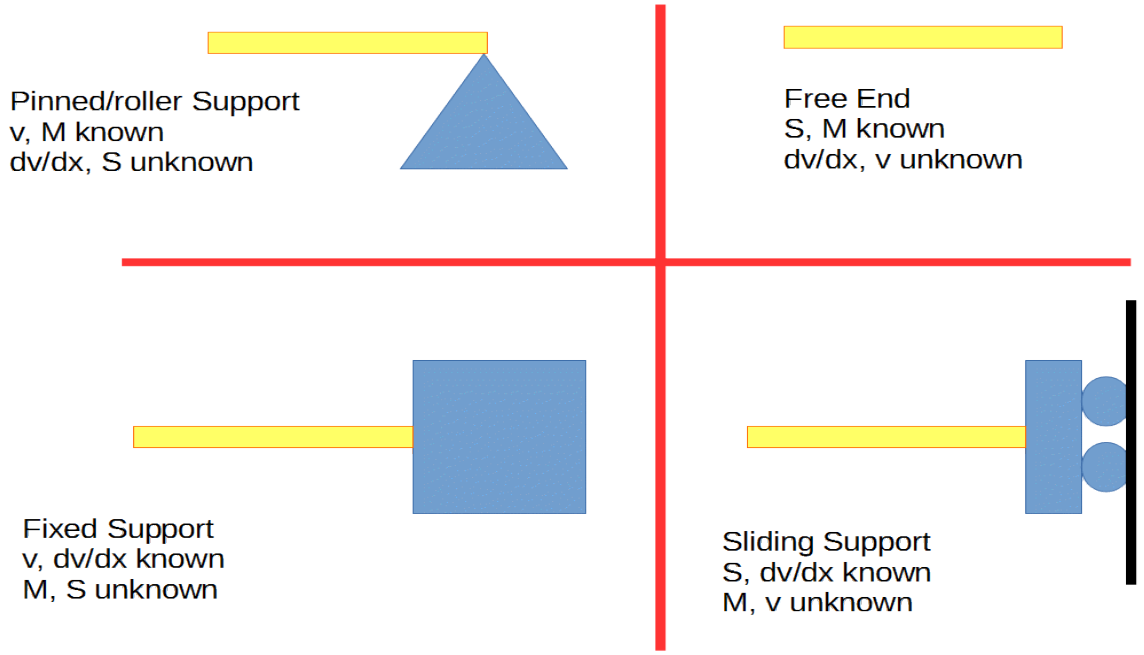

Figure 4.1.1: Summary of the 2-D beam supports and boundary conditions

4.2 Interpolation function

The interpolation function for the thin beams needs to satisfy the essential boundary conditions. In our case, the essential boundary conditions are the displacements and slopes of the beam at each

end of the element. Thus, we need to satisfy four conditions, which leads to the need of, at least, a third order polynomial. The polynomial may, generally, be written as:

$$v(x)=a_0+a_1x+a_2x^2+a_3x^3$$

Or, in vector form:

$$v(x)=\lfloor 1 \quad x \quad x^2 \quad x^3 \rfloor \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{Bmatrix} = \lfloor H(x) \rfloor \{a\}$$

The boundary conditions that need to be satisfied by this polynomial are the values of the displacements and slopes at each of the end of the beam elements presented in Figure 4.2.1.

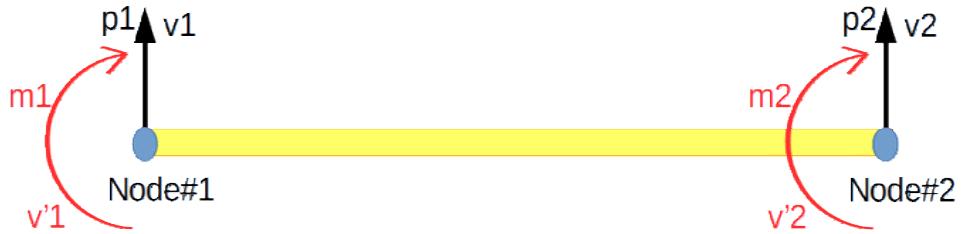

Figure 4.2.1. Beam element with end displacements, slopes, point forces, and point moments.

The slope function of the beam elements is given as:

$$\frac{dv}{dx}=v'(x)=a_1+2a_2x+3a_3x^2$$

Or:

$$\frac{dv}{dx}=\lfloor 0 \quad 1 \quad 2x \quad 3x^2 \rfloor \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{Bmatrix} = \lfloor H'(x) \rfloor \{a\}$$

Using the boundary conditions, we may get:

$$v(0)=v_1=\lfloor H(0) \rfloor \{a\}$$
$$v'(0)=v'_1=\lfloor H'(0) \rfloor \{a\}$$
$$v(L)=v_2=\lfloor H(L) \rfloor \{a\}$$
$$v'(L)=v'_2=\lfloor H'(L) \rfloor \{a\}$$

In matrix form:

$$\begin{Bmatrix} v_1 \\ v'_1 \\ v_2 \\ v'_2 \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & L & L^2 & L^3 \\ 0 & 1 & 2L & 3L^2 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{Bmatrix}$$

$$\{v\} = [T_b]\{a\}$$

Which may, readily, be solved to get:

$$\{a\} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \frac{-3}{L^2} & \frac{-2}{L} & \frac{3}{L^2} & \frac{-1}{L} \\ \frac{2}{L^3} & \frac{1}{L^2} & \frac{-2}{L^3} & \frac{1}{L^2} \end{bmatrix} \{v\}$$

$$\{a\} = [T_b]^{-1}\{v\}$$

From which we may write:

$$v(x) = \lfloor H(x) \rfloor [T_b]^{-1}\{v\} = \lfloor N(x) \rfloor \{v\}$$

where:

$$\lfloor N(x) \rfloor^T = \{N(x)\} = \begin{Bmatrix} 1 - \frac{3x^2}{L^2} + \frac{2x^3}{L^3} \\ x - \frac{2x^2}{L} + \frac{x^3}{L^2} \\ \frac{3x^2}{L^2} - \frac{2x^3}{L^3} \\ -\frac{x^2}{L^2} + \frac{x^3}{L^3} \end{Bmatrix}$$

If you plot the four functions, you will get the graph in …

4.3 Element Equations

Following the procedure of section 2.3, and using the beam equation:

$$\frac{d^2}{dx^2}EI\frac{d^2v}{dx^2}=f(x)$$

If we use the proposed solution into the differential equation, we get:

$$\sum_{i=1}^{4}\frac{d^2}{dx^2}\left(EI\frac{d^2N_i}{dx^2}v_i\right)-f(x)=R(x)$$

Applying the Galerkin method, we get:

$$\int_{x_1}^{x_2}N_jR(x)\,dx=\sum_{i=1}^{4}\int_{x_1}^{x_2}N_j\frac{d^2}{dx^2}\left(EI\frac{d^2N_i}{dx^2}v_i\right)dx-\int_{x_1}^{x_2}N_jf(x)\,dx=0$$

Where x_1 and x_2 are the boundaries of the element we are concerned with. However, since all the integrations we are performing are bound by x_1 and x_2, we may transfer the coordinates such that we perform all the integrals from 0 to L where L is the element length. We may also use the vector notation to present the four equations in the form:

$$\int_0^L\{N\}R(x)dx=\int_0^L\{N\}\frac{d^2}{dx^2}\left(EI\lfloor N''\rfloor\{v\}\right)dx-\int_0^L\{N\}f(x)\,dx=\{0\}$$

Applying the integration by parts, the above equation may be rewritten as:

$$\left(\{N\}\frac{d}{dx}\left(EI\lfloor N''\rfloor\right)\Big|_0^L-\{N'\}EI\lfloor N''\rfloor\Big|_0^L+\int_0^L EI\{N''\}\lfloor N''\rfloor dx\right)\{u\}=\int_0^L\{N\}f(x)\,dx$$

Separating the boundary evaluations, we get:

$$\int_0^L EI\{N''\}\lfloor N''\rfloor dx\{u\}=\int_0^L\{N\}f(x)\,dx$$
$$-\{N\}\frac{d}{dx}\left(EI\lfloor N''\rfloor\right)\{u\}\Big|_{x=L}+\{N\}\frac{d}{dx}\left(EI\lfloor N''\rfloor\right)\{u\}\Big|_{x=0}+\{N'\}EI\lfloor N''\rfloor\{u\}\big|_{x=L}-\{N'\}EI\lfloor N''\rfloor\{u\}\big|_{x=0}$$

The boundary integrals will directly translate into the concentrated forces and moments of Figure 4.2.1, thus the above equation may be written as:

$$\int_0^L EI\{N''\}\lfloor N''\rfloor dx\{u\}=\int_0^L\{N\}f(x)\,dx+\begin{Bmatrix}p_1\\m_1\\p_2\\m_2\end{Bmatrix}$$

Or

$$[K]\{v\}=\{q\}$$

Where [K] is called the stiffness matrix and {q} is called the generalized force vector. Note that the generalized force vector includes the effect of the distributed loads as well as the concentrated ones.

Also note that we can not write an explicit expression for the generalized force without knowing the force function. However, we may write:

$$\int_0^L f(x)\{N\} = \begin{Bmatrix} \int_0^L f(x)N_1\,dx \\ \int_0^L f(x)N_2\,dx \\ \int_0^L f(x)N_3\,dx \\ \int_0^L f(x)N_4\,dx \end{Bmatrix} = \begin{Bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{Bmatrix}$$

Meanwhile, we may get the stiffness matrix as:

$$[K] = \frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L \\ 6L & 4L^2 & -6L & 2L^2 \\ -12 & -6L & 12 & -6L \\ 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}$$

All the above procedure may be performed using Maxima or Mathematica to evaluate the stiffness matrix. The Mathematica program is presented in section 11.1 and the Maxima program is in section 12.1

4.4 Assembling the Structure Equations

Let’s consider a simple beam structure that is divided into two elements as in Figure 4.4.1. We should consider two things:

1. The displacement and slope at the second node of the first element is exactly the same as those at the first node of the second element. This should apply to ensure the continuity of the displacement and slope from one element to the neighboring one.
2. The concentrated loads maybe divided among the elements at the connecting node, but, the summation of the divided parts should add up to the total externally applied concentrated load.

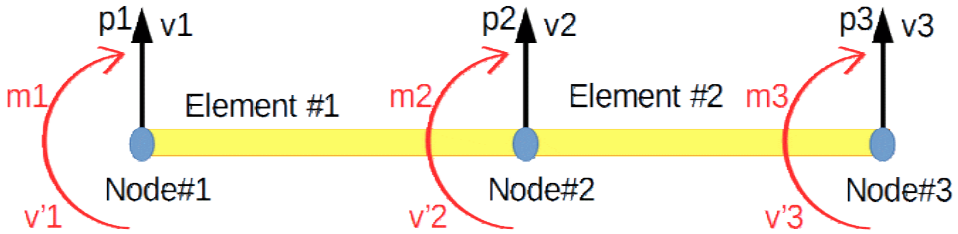

Figure 4.4.1. Two-Element beam structures.

We may write the element equation for each element to be:

$$[K^{(1)}]\{v\}^{(1)}=\{q\}^{(1)}$$

$$[K^{(2)}]\{v\}^{(2)}=\{q\}^{(2)}$$

Each of the above equations consists of four scalar equations. Now, if we want to model the whole structure, we will need to add the generalized forces to obtain the total external concentrated loads. Keeping in mind that the displacements at the connecting node is the same, we can get the equations for the assembled structure to be:

$$\frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L & 0 & 0 \\ 6L & 4L^2 & -6L & 2L^2 & 0 & 0 \\ -12 & -6L & 24 & 0 & -12 & 6L \\ 6L & 2L^2 & 0 & 8L^2 & -6L & 2L^2 \\ 0 & 0 & -12 & -6L & 12 & -6L \\ 0 & 0 & 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}\begin{Bmatrix} v_1 \\ v'_1 \\ v_2 \\ v'_2 \\ v_3 \\ v'_3 \end{Bmatrix}=\begin{Bmatrix} f_1^{(1)} \\ f_2^{(1)} \\ f_3^{(1)}+f_1^{(2)} \\ f_4^{(1)}+f_2^{(2)} \\ f_3^{(2)} \\ f_4^{(2)} \end{Bmatrix}+\begin{Bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \\ p_3 \\ m_3 \end{Bmatrix}$$

Where we assumed that the element lengths, modulus of elasticity, and second moment of area are constant for both elements. Also, the superscripts (1) and (2) denote the information from the first and second element matrices respectively.

4.5 Applying the boundary conditions

Up to this moment, all the procedure described for the finite element model may apply to any beam with any number of elements. Differences will appear when different beams are subjected to different boundary conditions. For illustration purposes, we will consider two cases here.

4.5.1 Case of a cantilever beam:

For a cantilever beam fixed from the left side (see Figure 4.5.1), we have:

$$v_1 = 0 \quad \text{and} \quad v'_1 = 0$$

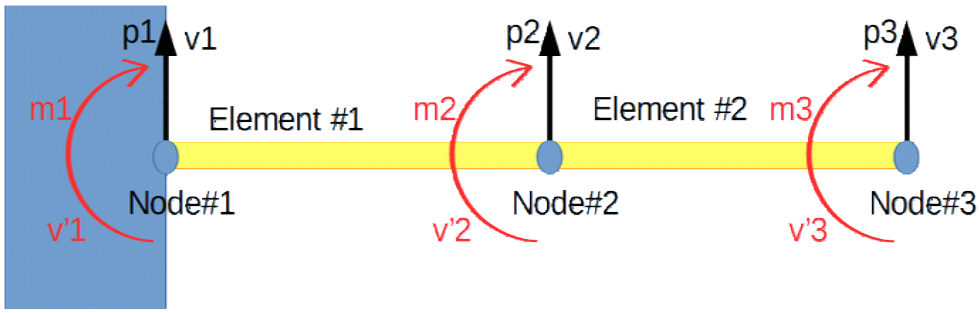

Figure 4.5.1. A Cantilever Beam.

Thus, the equation we obtained may be written as:

$$\frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L & 0 & 0 \\ 6L & 4L^2 & -6L & 2L^2 & 0 & 0 \\ -12 & -6L & 24 & 0 & -12 & 6L \\ 6L & 2L^2 & 0 & 8L^2 & -6L & 2L^2 \\ 0 & 0 & -12 & -6L & 12 & -6L \\ 0 & 0 & 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}\begin{Bmatrix} 0 \\ 0 \\ v_2 \\ v'_2 \\ v_3 \\ v'_3 \end{Bmatrix} = \begin{Bmatrix} f_1^{(1)} \\ f_2^{(1)} \\ f_3^{(1)} + f_1^{(2)} \\ f_4^{(1)} + f_2^{(2)} \\ f_3^{(2)} \\ f_4^{(2)} \end{Bmatrix} + \begin{Bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \\ p_3 \\ m_3 \end{Bmatrix}$$

Which may be separated into two distinct equations given as:

$$\frac{EI}{L^3}\begin{bmatrix} 24 & 0 & -12 & 6L \\ 0 & 8L^2 & -6L & 2L^2 \\ -12 & -6L & 12 & -6L \\ 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}\begin{Bmatrix} v_2 \\ v'_2 \\ v_3 \\ v'_3 \end{Bmatrix} = \begin{Bmatrix} f_3^{(1)} + f_1^{(2)} \\ f_4^{(1)} + f_2^{(2)} \\ f_3^{(2)} \\ f_4^{(2)} \end{Bmatrix} + \begin{Bmatrix} p_2 \\ m_2 \\ p_3 \\ m_3 \end{Bmatrix}$$

Which is called the *primary equations*. The primary equations could be readily solved for the unknown displacements and slopes by inverting the matrix on the left hand side and multiplying it by the right hand side vectors. The second equation becomes:

$$\frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L & 0 & 0 \\ 6L & 4L^2 & -6L & 2L^2 & 0 & 0 \end{bmatrix}\begin{Bmatrix} v_2 \\ v'_2 \\ v_3 \\ v'_3 \end{Bmatrix} = \begin{Bmatrix} f_1^{(1)} \\ f_2^{(1)} \end{Bmatrix} + \begin{Bmatrix} p_1 \\ m_1 \end{Bmatrix}$$

Which is called the *secondary equations* or the *auxiliary equations*. The secondary equations have the unknown variables as the concentrated force and moment at the fixed end, thus, they are the support reactions which are found by substituting the solution obtained from the primary equations directly.

4.5.2 Case of a simply-supported beam:

For a simply-supported beam (see Figure 4.5.2)we have:

$v_1 = 0$ and $v_3 = 0$

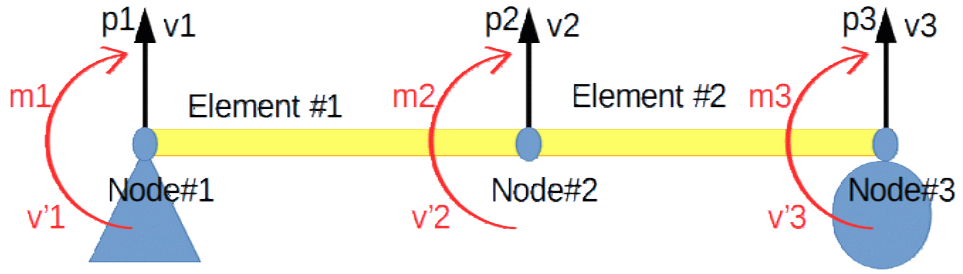

Figure 4.5.2. Simply Supported Beam.

Thus, the equation we obtained may be written as:

$$\frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L & 0 & 0 \\ 6L & 4L^2 & -6L & 2L^2 & 0 & 0 \\ -12 & -6L & 24 & 0 & -12 & 6L \\ 6L & 2L^2 & 0 & 8L^2 & -6L & 2L^2 \\ 0 & 0 & -12 & -6L & 12 & -6L \\ 0 & 0 & 6L & 2L^2 & -6L & 4L^2 \end{bmatrix} \begin{Bmatrix} 0 \\ v'_1 \\ v_2 \\ v'_2 \\ 0 \\ v'_3 \end{Bmatrix} = \begin{Bmatrix} f_1^{(1)} \\ f_2^{(1)} \\ f_3^{(1)} + f_1^{(2)} \\ f_4^{(1)} + f_2^{(2)} \\ f_3^{(2)} \\ f_4^{(2)} \end{Bmatrix} + \begin{Bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \\ p_3 \\ m_3 \end{Bmatrix}$$

which gives the primary equations as:

$$\frac{EI}{L^3}\begin{bmatrix} 4L^2 & -6L & 2L^2 & 0 \\ -6L & 24 & 0 & 6L \\ 2L^2 & 0 & 8L^2 & 2L^2 \\ 0 & 6L & 2L^2 & 4L^2 \end{bmatrix} \begin{Bmatrix} v'_1 \\ v_2 \\ v'_2 \\ v'_3 \end{Bmatrix} = \begin{Bmatrix} f_2^{(1)} \\ f_3^{(1)} + f_1^{(2)} \\ f_4^{(1)} + f_2^{(2)} \\ f_4^{(2)} \end{Bmatrix} + \begin{Bmatrix} m_1 \\ p_2 \\ m_2 \\ m_3 \end{Bmatrix}$$

And the secondary equations as:

$$\frac{EI}{L^3}\begin{bmatrix} 6L & -12 & 6L & 0 \\ 0 & -12 & -6L & -6L \end{bmatrix}\begin{Bmatrix} v'_1 \\ v_2 \\ v'_2 \\ v'_3 \end{Bmatrix} = \begin{Bmatrix} f_1^{(1)} \\ f_3^{(2)} \end{Bmatrix} + \begin{Bmatrix} p_1 \\ p_3 \end{Bmatrix}$$

Which may be used to evaluate the support reactions at both ends.

In section 13.2 you will find an Octave code that may be used to solve the beam problem described in the above section.

5. Frames

Frames are similar to trusses except for the elements that they are made of are beam elements. The beam elements making up the frame are subjected to bending loads as well as axial loads. Thus, the elements are essentially acting as beams and bars and that is reflected in the element equations by adding two degrees of freedom to represent the axial deflections in the beams.

5.1 The Element Equation in Local Coordinates

Recall for bars, the element equation was:

$$\frac{EA}{L}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}\begin{Bmatrix}u_1\\ u_2\end{Bmatrix}=\begin{Bmatrix}f_1\\ f_2\end{Bmatrix}+\begin{Bmatrix}p_1\\ p_2\end{Bmatrix}$$

While for beams:

$$\frac{EI}{L^3}\begin{bmatrix}12 & 6L & -12 & 6L\\ 6L & 4L^2 & -6L & 2L^2\\ -12 & -6L & 12 & -6L\\ 6L & 2L^2 & -6L & 4L^2\end{bmatrix}\begin{Bmatrix}v_1\\ v'_1\\ v_2\\ v'_2\end{Bmatrix}=\begin{Bmatrix}f_1\\ f_2\\ f_3\\ f_4\end{Bmatrix}+\begin{Bmatrix}p_1\\ m_1\\ p_2\\ m_2\end{Bmatrix}$$

To obtain the equations for the beam-bar element, we may go through all the derivation procedure used before for the beams and bars. We will skip all that and write down the equations directly to be:

$$\frac{E}{L}\begin{bmatrix}A & 0 & 0 & -A & 0 & 0\\ 0 & 12\frac{I}{L^2} & 6\frac{I}{L} & 0 & -12\frac{I}{L^2} & 6\frac{I}{L}\\ 0 & 6\frac{I}{L} & 4I & 0 & -6\frac{I}{L} & 2I\\ -A & 0 & 0 & A & 0 & 0\\ 0 & -12\frac{I}{L^2} & -6\frac{I}{L} & 0 & 12\frac{I}{L^2} & -6\frac{I}{L}\\ 0 & 6\frac{I}{L} & 2I & 0 & -6\frac{I}{L} & 4I\end{bmatrix}\begin{Bmatrix}u_1\\ v_1\\ v'_1\\ u_2\\ v_2\\ v'_2\end{Bmatrix}=\begin{Bmatrix}f_{a1}\\ f_1\\ f_2\\ f_{a2}\\ f_3\\ f_4\end{Bmatrix}+\begin{Bmatrix}p_{a1}\\ p_1\\ m_1\\ p_{a2}\\ p_2\\ m_2\end{Bmatrix}\quad \text{or}\quad [K]\{\delta\}=\{q\}$$

Where the subscript *a* is used to denote the axial loading (of the bar). The above element equation is one with six degrees of freedom per element that describes the relation between the axial and lateral loads to the axial and lateral deflection. Note that the axial and lateral deflections in the element equation are not coupled since we are still assuming linearity and small deflections.

5.2 Rotation of Axes

Similar to what we did in section 3.4.1, we may consider a general bar element that is oriented in the plane with its primary axis inclined to the global x-axis with an angle θ, we may draw Figure 5.2.1. We may write the global displacements in terms of the local ones using the rotation matrix as follows:

$$\begin{Bmatrix} u_1 \\ v_1 \\ v'_1 \\ u_2 \\ v_2 \\ v'_2 \end{Bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & 0 & 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{Bmatrix} u_{e1} \\ v_{e1} \\ v'_{e1} \\ u_{e2} \\ v_{e2} \\ v'_{e2} \end{Bmatrix}$$

Which may be rewritten in compact form as:

$$\{\delta\} = [R]\{\delta_e\}$$

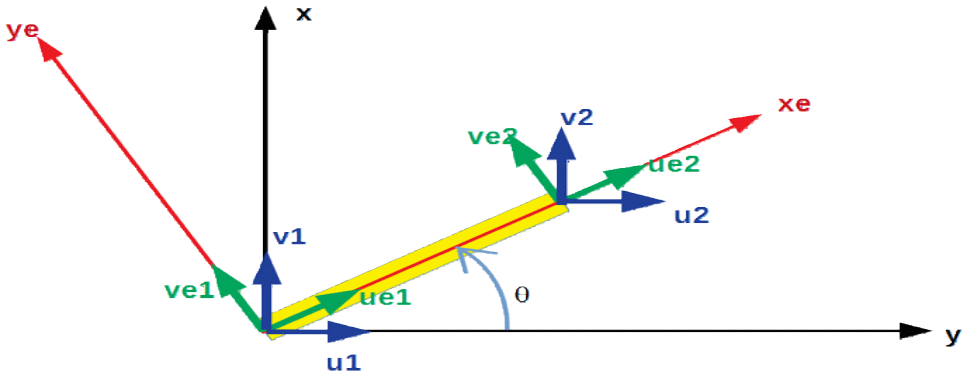

Figure 5.2.1. A generally oriented beam in the x-y plane

Similarly, we may write the equation for the rotation of the nodal forces as:

$$\{q\} = [R]\{q_e\}$$

And we may write the element equation in global coordinates as:

$$[R][K][R]^{-1} = \{q\}$$

The logistic problem of the frames is similar to that of the trusses. In section 13.3 you will find a code that solves the frame problem. The problem described in that program is the same as the truss problem described in section 3.4 with the difference of using a beam-bar element and assuming that the frame elements are bonded at the nodes instead of the pin connection of the trusses.

6. 2-D Problems

In this section, we will be discussing 2-D problems with one or more variables. The logistic problem of 2-D problems is going to be solved similar to that of the trusses and frames. Also, the element equations will be derived in a similar way to that of bars and beams but using 2-D interpolation functions. Further, we will focus on rectangular elements in order to reduce the mathematical complexity in the introductory phase.

6.1 The interpolation function

For the rectangular element in Figure 6.1.1, if we want to interpolate the function *f(x,y)* knowing the values of the function values at the nodes, we may select the first four terms of the Pascal triangle and write:

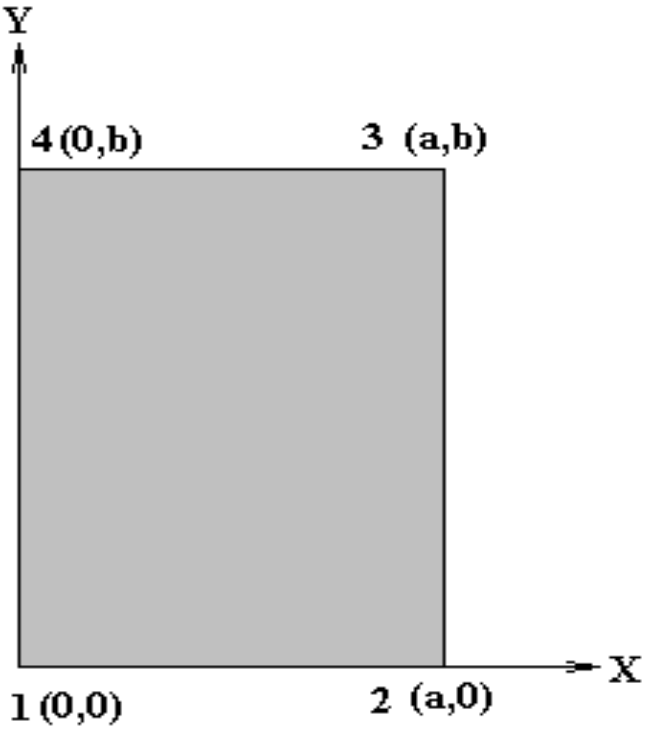

Figure 6.1.1. Rectangular Element

$$f(x,y)=a_0+a_1 x+a_2 y+a_3 x y$$

Which may be written in matrix form as:

$$f(x,y)=[H(x,y)]\{a\}$$

Now, we force the interpolation function to satisfy the values of the function at the four nodes to obtain the equations:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & a & 0 & 0 \\ 1 & a & b & ab \\ 1 & 0 & b & 0 \end{bmatrix} \begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{Bmatrix} = \begin{Bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{Bmatrix} \quad \text{or} \quad [T]\{a\} = \{f\}$$

Solving, we get:

$$\begin{Bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ \frac{-1}{a} & \frac{1}{a} & 0 & 0 \\ \frac{-1}{b} & 0 & 0 & \frac{1}{b} \\ \frac{1}{ab} & \frac{-1}{ab} & \frac{1}{ab} & \frac{-1}{ab} \end{bmatrix} \begin{Bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{Bmatrix} \quad \text{or} \quad \{a\} = [T]^{-1}\{f\}$$

Now, we may write:

$$f(x,y) = \lfloor H(x,y) \rfloor \{a\} = \lfloor H(x,y) \rfloor [T]^{-1} \{f\} = \lfloor N(x,y) \rfloor \{f\}$$

Where the Lagrange interpolation function in the vector *N(x,y)* may be written as:

$$\lfloor N(x,y) \rfloor^T = \{N(x,y)\} = \begin{Bmatrix} 1 - \frac{x}{a} - \frac{y}{b} + \frac{xy}{ab} \\ \frac{x}{a} - \frac{xy}{ab} \\ \frac{xy}{ab} \\ \frac{y}{b} - \frac{xy}{ab} \end{Bmatrix}$$

You may check that each of the functions is equal to one at one of the nodes zero at the three other nodes. In Figure 6.1.2, you may see the change of value of the third function (*N(x,y)=xy/ab*) as an illustration.

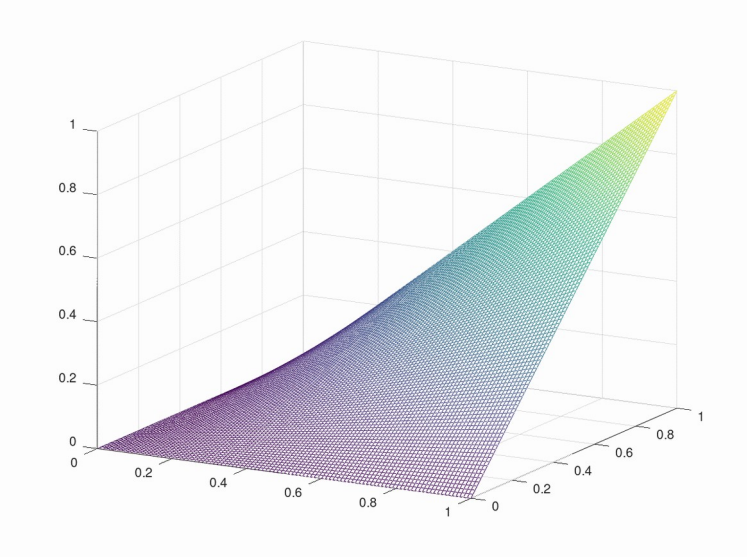

Figure 6.1.2: A Plot of N3

6.2 Laplace Equation

The Laplace equation is a second order partial differential equation that may be used to describe several physical phenomena such as the potential flow (fluid mechanics), where the unknown function may present the flow potential or the streamlines, and 2-D heat conduction where the unknown function presents the temperature distribution. The equation may be written as:

$$\frac{\partial^2 \phi}{\partial x^2}+\frac{\partial^2 \phi}{\partial y^2}=0 \quad \text{or} \quad \phi_{xx}+\phi_{yy}=0$$

Where $\phi(x,y)$ the function we are seeking the solution of. When proposing the solution

$$\phi(x,y)=\lfloor N(x,y)\rfloor\{\phi\}$$

and use it into the differential equation, we get:

$$\left(\lfloor N_{xx}\rfloor+\lfloor N_{yy}\rfloor\right)\{\phi\}=R(x,y)$$

Applying the Galerkin method and integrating over the element, we get:

$$\int_0^a\int_0^b \{N\}R(x,y)\,dy\,dx=\int_0^a\int_0^b \{N\}\left(\lfloor N_{xx}\rfloor+\lfloor N_{yy}\rfloor\right)dy\,dx\{\phi\}=0$$

Applying integration by parts, we get:

$$\int_0^a\int_0^b \left(\{N_x\}\lfloor N_x\rfloor+\{N_y\}\lfloor N_y\rfloor\right)dy\,dx\{\phi\}=\int_\Gamma \{N\}\left(\lfloor N_x\rfloor n_x+\lfloor N_y\rfloor n_y\right)\{\phi\}\,d\Gamma$$

Where the right-hand-side of the equation represents an integral over the boundary which we will get to shortly. Performing the integration of the left-hand-side, we get the element matrix as:

$$\int_0^a\int_0^b \left(\{N_x\}\lfloor N_x\rfloor+\{N_y\}\lfloor N_y\rfloor\right)dy\,dx\{\phi\}=\frac{1}{6ab}\begin{bmatrix} 2(a^2+b^2) & a^2-2b^2 & -a^2-b^2 & -2a^2+b^2 \\ a^2-2b^2 & 2(a^2+b^2) & -2a^2+b^2 & -a^2-b^2 \\ -a^2-b^2 & -2a^2+b^2 & 2(a^2+b^2) & a^2-2b^2 \\ -2a^2+b^2 & -a^2-b^2 & a^2-2b^2 & 2(a^2+b^2) \end{bmatrix}\begin{Bmatrix}\phi_1\\ \phi_2\\ \phi_3\\ \phi_4\end{Bmatrix}$$

The program that evaluates the element matrix is presented in section 11.2. Meanwhile, the left-hand-side of the equation has Γ representing the boundary, n_x and n_y represent the vector in the normal direction to the boundary in the outward direction from the element, and the *gradient* represented by the vector $\left(\lfloor N_x\rfloor n_x+\lfloor N_y\rfloor n_y\right)\{\phi\}$ represents the *inward* flux of the function. Thus the boundary integral may be rewritten as:

$$\int_\Gamma \{N\}\left(\lfloor N_x\rfloor n_x+\lfloor N_y\rfloor n_y\right)\{\phi\}\,d\Gamma=-\int_{1-2} \{N\}\lfloor N_y\rfloor\{\phi\}\,d\Gamma+\int_{2-3} \{N\}\lfloor N_x\rfloor\{\phi\}\,d\Gamma$$
$$+\int_{3-4} \{N\}\lfloor N_y\rfloor\{\phi\}\,d\Gamma-\int_{4-1} \{N\}\lfloor N_x\rfloor\{\phi\}\,d\Gamma$$

$$\int_{\Gamma} \{N\}\left(\lfloor N_x \rfloor n_x + \lfloor N_y \rfloor n_y\right)\{\phi\}\, d\Gamma =$$

$$-\int_0^a \{N\} q_{1-2}\, dx + \int_0^b \{N\} q_{2-3}\, dy + \int_a^0 \{N\} q_{3-4}\, dx - \int_b^0 \{N\} q_{4-1}\, dy$$

$$-\int_0^a \{N\} q_{1-2}\, dx + \int_0^b \{N\} q_{2-3}\, dy + \int_a^0 \{N\} q_{3-4}\, dx - \int_b^0 \{N\} q_{4-1}\, dy$$

$$= \frac{-a}{2} q_{1-2} \begin{Bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{Bmatrix} + \frac{b}{2} q_{2-3} \begin{Bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{Bmatrix} - \frac{a}{2} q_{3-4} \begin{Bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{Bmatrix} + \frac{b}{2} q_{4-1} \begin{Bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{Bmatrix}$$

6.3 Assembling the Global Equations

In this section, we will be describing a general algorithm that may be used to assemble the equations for a 2-D problem with single variable. In a 2-D problem, such as that shown in Figure 6.3.1, the node and element numbering becomes more complicated than the 1-D problem, although, an easy relation may still be obtained between the nodes' and the elements' numbering it is not advised to keep on applying this relation inside the program. Rather, a register should be filled in which each number is related to the nodes to which it is connected (Similar to what we did with trusses and frames). Figure 6.3.2 shows, graphically, the locations in the global matrix to which the element matrix entries, of element #5, should be added.

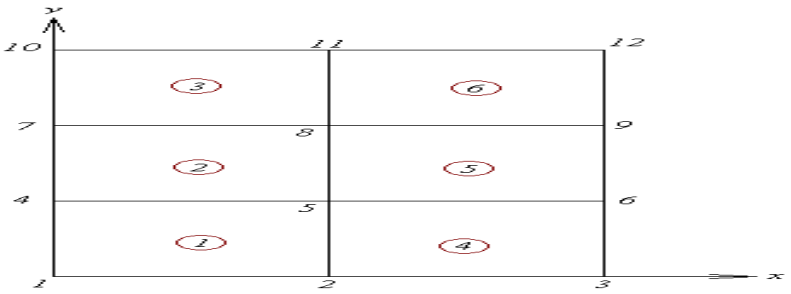

Figure 6.3.1. A sample grid in 2-D

	1	2	3	4	5	6	7	8	9	10	11	12
1												
2												
3												
4												
5					11	12		14	13			
6					21	22		24	23			
7												
8					41	42		44	43			
9					31	32		34	33			
10												
11												
12												

Figure 6.3.2. A Demonstation of the effect of an element on the global matrix

Below, is an algorithm that may be used to assemble the equations for a similar problem.

1. Create a *square* matrix "A"; N*N (N=Number of nodes)
2. For the i^{th} **element**
3. Get the element matrix "K"
4. For the j^{th} **node**
5. Get its global number k
6. For the m^{th} **node**

7. Get its global number n
8. Let $A_{kn}=A_{kn}+K_{jm}$
9. Repeat for all m
10. Repeat for all j
11. Repeat for all i

In section 13.4 you will find the octave code that may be used to solve the problem of the described above. However, you will find that the assembly algorithm used is different from the one described above. The one described above is useful using any programming language, however, in the program listed we used an algorithm that utilizes the abilities of the package Octave for convenience.

6.4 Applying the boundary conditions

After assembling the elements' of the Laplace equation problem, you will realize that the right hand side of the equations is zero, which will result in a trivial solution unless some of the values of the function are known. Those values will make up the boundary conditions of the problem. In this case, the boundary values are non-zero, so we will need to handle differently from those we handled in the earlier problems. To demonstrate, let us imagine a domain with only one element. If we know the value of the function at two nodes of the element (say nodes 1 and 2) the element equation will look like:

$$\frac{1}{6ab}\begin{bmatrix} 2(a^2+b^2) & a^2-2b^2 & -a^2-b^2 & -2a^2+b^2 \\ a^2-2b^2 & 2(a^2+b^2) & -2a^2+b^2 & -a^2-b^2 \\ -a^2-b^2 & -2a^2+b^2 & 2(a^2+b^2) & a^2-2b^2 \\ -2a^2+b^2 & -a^2-b^2 & a^2-2b^2 & 2(a^2+b^2) \end{bmatrix}\begin{Bmatrix} F_1 \\ F_2 \\ \phi_3 \\ \phi_4 \end{Bmatrix}=\{0\}$$

Which can be separated into two sets of equations:

$$\begin{bmatrix} 2(a^2+b^2) & a^2-2b^2 & -a^2-b^2 & -2a^2+b^2 \\ a^2-2b^2 & 2(a^2+b^2) & -2a^2+b^2 & -a^2-b^2 \end{bmatrix}\begin{Bmatrix} F_1 \\ F_2 \\ \phi_3 \\ \phi_4 \end{Bmatrix}=\{0\}$$

$$\begin{bmatrix} -a^2-b^2 & -2a^2+b^2 & 2(a^2+b^2) & a^2-2b^2 \\ -2a^2+b^2 & -a^2-b^2 & a^2-2b^2 & 2(a^2+b^2) \end{bmatrix}\begin{Bmatrix} F_1 \\ F_2 \\ \phi_3 \\ \phi_4 \end{Bmatrix}=\{0\}$$

For the second set, we may rewrite it as:

$$\begin{bmatrix} -a^2-b^2 & -2a^2+b^2 \\ -2a^2+b^2 & -a^2-b^2 \end{bmatrix}\begin{Bmatrix} F_1 \\ F_2 \end{Bmatrix}+\begin{bmatrix} 2(a^2+b^2) & a^2-2b^2 \\ a^2-2b^2 & 2(a^2+b^2) \end{bmatrix}\begin{Bmatrix} \phi_3 \\ \phi_4 \end{Bmatrix}=\{0\}$$

Or:

$$\begin{Bmatrix} \phi_3 \\ \phi_4 \end{Bmatrix}=\begin{bmatrix} 2(a^2+b^2) & a^2-2b^2 \\ a^2-2b^2 & 2(a^2+b^2) \end{bmatrix}^{-1}\begin{bmatrix} a^2+b^2 & 2a^2-b^2 \\ 2a^2-b^2 & a^2+b^2 \end{bmatrix}\begin{Bmatrix} F_1 \\ F_2 \end{Bmatrix}$$

Which yields the solution for the unknown values of the function. In section 13.4, you may find a code that solves the Laplace problem using rectangular finite element model for a problem similar to the one described above. For the problem, the Boundaries are set to be equal to zero on three sides of the domain and equal to unit on the fourth, this problem is known as the “Cavity Flow Problem”, the results are shown in Figure 6.4.1.

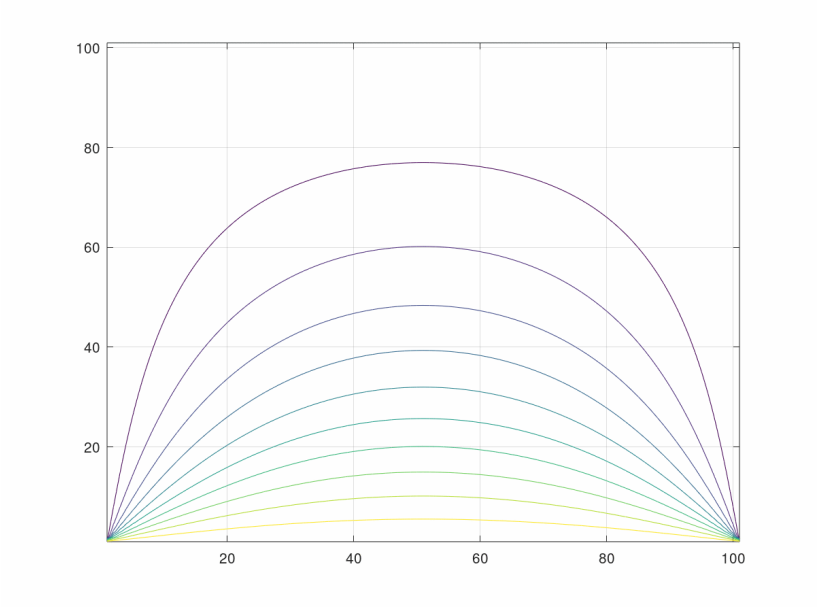

Figure 6.4.1: Cavity Flow Problem results

7. Numerical Integration Using Gauss Quadrature

Gauss Quadrature is a technique widely used for numerical integration. It, as all numerical integration techniques, transforms the integration into a summation. However, the summation is performed at some prescribed points over a domain that extends over the interval [-1,1] with weights given to the value of the function at each of the points. Thus, the physical domain [0,L] needs to be mapped to the domain of [-1,1]. Gauss quadrature method ensures the exact evaluation of the integral of polynomials over the domain according to the number of points taken in it. For example, a five-point Gauss quadrature numerical integration will give an exact integral for a 10^{th} order polynomial.[3] [4]

7.1 1-D Numerical Integration

The physical domain [0,L] may be mapped to the integration domain using the linear transformation:

$$\zeta = \frac{2x}{L} - 1 \qquad x \in [0, L], \quad \zeta \in [-1, 1]$$

$$\rightarrow d\zeta = \frac{2}{L} dx$$

Thus, the integrations transform into:

$$\int_0^L f(x)\,dx = \frac{L}{2} \int_{-1}^{1} f(\zeta)\,d\zeta = \frac{L}{2} \sum_{i=1}^{n} w_i f(\zeta_i)$$

The algorithm for evaluating the integration may be written as:

1- Initiate I=0, i=1

2- From Gauss Quadrature tables, get w_i and ζ_i

3- Evaluate the physical coordinate using $x_i = \frac{L}{2}(\zeta_i + 1)$

4- Evaluate $f(x_i)$

5- Let $I = I + w_i f(x_i)$

6- If i<n, let i=i+1, go to step 2

7- Otherwise $I = \frac{L}{2} I$

If we want to adopt the above algorithm for the evaluation of the stiffness matrix of a beam elements, it may look like the following:

1- Initiate $K=[0]_{4x4}$, i=1

2- From Gauss Quadrature tables, get w_i and ζ_i

3- Evaluate the physical coordinate using $x_i=\frac{L}{2}(\zeta_i+1)$

4- Evaluate $\lfloor H_{xx}(x_i)\rfloor$

5- Let $K=K+w_i\{H_{xx}(x_i)\}\lfloor H_{xx}(x_i)\rfloor$

6- If i<n, let i=i+1, go to step 2

7- Otherwise $K=\frac{L}{2}[T^{-1}]^T K[T^{-1}]$

Notice that we use H(x) instead of N(x) in the integration process. That was only because of the simplicity of evaluating H(x), meanwhile, since the transformation matrix is constant, we were able to easily keep it out of the integration and multiply it after the numerical evaluation. The octave program that may be used to perform the above algorithm is listed in section 13.5.

NOTE:

It might seem an *over-killing* to use numerical integration for the evaluation of the matrices of a beam and a bar. However, what we are trying to do here is building the programming skills one step at a time towards handling more complicated problems that may not be easily evaluated analytically or even do not have an analytical expression for the integration.

On the other hand, as you may see in the code, the problem of the integration is presented in its primitive building block, which can be readily used to evaluate any other matrix that may be needed in other problems as well as being easily expandable in case we need to create higher order elements (for example).

7.2 2-D Rectangular Domains

The procedure described in section 7.1 may be easily extended to 2-D rectangular domains (and 3-D cuboid domains) since the transformation from the physical domain to the Gauss domain is readily presented by a linear relation. The physical domain [0,a] [0,b] may be mapped to the integration domain using the linear transformation:

$$\xi = \frac{2x}{a} - 1 \qquad x \in [0,a], \quad \xi \in [-1,1]$$

$$\eta = \frac{2y}{b} - 1 \qquad y \in [0,b], \quad \eta \in [-1,1]$$

$$\rightarrow d\xi = \frac{2}{a} dx \text{ and } d\eta = \frac{2}{b} dy$$

Thus, the integrations transform into:

$$\int_0^a \int_0^b f(x,y)\,dy\,dx = \frac{ab}{4} \int_{-1}^{1} \int_{-1}^{1} f(\xi,\eta)\,d\eta\,d\xi = \frac{ab}{4} \sum_{i=1}^{n} \sum_{j=1}^{m} w_i w_j f(\xi_i, \eta_j)$$

The algorithm for evaluating the integration may be written as:

1- Initiate I=0, i=1, j=1

2- From Gauss Quadrature tables, get w_i and ξ_i

3- From Gauss Quadrature tables, get w_j and η_j

4- Evaluate the physical coordinates using $x_i = \frac{a}{2}(\xi_i + 1)$ and $y_j = \frac{a}{2}(\eta_j + 1)$

5- Evaluate $f(x_i, y_j)$

6- Let $I = I + w_i w_j f(x_i, y_j)$

7- If j<m, let j=j+1, go to step 3

8- Otherwise, let j=1, if i<n, let i=i+1, go to step 2

9- Otherwise $I = \frac{ab}{4} I$

If we want to adopt the above algorithm for the evaluation of the stiffness matrix of a 2-D element for the Laplace problem described in section 6.2, it may look like the following:

1- Initiate $K=[0]_{4x4}$, i=1, j=1

2- From Gauss Quadrature tables, get w_i and ξ_i

3- From Gauss Quadrature tables, get w_j and η_j

4- Evaluate the physical coordinates using $x_i = \frac{a}{2}(\xi_i + 1)$ and $y_j = \frac{a}{2}(\eta_j + 1)$

5- Evaluate $\lfloor H_x(x_i, y_j) \rfloor$ and $\lfloor H_y(x_i, y_j) \rfloor$

6- Let $K = K + w_i w_j (\{H_x(x_i, y_j)\} \lfloor H_x(x_i, y_j) \rfloor + \{H_y(x_i, y_j)\} \lfloor H_y(x_i, y_j) \rfloor)$

7- If j<m, let j=j+1, go to step 3

8- Otherwise, if i<n, let i=i+1, let j=1, go to step 2

9- Otherwise $K = \frac{ab}{4} [T^{-1}]^T K [T^{-1}]$

The Octave functions used to perform the above algorithm are listed in section 13.6.

7.3 2-D Quadrilateral Domains

In many practical 2-D problems, the elements created in the domain may not be rectangular and perfectly aligned with the physical coordinates. An example of such elements is demonstrated in Figure 7.3.1 (shown earlier in Figure 2.1.2). for such problems, the transformation from the physical domain to the Gauss domain can not be presented as linear transformations as presented in the beginning of section 7.2.

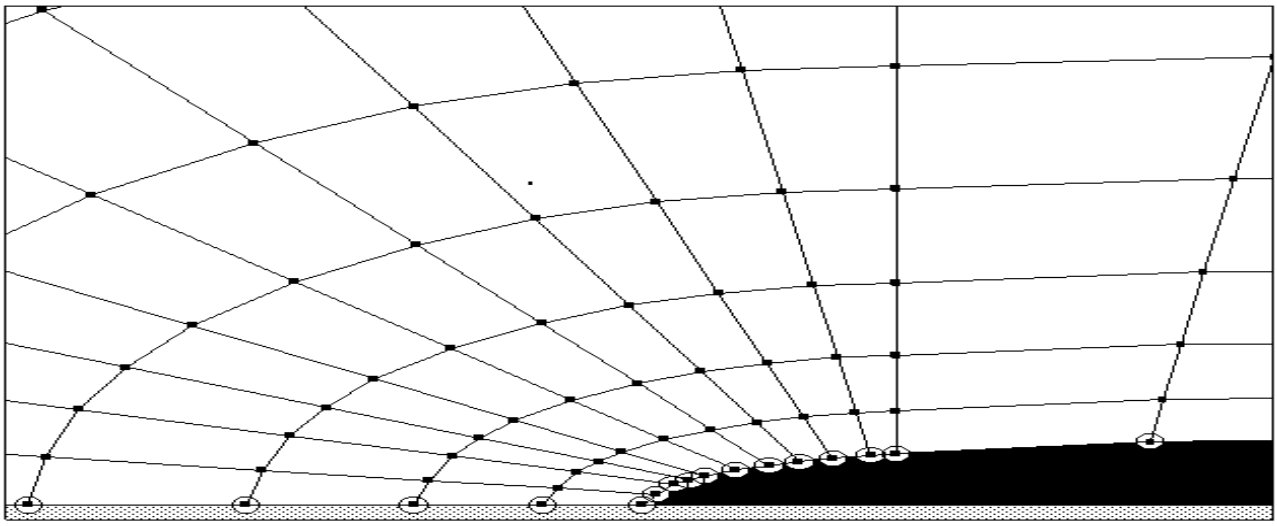

Figure 7.3.1: An example of a structured grid near the leading edge of an airfoil

For such problems, we interpolate the relations between the physical domain and the Gauss domain using polynomial functions. The best candidate for such interpolation procedure would be the same ones used for the interpolation of the solution function. This way, the same functions used to evaluate the function may be used to evaluate the domain transformation, thus, reducing the

programming effort [5] . We have to note here that using the same interpolation function may not always be the case in many problems, which will be demonstrated when attempting the modeling of later problems. We may interpolate the physical coordinates using the following relation:

$$x(\xi,\eta)=a_0+a_1\xi+a_2\eta+a_3\xi\eta=\lfloor H(\xi,\eta)\rfloor\{a\}$$

If we force this polynomial to satisfy the physical coordinates at the four corners, we get:

$$\begin{bmatrix}1 & -1 & -1 & 1\\ 1 & 1 & -1 & -1\\ 1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\end{bmatrix}\begin{Bmatrix}a_0\\ a_1\\ a_2\\ a_3\end{Bmatrix}=\begin{Bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{Bmatrix}$$

Which gives:

$$\begin{Bmatrix}a_0\\ a_1\\ a_2\\ a_3\end{Bmatrix}=\frac{1}{4}\begin{bmatrix}1 & 1 & 1 & 1\\ -1 & 1 & 1 & -1\\ -1 & -1 & 1 & 1\\ 1 & -1 & 1 & -1\end{bmatrix}\begin{Bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{Bmatrix}=[T]^{-1}\{x\}$$

Which gives the interpolation polynomial as:

$$x(\xi,\eta)=\lfloor H(\xi,\eta)\rfloor[T]^{-1}\{x\}=$$

$$\frac{1}{4}\lfloor 1-\xi-\eta+\xi\eta \quad 1+\xi-\eta-\xi\eta \quad 1+\xi+\eta+\xi\eta \quad 1-\xi+\eta-\xi\eta\rfloor\begin{Bmatrix}x_1\\ x_2\\ x_3\\ x_4\end{Bmatrix}$$

From which, we may write:

$$dx=\frac{\partial x}{\partial\xi}d\xi+\frac{\partial x}{\partial\eta}d\eta=(\lfloor H_\xi\rfloor d\xi+\lfloor H_\eta\rfloor d\eta)[T]^{-1}\{x\}$$

Similarly, we may write:

$$y(\xi,\eta)=b_0+b_1\xi+b_2\eta+b_3\xi\eta=\lfloor H(\xi,\eta)\rfloor\{b\}$$

and:

$$dy=\frac{\partial y}{\partial\xi}d\xi+\frac{\partial y}{\partial\eta}d\eta=(\lfloor H_\xi\rfloor d\xi+\lfloor H_\eta\rfloor d\eta)[T]^{-1}\{y\}$$

Thus, the integration over the 2-D domain may be written as:

$$I=\iint_{Element} f(x,y)\,dy\,dx=\iint_{Element} f(x,y)\left(\left(\frac{\partial x}{\partial\xi}d\xi+\frac{\partial x}{\partial\eta}d\eta\right)\left(\frac{\partial y}{\partial\xi}d\xi+\frac{\partial y}{\partial\eta}d\eta\right)\right)$$

Multiplying t he brackets and ignoring the second order infinitesimal terms, we get:

$$I = \iint_{Element} f(x,y)\,dy\,dx = \iint_{Element} f(x,y)\left(\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} + \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}\right) d\eta\, d\xi$$
$$= \sum_{i=1}^{n}\sum_{j=1}^{m} w_i w_j f(x_i, y_j)\left(\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} + \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}\right)_{x_i, y_j}$$

Which is similar to the relation we got for the rectangular elements with an evaluation of the derivatives of the physical domain with respect to the Gauss coordinates evaluated at each point of the domain. The code that needs to be changed from the case of rectangular to quadrilateral 2-D problem is presented in section 13.7.

8. Variational Methods and Hamilton's Principle

(Videos explaining this topic are available on http://AcademyOfKnowledge.org under "Introduction to the Finite Element Method: Stationary Functional Approach")

8.1 Simplified Concepts of Functional and Variation

A *functional* can be defined, simply, as a function of functions that has a scalar value. The functionals we use in our work are in the form of integration of functions on the domain of interest. For example:

$$\Pi = \int_{x_1}^{x_2} f(x)\,dx$$

This is a functional that is defined as the integral of f(x) over the domain [x1,x2]. As you may notice, the resultant of the integral will be a scalar. The *variation* of the functional is similar to the differentiation of the integral with respect to the function f(x). Such that:

$$\delta\Pi = \int_{x_1}^{x_2} \delta f(x)\,dx$$

The above equations says that the variation of Π is equal to the integration of the variation of f(x) over the domain [x1,x2]. The variation of the function f(x) is said to be equal to zero where the values of the function are defined.

$$if\ f(x_3) = f_3 \rightarrow \delta f(x_3) = 0$$

The rules of differentiation apply directly to variations. For example, if the functional is related to the square of a function, then the variation of the functional is given as:

$$\Pi = \int_{x_1}^{x_2} f^2(x)\,dx$$

$$\delta\Pi = \int_{x_1}^{x_2} 2f(x)\delta f(x)\,dx$$

Also, the product rule applies:

$$\Pi = \int_{x_1}^{x_2} f(x)g(x)\,dx$$

$$\delta\Pi = \int_{x_1}^{x_2} (g(x)\delta f(x) + f(x)\delta g(x))\,dx$$

These relations are what we need to work with in the coming parts.

8.2 Hamilton's Principle

Hamilton's principle, in simple terms, is an approach for obtaining the equations of motion for mechanical systems through the manipulation of the energy terms. In mechanical systems, there are three main types of energy that we are concerned with.

1. The kinetic energy which reflects the motion information of the masses
2. The potential energy which is concerned with the conservative fields of energy such as the elastic and gravitational energy terms
3. The external work which is concerned with non-conservative work done by external forces or damping elements

Hamilton's principle defines the total energy of the system at any given point as:

$$\Pi = T - U + W_{ex}$$

Then the principle states that the variation of the total energy over an arbitrary period of time should sum up to zero, or in mathematical terms:

$$\int_{t_1}^{t_2} \delta \Pi \, dt = 0$$

The above relation may be rewritten in terms of the components of the total energy to read:

$$\int_{t_1}^{t_2} (\delta T - \delta U + \delta W_{ex}) dt = 0$$

Hamilton's principle contains everything related to the dynamics of the system. When manipulated to obtain the differential equations, the principle expresses the boundary and initial conditions as well. For static problems, the Hamilton's principle reduces to:

$$\delta \Pi = \delta W_{ex} - \delta U = 0$$

Which is what we are going to be using in the following sections to *re-derive* the load-deflection relations for different structural elements.

8.3 The Finite Element Model Using Hamilton's Principle

8.3.1 Bar Element Equations

For the bar element in Figure 8.3.1, we may write down the energy equations as derived earlier to obtain the external work and potential energy as:

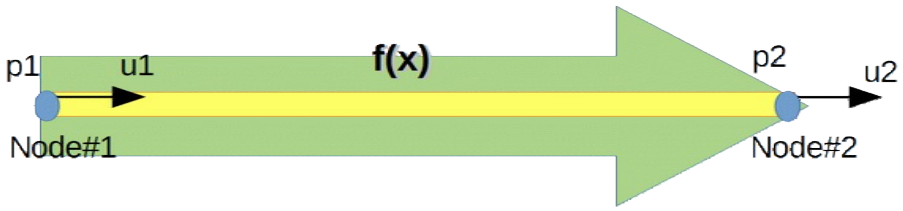

Figure 8.3.1: Bar Element

$$W_{ex}=\int_0^L f(x)u\,dx+p_1u\big|_{x=0}+p_2u\big|_{x=L}$$

Recall that the strain energy is defined as the integration, over the structure, of half the stress times the strain:

$$U=\frac{1}{2}\int_{Volume}\sigma\,\epsilon\,dV$$

Also, for linear materials, the stress may be expressed as the product of the modulus of elasticity times the strain $\sigma=E\epsilon$, thus:

$$U=\frac{1}{2}\int_{Volume}\sigma\,\epsilon\,dV=\frac{1}{2}\int_{Volume}E\,\epsilon^2\,dV$$

Finally, we may recall that the axial strain of a bar is equal to the rate of change of displacement with respect to the axial direction, $\epsilon=\frac{du}{dx}$, thus:

$$U=\frac{1}{2}\int_{Volume}E\,\epsilon^2\,dV=\frac{1}{2}\int_{Volume}E\left(\frac{du}{dx}\right)^2dV$$

Applying the variation on both external work and potential energy, we get:

$$\delta W_{ex}=\int_0^L f(x)\delta u\,dx+p_1\delta u\big|_{x=0}+p_2\delta u\big|_{x=L}$$

$$\delta U = \int_{Volume} E\,\delta\left(\frac{du}{dx}\right).\left(\frac{du}{dx}\right) dV$$

Remember that the strain in a bar was assumed to be independent of the y and z directions (review mechanics of material fundamentals), and if the modulus of elasticity is constant over any given cross-section, homogeneous cross-section, we may perform the y and z-integrations to obtain the area, thus the equation becomes:

$$\delta U = \int_0^L EA\,\delta\left(\frac{du}{dx}\right).\left(\frac{du}{dx}\right) dx$$

If we use the potential energy and external work terms into the Hamilton's principle, we get:

$$\delta \Pi = \int_0^L \left(EA\,\delta\left(\frac{du}{dx}\right)\left(\frac{du}{dx}\right) - f(x)\,\delta u \right) dx - p_1\,\delta u\big|_{x=0} - p_2\,\delta u\big|_{x=L} = 0$$

Which is called the reduced form. If we use the interpolation functions we derived in section 2.2, we get:

$$\frac{du}{dx} = \lfloor H'(x) \rfloor [T_m]^{-1} \{u\} = \lfloor N'(x) \rfloor \{u\}$$

Where:

$$\lfloor H'(x) \rfloor = \lfloor 0 \quad 1 \rfloor$$

$$\lfloor N(x) \rfloor^T = \{N(x)\} = \begin{Bmatrix} 1 - \frac{x}{L} \\ \frac{x}{L} \end{Bmatrix}$$

$$\lfloor N'(x) \rfloor^T = \{N'(x)\} = \begin{Bmatrix} -\frac{1}{L} \\ \frac{1}{L} \end{Bmatrix}$$

when applying the variation, we get:

$$\delta u(x) = \lfloor H(x) \rfloor [T_m]^{-1} \{\delta u\} = \lfloor N(x) \rfloor \{\delta u\}$$

$$\delta u'(x) = \lfloor H'(x) \rfloor [T_m]^{-1} \{\delta u\} = \lfloor N'(x) \rfloor \{\delta u\}$$

Note also that the transpose of a scalar is itself, thus we may utilize this identity:

$$\delta u(x) = (\lfloor N(x) \rfloor \{\delta u\})^T = \lfloor \delta u \rfloor \{N(x)\}$$

$$\delta u'(x) = (\lfloor N'(x) \rfloor \{\delta u\})^T = \lfloor \delta u \rfloor \{N'(x)\}$$

Using the results above into the Hamilton's principle, we get:

$$\delta\Pi=\lfloor\delta u\rfloor\left(\int_0^L\left(EA\{N'\}\lfloor N'\rfloor\{u\}-f(x)\{N\}\right)dx-p_1\{N\}\big|_{x=0}-p_2\{N\}\big|_{x=L}\right)=0$$

Which implies that:

$$\int_0^L\left(EA\{N'\}\lfloor N'\rfloor\{u\}-f(x)\{N\}\right)dx-p_1\{N\}\big|_{x=0}-p_2\{N\}\big|_{x=L}=0$$

Note that:

$$\lfloor N\rfloor_{x=0}=\lfloor 1 \quad 0\rfloor$$
$$\lfloor N\rfloor_{x=L}=\lfloor 0 \quad 1\rfloor$$

Thus, the equation becomes:

$$\int_0^L EA\{N'\}\lfloor N'\rfloor dx\{u\}=\int_0^L f(x)\{N\}+\begin{Bmatrix}p_1\\p_2\end{Bmatrix}$$

Or

$$[K]\{u\}=\{q\}$$

Where [K] is called the stiffness matrix and {q} is called the generalized force vector. Note that the generalized force vector includes the effect of the distributed loads as well as the concentrated ones. Also note that we can not write an explicit expression for the generalized force without knowing the force function. However, we may write:

$$\int_0^L f(x)\{N\}=\begin{Bmatrix}\int_0^L f(x)N_1dx\\ \int_0^L f(x)N_2dx\end{Bmatrix}=\begin{Bmatrix}f_1\\f_2\end{Bmatrix}$$

Meanwhile, we may get the stiffness matrix as:

$$[K]=\frac{EA}{L}\begin{bmatrix}1 & -1\\ -1 & 1\end{bmatrix}$$

Which is exactly, the same matrix we obtained earlier in section 2.3.

8.3.2 Beam Element Equation

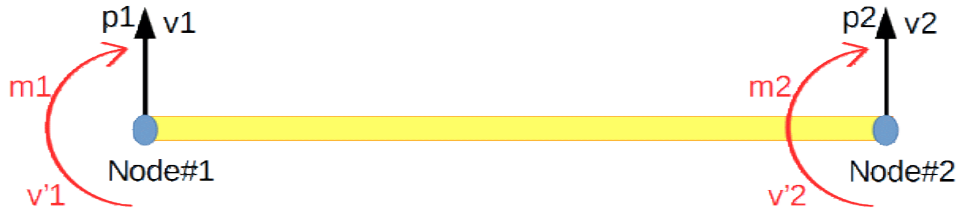

Figure 8.3.2: Beam Element

For a thin beam element shown in Figure 8.3.2 with distributed lateral load, we may write:

$$W_{ex}=\int_0^L f(x)v\,dx+p_1v_1+m_1\theta_1+p_2v_2+m_2\theta_2$$

Where θ_1 and θ_2 are the angles rotated by the nodes 1 and 2 respectively. For small angles, we may approximate the rotations by the slopes of the beam at those points, thus, we may write:

$$W_{ex}=\int_0^L f(x)v\,dx+p_1v_1+m_1v'_1+p_2v_2+m_2v'_2$$

Obtaining the variation of the external work, we get:

$$\delta W_{ex}=\int_0^L f(x)\delta v\,dx+p_1\delta v|_{x=0}+m_1\delta v'|_{x=0}+p_2\delta v|_{x=L}+m_2\delta v'|_{x=L}$$

For the elastic energy:

$$U=\frac{1}{2}\int_{Volume}\sigma\epsilon\,dV=\frac{1}{2}\int_{Volume}E\epsilon^2\,dV=\frac{1}{2}\int_{Volume}E\left(\frac{d^2v}{dx^2}\right)^2y^2dV$$

Again, we have the deflection of the neutral axis is independent of the y and z-directions, and if we have the modulus of elasticity constant over the cross-section, we get:

$$U=\frac{1}{2}\int_0^L E\left(\frac{d^2v}{dx^2}\right)^2\left(\int_{Area}y^2dy\,dz\right)dx=\frac{1}{2}\int_0^L EI\left(\frac{d^2v}{dx^2}\right)^2dx$$

Obtaining the variation, we get:

$$\delta U=\int_0^L EI\left(\frac{d^2v}{dx^2}\right).\delta\left(\frac{d^2v}{dx^2}\right)dx$$

Into the Hamilton's principle:

$$\delta\Pi=\int_0^L\left(EI\left(\frac{d^2v}{dx^2}\right)\delta\left(\frac{d^2v}{dx^2}\right)-f(x)\delta v-p_1\delta v\big|_{x=0}-p_2\delta v\big|_{x=L}-m_1\delta\frac{dv}{dx}\bigg|_{x=0}-m_2\delta\frac{dv}{dx}\bigg|_{x=L}\right)dx=0$$

If we use the interpolation functions we derived in the section 4.2, we get:

$$\frac{dv}{dx}=\lfloor H'(x)\rfloor[T_b]^{-1}\{v\}=\lfloor N'(x)\rfloor\{v\}$$

$$\frac{d^2v}{dx^2}=\lfloor H''(x)\rfloor[T_b]^{-1}\{v\}=\lfloor N''(x)\rfloor\{v\}$$

Where:

$$\lfloor H'(x)\rfloor=\begin{bmatrix}0 & 1 & 2x & 3x^2\end{bmatrix}$$

$$\lfloor H''(x)\rfloor=\begin{bmatrix}0 & 0 & 2 & 6x\end{bmatrix}$$

$$\lfloor N(x)\rfloor^T=\{N(x)\}=\begin{Bmatrix}1-\frac{3x^2}{L^2}+\frac{2x^3}{L^3}\\ x-\frac{2x^2}{L}+\frac{x^3}{L^2}\\ \frac{3x^2}{L^2}-\frac{2x^3}{L^3}\\ -\frac{x^2}{L^2}+\frac{x^3}{L^3}\end{Bmatrix}$$

$$\lfloor N'(x)\rfloor^T=\{N'(x)\}=\begin{Bmatrix}\frac{-6x}{L^2}+\frac{6x^2}{L^3}\\ 1-\frac{4x}{L}+\frac{3x^2}{L^2}\\ \frac{6x}{L^2}-\frac{6x^2}{L^3}\\ -\frac{2x}{L^2}+\frac{3x^2}{L^3}\end{Bmatrix}$$

$$\lfloor N''(x)\rfloor^T=\{N''(x)\}=\begin{Bmatrix}\frac{-6}{L^2}+\frac{12x}{L^3}\\ \frac{-4}{L}+\frac{6x}{L^2}\\ \frac{6}{L^2}-\frac{12x}{L^3}\\ -\frac{2}{L^2}+\frac{6x}{L^3}\end{Bmatrix}$$

when applying the variation, we get:

$$\delta v(x)=\lfloor H(x)\rfloor[T_b]^{-1}\{\delta v\}=\lfloor N(x)\rfloor\{\delta v\}$$

$$\delta v'(x)=\lfloor H'(x)\rfloor[T_b]^{-1}\{\delta v\}=\lfloor N'(x)\rfloor\{\delta v\}$$

$$\delta v''(x)=\lfloor H''(x)\rfloor[T_b]^{-1}\{\delta v\}=\lfloor N''(x)\rfloor\{\delta v\}$$

Using the results above into the Hamilton's principle, we get:

$$\delta\Pi=$$
$$\{\delta v\}^T\int_0^L\left(EI\{N''\}\lfloor N''\rfloor\{v\}-f(x)\{N\}-p_1\{N\}\big|_{x=0}-p_2\{N\}\big|_{x=L}-m_1\{N'\}\big|_{x=0}-m_2\{N'\}\big|_{x=L}\right)dx=0$$

Note that:

$$\lfloor N\rfloor_{x=0}=\lfloor 1\quad 0\quad 0\quad 0\rfloor$$
$$\lfloor N'\rfloor_{x=0}=\lfloor 0\quad 1\quad 0\quad 0\rfloor$$
$$\lfloor N\rfloor_{x=L}=\lfloor 0\quad 0\quad 1\quad 0\rfloor$$
$$\lfloor N'\rfloor_{x=L}=\lfloor 0\quad 0\quad 0\quad 1\rfloor$$

Thus, the equation becomes:

$$\int_0^L EI\{N''\}\lfloor N''\rfloor dx\{v\}=\int_0^L f(x)\{N\}+\begin{Bmatrix}p_1\\m_1\\p_2\\m_2\end{Bmatrix}$$

Or

$$[K]\{v\}=\{q\}$$

Where [K] is called the stiffness matrix and {q} is called the generalized force vector. Note that the generalized force vector includes the effect of the distributed loads as well as the concentrated ones. Also note that we can not write an explicit expression for the generalized force without knowing the force function. However, we may write:

$$\int_0^L f(x)\{N\} = \begin{Bmatrix} \int_0^L f(x)N_1\,dx \\ \int_0^L f(x)N_2\,dx \\ \int_0^L f(x)N_3\,dx \\ \int_0^L f(x)N_4\,dx \end{Bmatrix} = \begin{Bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{Bmatrix}$$

Meanwhile, we may get the stiffness matrix as:

$$[K] = \frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L \\ 6L & 4L^2 & -6L & 2L^2 \\ -12 & -6L & 12 & -6L \\ 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}$$

Which, again, is the same matrix we obtained using the Galerkin method in section 4.3. Now, we may utilize this method to try and create finite element models for more sophisticated problems.

9. Further Problems Involving Beams

9.1 Buckling of Columns

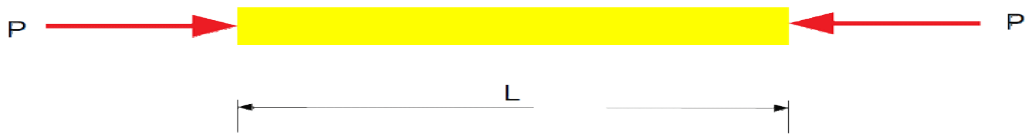

Figure 9.1.1. Beam under axial loading.

When a beam (column) is under compression (see Figure 9.1.1) we may ignore the axial strain and deflection. We get he external work expressed as:

$$W_{ex}=\int_0^L f(x)v\,dx+\frac{1}{2}\int_0^L P\left(\frac{dv}{dx}\right)^2 dx$$

Note:

The extra term obtained in the external work expression is due to the work done by the axial load in the bending direction (Figure 9.1.2). If we consider an infinitesimal length of the beam that has a length of *dx* and becomes deflected with a slope *dv/dx*, the sheer component of the axial load (*P sin(θ)*) will exert an infinitesimal work that may be given by:

$$d\,W_{ex}=\frac{1}{2}P\sin(\theta)dx\tan(\theta)$$

For a small angle, we may write:

$$d\,W_{ex}=\frac{1}{2}P\frac{dv}{dx}dx\frac{dv}{dx}=\frac{1}{2}P\left(\frac{dv}{dx}\right)^2 dx$$

Meanwhile, the one half factor appears because the work builds up with the deflection just like that of a spring.

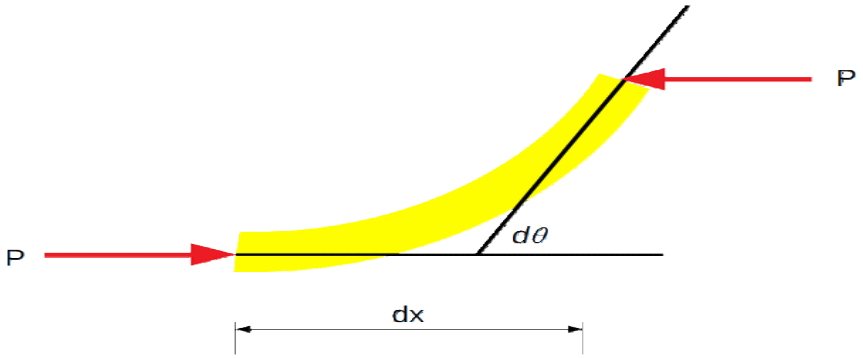

Figure 9.1.2. Work done by the axial loading.

Obtaining the variation of the external work, we get:

$$\delta W_{ex} = \int_0^L f(x)\,\delta v\,dx + \int_0^L P\left(\frac{dv}{dx}\right)\delta\left(\frac{dv}{dx}\right)dx$$

For the elastic energy, we get:

$$U = \frac{1}{2}\int_{Volume} \sigma\,\epsilon\,dV = \frac{1}{2}\int_{Volume} E\,\epsilon^2\,dV$$

But, in this case, we have:

$$\epsilon = -y\frac{d^2v}{dx^2}$$

Which results in the expression of the elastic energy in the form:

$$U = \frac{1}{2}\int_{Volume} E\left(\frac{d^2v}{dx^2}\right)^2 y^2\,dV$$

Obtaining the variation, we get:

$$\delta U = \int_{Volume} E\left(\frac{d^2v}{dx^2}\right)\delta\left(\frac{d^2v}{dx^2}\right)y^2\,dV$$

Into Hamilton's principle, we get:

$$\delta\Pi = \int_0^L \left(EI\left(\frac{d^2v}{dx^2}\right)\delta\left(\frac{d^2v}{dx^2}\right) - P\frac{dv}{dx}\delta\frac{dv}{dx} - f(x)\,\delta v - p_1\,\delta v\big|_{x=0} - p_2\,\delta v\big|_{x=L} - m_1\,\delta\frac{dv}{dx}\bigg|_{x=0} - m_2\,\delta\frac{dv}{dx}\bigg|_{x=L}\right)dx = 0$$

Which is similar to the equation we had for section 8.3.2 with an extra term due to the axial load, namely:

$$\int_0^L P \frac{dv}{dx} \delta \frac{dv}{dx} dx$$

When using the beam interpolation functions, we get:

$$\int_0^L P \frac{dv}{dx} \delta \frac{dv}{dx} dx = \{\delta v\} \int_0^L P \{N'\}[N'] dx \{v\} = \{\delta v\} P[K_G]\{v\}$$

Where the matrix $[K_G]$ is called the geometric stiffness matrix. If we perform the integration, the matrix will look like:

$$[K_G] = \frac{1}{30L} \begin{bmatrix} 36 & 3L & -36 & 3L \\ 3L & 4L^2 & -3L & -L^2 \\ -36 & -3L & 36 & -3L \\ 3L & -L^2 & -3L & 4L^2 \end{bmatrix}$$

Which when used in the element equation, we get the element equation for a column element as:

$$([K] - P[K_G])\{v\} = \{q\}$$

This element equation gets assembled using the same procedure described before and the boundary condition get applied as in the same way. However, since we are interested in the buckling load, we consider the case without externally applied load. For that case, the equation becomes:

$$([K] - P[K_G])\{v\} = \{0\}$$

Which will give a trivial solution except in the case of:

$$|[K] - P[K_G]| = 0$$

Which is an Eigenvalue problem with the eigenvalues giving the critical axial loads *P*. The associated Eigenvectors will determine the buckling shape of the column. The code solving for the buckling load of a column is listed in section 13.8.

9.2 Dynamics of Beams

9.2.1 Equations of Motion

We will be using Hamilton's principle to find the equations of motion for an Euler-Bernoulli beam. Assuming that the cross-section is homogeneous (constant modulus of elasticity, we may write the elastic energy as:

$$U=\frac{1}{2}\int_0^L EI\left(\frac{\partial^2 v}{\partial x^2}\right)^2 dx$$

Applying the variation:

$$\delta U=\int_0^L EI\left(\frac{\partial^2 v}{\partial x^2}\right)\delta\left(\frac{\partial^2 v}{\partial x^2}\right)dx$$

Applying integration by parts twice:

$$\delta U=\left(EI\frac{\partial^2 v}{\partial x^2}\delta\left(\frac{\partial v}{\partial x}\right)\right)\Bigg|_0^L-\left(\frac{\partial}{\partial x}\left(EI\frac{\partial^2 v}{\partial x^2}\right)\delta v\right)\Bigg|_0^L+\int_0^L\frac{\partial^2}{\partial x^2}\left(EI\left(\frac{\partial^2 v}{\partial x^2}\right)\right)\delta v\,dx$$

For the kinetic energy:

$$T=\frac{1}{2}\int_0^L \rho A\dot{v}^2 dx$$

Taking the variation of the kinetic energy:

$$\delta T=\int_0^L \rho A\dot{v}\,\delta\dot{v}\,dx$$

Applying the time integral:

$$\int_{t_1}^{t_2}\delta T\,dt=\int_{t_1}^{t_2}\int_0^L \rho A\dot{v}\,\delta\dot{v}\,dx\,dt=\int_0^L\left(\left(\rho A\dot{v}\,\delta v\right)\Big|_{t_1}^{t_2}-\int_{t_1}^{t_2}\rho A\ddot{v}\,\delta v\,dt\right)dx$$

For the work done by external forces:

$$W_{ex}=\int_0^L f(x,t)v\,dx$$

Applying the variation and tim integration:

$$\int_{t_1}^{t_2}\delta W_{ex}\,dt=\int_{t_1}^{t_2}\int_0^L f(x,t)\,\delta v\,dx\,dt$$

Into Hamilton's principle, and separating integration terms from boundary terms we get:

$$\rho A \ddot{v} + \frac{\partial^2}{\partial x^2}\left(EI \frac{\partial^2 v}{\partial x^2} \right) = f(x,t)$$

With boundary conditions:

$$\left(EI \frac{\partial^2 v}{\partial x^2} \delta\left(\frac{\partial v}{\partial x} \right) \right)\Bigg|_0^L = 0$$

$$\left(\frac{\partial}{\partial x}\left(EI \frac{\partial^2 v}{\partial x^2} \right) \delta v \right)\Bigg|_0^L = 0$$

And the initial conditions:

$$\left(\rho A \dot{v} \delta v \right)\Big|_{t_1}^{t_2}$$

9.2.2 Element Equations

To derive the element equations for the finite element method, we will use the *reduced form*. The reduced form, in simple terms, is the form of the energy equation *before* applying the integration by parts. From section 9.2.1, we may recall the reduced form to be:

$$\int_{t_1}^{t_2} \delta \Pi \, dt = \int_{t_1}^{t_2} \int_0^L \left(\begin{array}{c} -\rho A \left(\frac{\partial^2 v}{\partial t^2} \right) \delta v - EI \left(\frac{\partial^2 v}{\partial x^2} \right) \delta \left(\frac{\partial^2 v}{\partial x^2} \right) \\ + f(x,t) \delta v + p_1 \delta v\big|_{x=0} + p_2 \delta v\big|_{x=L} + m_1 \delta \left. \frac{dv}{dx} \right|_{x=0} + m_2 \delta \left. \frac{dv}{dx} \right|_{x=L} \end{array} \right) dx \, dt = 0$$

If we use the interpolation functions we derived before, we get:

$$\frac{dv}{dx} = \lfloor H'(x) \rfloor [T_b]^{-1} \{v\} = \lfloor N'(x) \rfloor \{v\}$$

$$\frac{d^2 v}{dx^2} = \lfloor H''(x) \rfloor [T_b]^{-1} \{v\} = \lfloor N''(x) \rfloor \{v\}$$

$$\frac{d^2 v}{dt^2} = \lfloor H(x) \rfloor [T_b]^{-1} \left\{ \frac{d^2 v}{dt^2} \right\} = \lfloor N(x) \rfloor \left\{ \frac{d^2 v}{dt^2} \right\}$$

Where:

$$\lfloor H'(x) \rfloor = \begin{bmatrix} 0 & 1 & 2x & 3x^2 \end{bmatrix}$$

$$\lfloor H''(x) \rfloor = \begin{bmatrix} 0 & 0 & 2 & 6x \end{bmatrix}$$

$$\lfloor N'(x)\rfloor^T=\{N'(x)\}=\begin{Bmatrix}\frac{-6x}{L^2}+\frac{6x^2}{L^3}\\ 1-\frac{4x}{L}+\frac{3x^2}{L^2}\\ \frac{6x}{L^2}-\frac{6x^2}{L^3}\\ -\frac{2x}{L^2}+\frac{3x^2}{L^3}\end{Bmatrix}$$

$$\lfloor N''(x)\rfloor^T=\{N''(x)\}=\begin{Bmatrix}\frac{-6}{L^2}+\frac{12x}{L^3}\\ \frac{-4}{L}+\frac{6x}{L^2}\\ \frac{6}{L^2}-\frac{12x}{L^3}\\ -\frac{2}{L^2}+\frac{6x}{L^3}\end{Bmatrix}$$

Using the results above into the Hamilton's principle, we get:

$$\int_{t_1}^{t_2}\delta\Pi\,dt=\{\delta v\}^T\int_{t_1}^{t_2}\int_0^L\begin{pmatrix}-\rho A\{N\}\lfloor N\rfloor\left\{\frac{\partial^2 v}{\partial t^2}\right\}-EI\{N''\}\lfloor N''\rfloor\{v\}\\ +f(x,t)\{N\}+p_1\{N\}\big|_{x=0}+p_2\{N\}\big|_{x=L}+m_1\{N'\}\big|_{x=0}+m_2\{N'\}\big|_{x=L}\end{pmatrix}dx\,dt=0$$

Note that:

$$\lfloor N\rfloor_{x=0}=\lfloor 1\quad 0\quad 0\quad 0\rfloor$$
$$\lfloor N'\rfloor_{x=0}=\lfloor 0\quad 1\quad 0\quad 0\rfloor$$
$$\lfloor N\rfloor_{x=L}=\lfloor 0\quad 0\quad 1\quad 0\rfloor$$
$$\lfloor N'\rfloor_{x=L}=\lfloor 0\quad 0\quad 0\quad 1\rfloor$$

Thus, the equation becomes:

$$\int_0^L\rho A\{N\}\lfloor N\rfloor dx\left\{\frac{d^2v}{dt^2}\right\}+EI\{N''\}\lfloor N''\rfloor dx\{v\}=\int_0^L f(x)\{N\}+\begin{Bmatrix}p_1\\ m_1\\ p_2\\ m_2\end{Bmatrix}$$

Or

$$[M]\{\ddot{v}\}+[K]\{v\}=\{q\}$$

Where [M] is called the mass matrix, [K] is called the stiffness matrix, and {q} is called the generalized force vector. Note that the generalized force vector includes the effect of the distributed loads as well as the concentrated ones. Also note that we can not write an explicit expression for the generalized force without knowing the force function. However, we may write:

$$\int_0^L f(x,t)\{N\} = \begin{Bmatrix} \int_0^L f(x,t)N_1\,dx \\ \int_0^L f(x,t)N_2\,dx \\ \int_0^L f(x,t)N_3\,dx \\ \int_0^L f(x,t)N_4\,dx \end{Bmatrix} = \begin{Bmatrix} f_1(t) \\ f_2(t) \\ f_3(t) \\ f_4(t) \end{Bmatrix}$$

Meanwhile, we may get the mass and stiffness matrices as:

$$[M] = \frac{\rho A L}{420}\begin{bmatrix} 156 & 22L & 54 & -13L \\ 22L & 4L^2 & 13L & -3L^2 \\ 54 & 13L & 156 & -22L \\ -13L & -3L^2 & -22L & 4L^2 \end{bmatrix}$$

$$[K] = \frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L \\ 6L & 4L^2 & -6L & 2L^2 \\ -12 & -6L & 12 & -6L \\ 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}$$

9.2.3 Assembling the Structure Equations

Let’s consider a simple beam structure that is divided into two elements as in Figure 9.2.1. We should consider two things:

1. The displacement and slope at the second node of the first element is exactly the same as those at the first node of the second element. This should apply to ensure the continuity of the displacement and slope from one element to the neighboring one.
2. The concentrated loads maybe divided among the elements at the connecting node, but, the summation of the divided parts should add up to the total externally applied concentrated load.

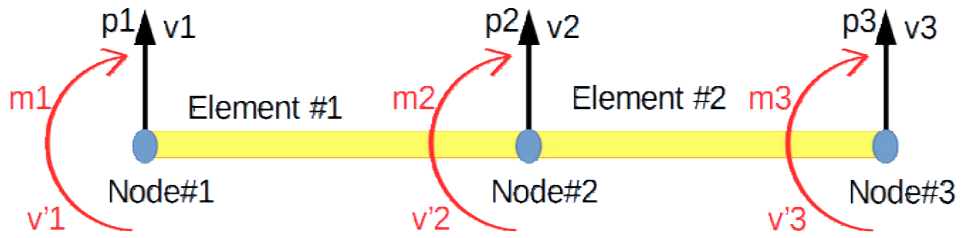

Figure 9.2.1. Two-Element beam structures.

We may write the element equation for each element to be:

$$[M^{(1)}]\{\ddot{v}\}^{(1)}+[K^{(1)}]\{v\}^{(1)}=\{q\}^{(1)}$$

$$[M^{(2)}]\{\ddot{v}\}^{(2)}+[K^{(2)}]\{v\}^{(2)}=\{q\}^{(2)}$$

Each of the above equations consists of four scalar equations. Now, if we want to model the whole structure, we will need to add the generalized forces to obtain the total external concentrated loads. Keeping in mind that the displacements at the connecting node is the same, we can get the equations for the assembled structure to be:

$$\frac{\rho A L}{420}\begin{bmatrix} 156 & 22L & 54 & -13L & 0 & 0 \\ 22L & 4L^2 & 13L & -3L^2 & 0 & 0 \\ 54 & 13L & 312 & 0 & 54 & -13L \\ -13L & -3L^2 & 0 & 8L^2 & 13L & -3L^2 \\ 0 & 0 & 54 & 13L & 156 & -22L \\ 0 & 0 & -13L & -3L^2 & -22L & 4L^2 \end{bmatrix}\begin{Bmatrix} \ddot{v}_1 \\ \ddot{v}'_1 \\ \ddot{v}_2 \\ \ddot{v}'_2 \\ \ddot{v}_3 \\ \ddot{v}'_3 \end{Bmatrix}$$

$$+\frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L & 0 & 0 \\ 6L & 4L^2 & -6L & 2L^2 & 0 & 0 \\ -12 & -6L & 24 & 0 & -12 & 6L \\ 6L & 2L^2 & 0 & 8L^2 & -6L & 2L^2 \\ 0 & 0 & -12 & -6L & 12 & -6L \\ 0 & 0 & 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}\begin{Bmatrix} v_1 \\ v'_1 \\ v_2 \\ v'_2 \\ v_3 \\ v'_3 \end{Bmatrix}=\begin{Bmatrix} f_1^{(1)} \\ f_2^{(1)} \\ f_3^{(1)}+f_1^{(2)} \\ f_4^{(1)}+f_2^{(2)} \\ f_3^{(2)} \\ f_4^{(2)} \end{Bmatrix}+\begin{Bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \\ p_3 \\ m_3 \end{Bmatrix}$$

Where we assumed that the element lengths, density, cross-section area, modulus of elasticity, and second moment of area are constant for both elements. Also, the superscripts (1) and (2) denote the information from the first and second element matrices respectively.

9.2.4 Applying the boundary conditions

Up to this moment, all the procedure described for the finite element model may apply to any beam with any number of elements. Differences will appear when different beams are subjected to different boundary conditions. For illustration purposes, we will consider two cases here.

Case of a cantilever beam:

For a cantilever beam fixed from the left side, we have:

$$v_1 = 0 \text{ and } v'_1 = 0 \rightarrow \ddot{v}_1 = 0 \text{ and } \ddot{v}'_1 = 0$$

Thus, the equation we obtained may be written as:

$$\frac{\rho A L}{420}\begin{bmatrix} 156 & 22L & 54 & -13L & 0 & 0 \\ 22L & 4L^2 & 13L & -3L^2 & 0 & 0 \\ 54 & 13L & 312 & 0 & 54 & -13L \\ -13L & -3L^2 & 0 & 8L^2 & 13L & -3L^2 \\ 0 & 0 & 54 & 13L & 156 & -22L \\ 0 & 0 & -13L & -3L^2 & -22L & 4L^2 \end{bmatrix}\begin{Bmatrix} 0 \\ 0 \\ \ddot{v}_2 \\ \ddot{v}'_2 \\ \ddot{v}_3 \\ \ddot{v}'_3 \end{Bmatrix}$$

$$+\frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L & 0 & 0 \\ 6L & 4L^2 & -6L & 2L^2 & 0 & 0 \\ -12 & -6L & 24 & 0 & -12 & 6L \\ 6L & 2L^2 & 0 & 8L^2 & -6L & 2L^2 \\ 0 & 0 & -12 & -6L & 12 & -6L \\ 0 & 0 & 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}\begin{Bmatrix} 0 \\ 0 \\ v_2 \\ v'_2 \\ v_3 \\ v'_3 \end{Bmatrix} = \begin{Bmatrix} f_1^{(1)} \\ f_2^{(1)} \\ f_3^{(1)} + f_1^{(2)} \\ f_4^{(1)} + f_2^{(2)} \\ f_3^{(2)} \\ f_4^{(2)} \end{Bmatrix} + \begin{Bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \\ p_3 \\ m_3 \end{Bmatrix}$$

Which may be separated into two distinct equations given as:

$$\frac{\rho A L}{420}\begin{bmatrix} 312 & 0 & 54 & -13L \\ 0 & 8L^2 & 13L & -3L^2 \\ 54 & 13L & 156 & -22L \\ -13L & -3L^2 & -22L & 4L^2 \end{bmatrix}\begin{Bmatrix} \ddot{v}_2 \\ \ddot{v}'_2 \\ \ddot{v}_3 \\ \ddot{v}'_3 \end{Bmatrix}$$

$$+\frac{EI}{L^3}\begin{bmatrix} 24 & 0 & -12 & 6L \\ 0 & 8L^2 & -6L & 2L^2 \\ -12 & -6L & 12 & -6L \\ 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}\begin{Bmatrix} v_2 \\ v'_2 \\ v_3 \\ v'_3 \end{Bmatrix} = \begin{Bmatrix} f_3^{(1)} + f_1^{(2)} \\ f_4^{(1)} + f_2^{(2)} \\ f_3^{(2)} \\ f_4^{(2)} \end{Bmatrix} + \begin{Bmatrix} p_2 \\ m_2 \\ p_3 \\ m_3 \end{Bmatrix}$$

Which is called the *primary equations*. The primary equations could be solved for the time response of the unknown displacements and slopes using any exat or approximate techniques (usually numerical). The secondary equation becomes:

$$\frac{\rho A L}{420}\begin{bmatrix} 54 & -13L & 0 & 0 \\ 13L & -3L^2 & 0 & 0 \end{bmatrix}\begin{Bmatrix} \ddot{v}_2 \\ \ddot{v}'_2 \\ \ddot{v}_3 \\ \ddot{v}'_3 \end{Bmatrix}$$

$$+\frac{EI}{L^3}\begin{bmatrix} -12 & 6L & 0 & 0 \\ -6L & 2L^2 & 0 & 0 \end{bmatrix}\begin{Bmatrix} v_2 \\ v'_2 \\ v_3 \\ v'_3 \end{Bmatrix}=\begin{Bmatrix} f_1^{(1)} \\ f_2^{(1)} \end{Bmatrix}+\begin{Bmatrix} p_1 \\ m_1 \end{Bmatrix}$$

Which is called the *secondary equations* or the *auxiliary equations*. The secondary equations have the unknown variables as the time history of the concentrated force and moment at the fixed end, thus, they are the support reactions which are found by substituting the solution obtained from the primary equations directly.

Case of a simply-supported beam:

For a simply-supported beam we have:

$v_1=0$ and $v_3=0$ → $\ddot{v}_1=0$ and $\ddot{v}_3=0$

Thus, the equation we obtained may be written as:

$$\frac{\rho A L}{420}\begin{bmatrix} 156 & 22L & 54 & -13L & 0 & 0 \\ 22L & 4L^2 & 13L & -3L^2 & 0 & 0 \\ 54 & 13L & 312 & 0 & 54 & -13L \\ -13L & -3L^2 & 0 & 8L^2 & 13L & -3L^2 \\ 0 & 0 & 54 & 13L & 156 & -22L \\ 0 & 0 & -13L & -3L^2 & -22L & 4L^2 \end{bmatrix}\begin{Bmatrix} 0 \\ \ddot{v}'_1 \\ \ddot{v}_2 \\ \ddot{v}'_2 \\ 0 \\ \ddot{v}'_3 \end{Bmatrix}$$

$$+\frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L & 0 & 0 \\ 6L & 4L^2 & -6L & 2L^2 & 0 & 0 \\ -12 & -6L & 24 & 0 & -12 & 6L \\ 6L & 2L^2 & 0 & 8L^2 & -6L & 2L^2 \\ 0 & 0 & -12 & -6L & 12 & -6L \\ 0 & 0 & 6L & 2L^2 & -6L & 4L^2 \end{bmatrix}\begin{Bmatrix} 0 \\ v'_1 \\ v_2 \\ v'_2 \\ 0 \\ v'_3 \end{Bmatrix}=\begin{Bmatrix} f_1^{(1)} \\ f_2^{(1)} \\ f_3^{(1)}+f_1^{(2)} \\ f_4^{(1)}+f_2^{(2)} \\ f_3^{(2)} \\ f_4^{(2)} \end{Bmatrix}+\begin{Bmatrix} p_1 \\ m_1 \\ p_2 \\ m_2 \\ p_3 \\ m_3 \end{Bmatrix}$$

which gives the primary equations as:

$$\frac{\rho A L}{420}\begin{bmatrix} 4L^2 & 13L & -3L^2 & 0 \\ 13L & 312 & 0 & -13L \\ -3L^2 & 0 & 8L^2 & -3L^2 \\ 0 & -13L & -3L^2 & 4L^2 \end{bmatrix}\begin{Bmatrix} \ddot{v}'_1 \\ \ddot{v}_2 \\ \ddot{v}'_2 \\ \ddot{v}'_3 \end{Bmatrix}$$

$$+\frac{EI}{L^3}\begin{bmatrix} 4L^2 & -6L & 2L^2 & 0 \\ -6L & 24 & 0 & 6L \\ 2L^2 & 0 & 8L^2 & 2L^2 \\ 0 & 6L & 2L^2 & 4L^2 \end{bmatrix}\begin{Bmatrix} v'_1 \\ v_2 \\ v'_2 \\ v'_3 \end{Bmatrix}=\begin{Bmatrix} f_2^{(1)} \\ f_3^{(1)}+f_1^{(2)} \\ f_4^{(1)}+f_2^{(2)} \\ f_4^{(2)} \end{Bmatrix}+\begin{Bmatrix} m_1 \\ p_2 \\ m_2 \\ m_3 \end{Bmatrix}$$

And the secondary equations as:

$$\frac{\rho A L}{420}\begin{bmatrix} 22L & 54 & -13L & 0 \\ 0 & 54 & 13L & -22L \end{bmatrix}\begin{Bmatrix} \ddot{v}'_1 \\ \ddot{v}_2 \\ \ddot{v}'_2 \\ \ddot{v}'_3 \end{Bmatrix}$$

$$+\frac{EI}{L^3}\begin{bmatrix} 6L & -12 & 6L & 0 \\ 0 & -12 & -6L & -6L \end{bmatrix}\begin{Bmatrix} v'_1 \\ v_2 \\ v'_2 \\ v'_3 \end{Bmatrix}=\begin{Bmatrix} f_1^{(1)} \\ f_3^{(2)} \end{Bmatrix}+\begin{Bmatrix} p_1 \\ p_3 \end{Bmatrix}$$

Which may be used to evaluate the support reactions at both ends.

9.2.5 Natural Frequencies and Frequency Response

In the previous section we obtained the equations of motion of the beam using the finnite element method in the form:

$$[M]\{\ddot{v}\}+[K]\{v\}=\{q\}$$

Which is the form we have for a MDOF system. From this form we may calculate the natural frequencies by solving the Eigenvalue problem:

$$\left|-\omega^2[M]+[K]\right|=0$$

Also we may obtain obtain the response to harmonic excitation using the dynamic stiffness matrix which may be written in the form:

$$\{v\}_{Amplitude}=\left(-\omega^2[M]+[K]\right)^{-1}\{q\}_{Amplitude}$$

The code in section 13.9 evaluates the natural frequencies and frequency response of a beam. The boundary conditions set are for a simply supported beam.

We get the frequency response shown as

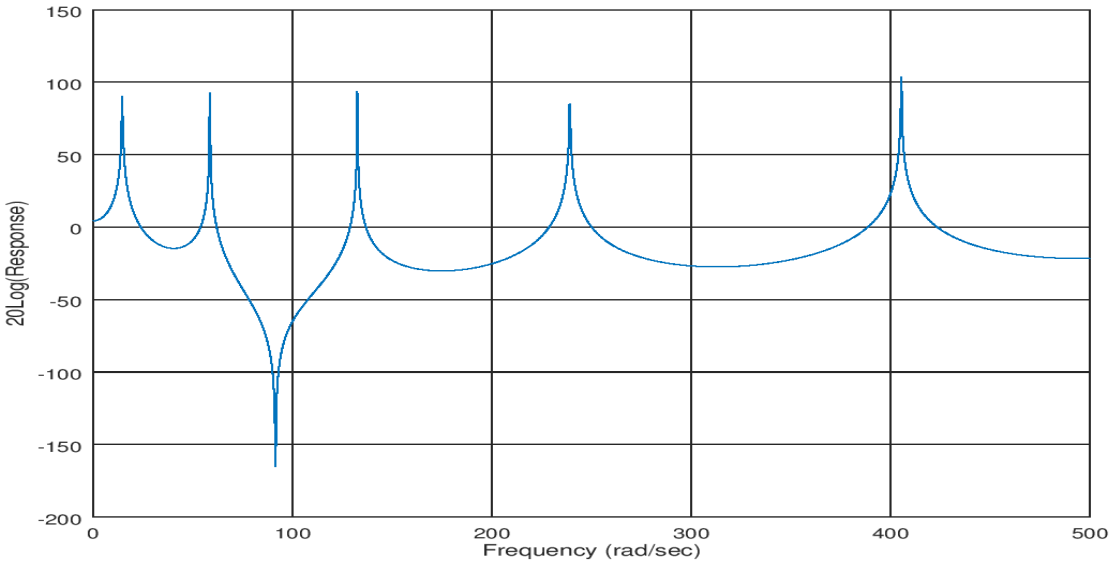

Figure 9.2.2: Beam frequency response function (Log scale)

9.2.6 The Effect of Axial Loads

For the special case of a beam with constant cross-section, we may write the homogeneous form of the PDE as:

$$\rho A \ddot{v} + EI \frac{\partial^4 v}{\partial x^4} + P \frac{\partial^2 v}{\partial x^2} = 0$$

If we go through the finite element model derivation, we will get the equation of motion as:

$$[M]\{\ddot{v}\} + ([K] - P[K_G])\{v\} = \{q\}$$

Where [KG] is called the *geometric stiffness matrix* and one the element level, it will be:

$$[K_G] = \frac{1}{30L}\begin{bmatrix} 36 & 3L & -36 & 3L \\ 3L & 4L^2 & -3L & -L^2 \\ -36 & -3L & 36 & -3L \\ 3L & -L^2 & -3L & 4L^2 \end{bmatrix}$$

We may use the static equation (removing the acceleration term) to evaluate the buckling load for a beam. However, since we are concerned with the dynamic problem, let's examine the effect of the axial load on the natural frequencies of the beam. To perform this task, we will add the geometric stiffness matrix to the equation of motion and add a loop to change the values of the axial load and evaluate the minimum natural frequency. The loop will have the line listed below

```
%Evaluating the Natural Frequencies
for ii=0:100
   PP=ii*0.015;
   PPP(ii+1)=PP;
   vvv(ii+1)=min(sort(sqrt(eig(inv(MGlobal)*(KGlobal-PP*KGGlobal)))));
endfor
```

The result is plotted in the graph of Figure 9.2.3. Notice that the natural frequency with no load is the first point on the graph. Also, notice that the natural frequency reduces with the increase of the axial load until it reaches zero as the load reached the buckling load.

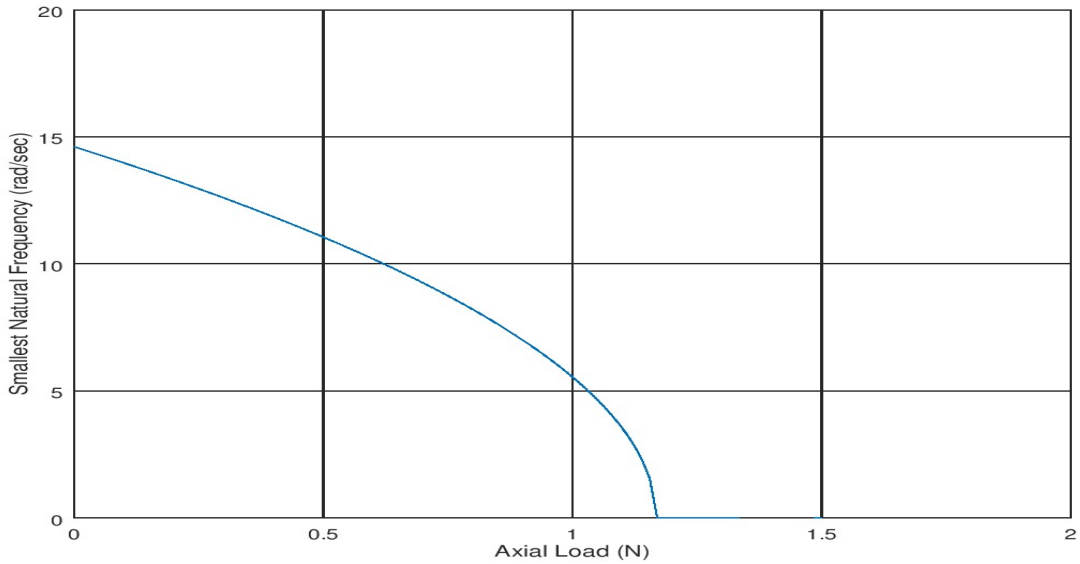

Figure 9.2.3: Change of minimum natural frequency of a simply-supported beam with axial load

9.2.7 The Fire-Hose Problem

A very interesting problem in structure dynamics is that of a pipe conveying fluid. The dynamics of such a problem describes some important problems such as that of petroleum pipes. As the fluid flows inside the pipe, it creates thrust against the pipe, much like the axial force, this reduces the bending stiffness of the pipe and may cause buckling. Also, another phenomenon may appear in such problems, namely *flutter*. Flutter is defines as self excited vibration of a structure. This phenomenon can be seen clearly in cases where the fluid is traveling very fast inside a compliant (soft) pipe such as the fire-hose. In such cases, you may notice that the fire-hose starts moving around without any external excitation vibrating as the body of a snake.

To model this problem, we will use Hamilton's principle to derive the equations of motion, then use the reduced form to create a finite element model to investigate the dynamic performance under different conditions. The potential energy in our problem will involve the elastic energy as well as the gravitational energy such that:

$$U=\int_0^L\left(\frac{1}{2}EI\left(\frac{\partial^2 v}{\partial x^2}\right)^2+(\rho_p A_p+\rho_f A_f)gv\right)dx$$

Where the subscript p denoted the pipe properties while the subscript f denotes the fluid. Obtaining the variation, we gt:

$$\delta U=\int_0^L\left(EI\left(\frac{\partial^2 v}{\partial x^2}\right)\delta\left(\frac{\partial^2 v}{\partial x^2}\right)+(\rho_p A_p+\rho_f A_f)g\,\delta v\right)dx$$

Integrating by parts to get the boundary terms:

$$\delta U=EI\left(\frac{\partial^2 v}{\partial x^2}\right)\delta\left(\frac{\partial v}{\partial x}\right)\Bigg|_0^L-\frac{\partial}{\partial x}\left(EI\left(\frac{\partial^2 v}{\partial x^2}\right)\right)\delta v\Bigg|_0^L+\int_0^L\left(EI\left(\frac{\partial^4 v}{\partial x^4}\right)\delta v+(\rho_p A_p+\rho_f A_f)g\,\delta v\right)dx$$

The kinetic energy will be:

$$T=\frac{1}{2}\int_0^L\left(\rho_p A_p\dot{v}^2+\rho_f A_f\left(V_f\frac{\partial v}{\partial x}+\dot{v}\right)^2+\rho_f A_f V_f^2\right)dx$$

Obtaining the variation, we get:

$$\delta T=\int_0^L\left((\rho_p A_p+\rho_f A_f)\dot{v}\,\delta\dot{v}+\rho_f A_f\left(V_f^2\frac{\partial v}{\partial x}\delta\left(\frac{\partial v}{\partial x}\right)+V_f\dot{v}\,\delta\left(\frac{\partial v}{\partial x}\right)+V_f\frac{\partial v}{\partial x}\delta\dot{v}\right)\right)dx$$

Performing integration by parts over x, we get:

$$\delta T = \rho_f A_f \left(V_f^2 \frac{\partial v}{\partial x} \delta v + V_f \dot{v} \delta v \right) \Big|_0^L$$
$$+ \int_0^L \left((\rho_p A_p + \rho_f A_f) \dot{v} \delta \dot{v} + \rho_f A_f \left(-V_f^2 \frac{\partial^2 v}{\partial x^2} \delta v - V_f \frac{\partial \dot{v}}{\partial x} \delta v + V_f \frac{\partial v}{\partial x} \delta \dot{v} \right) \right) dx$$

If we ignore writing the boundary terms explicitly, we get the total energy after performing the integration by part as:

$$\int_{t_1}^{t_2} \delta \Pi \, dt = \int_{t_1}^{t_2} \left(\int_0^L \left(\begin{array}{c} -(\rho_p A_p + \rho_f A_f) \ddot{v} - \rho_f A_f \left(V_f^2 \frac{\partial^2 v}{\partial x^2} + 2 V_f \frac{\partial \dot{v}}{\partial x} \right) \\ - EI \frac{\partial^4 v}{\partial x^4} - (\rho_p A_p + \rho_f A_f) g \end{array} \right) \delta v \, dx \right) dt = 0$$

From which we can get the equation of motion to be:

$$(\rho_p A_p + \rho_f A_f) \ddot{v} + EI \frac{\partial^4 v}{\partial x^4} + \rho_f A_f V_f^2 \frac{\partial^2 v}{\partial x^2} + 2 \rho_f A_f V_f \frac{\partial \dot{v}}{\partial x} = -(\rho_p A_p + \rho_f A_f) g$$

The first two terms of the above equation are the terms that we may get in case the fluid is not flowing inside the pipe. The third term describes the effect of the *thrust* created by the fluid flow similar to the term we had for axial loading of columns, while the last term describes the effect of the *Coriolis acceleration* due to the motion of the fluid inside a pipe that is also moving. On the right-hand side, the effect of the weight of the pipe and fluid appears as a static constant loading.

Obtaining the finite element model for the above problem, the equation of motion will be:

$$[M]\{\ddot{v}\} + ([K] - \rho_f A_f V_f^2 [K_G])\{v\} + 2 \rho_f A_f V_f [C]\{\dot{v}\} = \{f_g\}$$

In the above equation, the mass matrix will include the mass of the fluid as well as the pipe, while the stiffness and geometric stiffness matrices will not change. The Coriolis matrix will be:

$$[C] = \int_0^L \{N\} \lfloor N' \rfloor dx$$

Which will be:

$$[C] = \frac{1}{60} \begin{bmatrix} -30 & 6L & 30 & -6L \\ -6L & 0 & 6L & -L^2 \\ -30 & -6L & -30 & 6L \\ 6L & L^2 & -6L & 0 \end{bmatrix}$$

While the gravitational load vector will be given by:

$$\{f_g\} = -(\rho_p A_p + \rho_f A_f) g \int_0^L \{N\} dx$$

Which will be:

$$\{f_g\} = -\frac{(\rho_p A_p + \rho_f A_f) Lg}{12} \begin{Bmatrix} 6 \\ L \\ 6 \\ -L \end{Bmatrix}$$

A typical plot of the change of the natural frequency with the fluid speed is shown in Figure 9.2.4. As you may notice, the first natural frequency reaches zero at a certain velocity, which indicate buckling occurring in the beam due to the fluid thrust. At a higher speed, two natural frequencies come to be equal which results in the dynamic instability (flutter). The results below are plotted for a simply supported pipe made of aluminum and conveying water. The pipe length is 2 m, internal radius 10 mm, wall thickness 1 mm. Water density is 1000 kg/m3, Aluminum density is 2700 kg/m3, modulus of elasticity 71 GPa, and 10 elements were used to run the simulation.

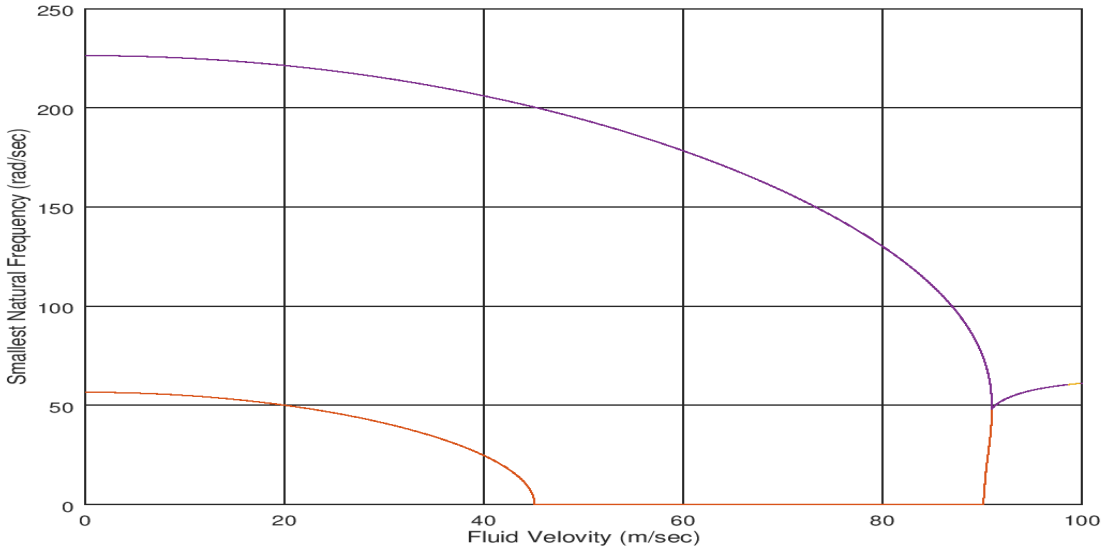

Figure 9.2.4: Typical change of natural frequency with flow speed

9.3 Post-Buckling Deflection Relations

For detailed derivation of the problem and its solution, you may refer to the lecture notes titled *Intermediate Mechanics of Materials*. To derive the element equations for the finite element method, we will use the *reduced form*. we may recall the reduced form to be:

$$W_{ex}=\int_0^L (f(x)v+g(x)u)\,dx-Pu\big|_{x=L}+\frac{1}{2}\int_0^L P\left(\frac{dv}{dx}\right)^2 dx$$

Obtaining the variation of the external work and applying integration by parts, we get:

$$\delta W_{ex}=\int_0^L (f(x)\delta v+g(x)\delta u)\,dx-P\,\delta u\big|_{x=L}+P\left(\frac{dv}{dx}\right)\delta v\Big|_0^L-\int_0^L P\left(\frac{d^2v}{dx^2}\right)\delta v\,dx$$

For the elastic energy, we get:

$$U=\frac{1}{2}\int_{Volume}\sigma\,\epsilon\,dV=\frac{1}{2}\int_{Volume}E\,\epsilon^2\,dV$$

But, in this case, we have we will need to involve the nonlinear strain relation. **The von Karman large deflection strain relation** that states:

$$\epsilon=\frac{du}{dx}-z\frac{d^2v}{dx^2}+\frac{1}{2}\left(\frac{dv}{dx}\right)^2$$

Which results in the expression of the elastic energy in the form:

$$U=\frac{1}{2}\int_{Volume}E\left(\frac{du}{dx}-z\frac{d^2v}{dx^2}+\frac{1}{2}\left(\frac{dv}{dx}\right)^2\right)^2 dV$$

Obtaining the variation, we get:

$$\delta U=\int_{Volume}E\left(\frac{du}{dx}-z\frac{d^2v}{dx^2}+\frac{1}{2}\left(\frac{dv}{dx}\right)^2\right)\delta\left(\frac{du}{dx}-z\frac{d^2v}{dx^2}+\frac{1}{2}\left(\frac{dv}{dx}\right)^2\right)dV$$

or

$$\delta U=\int_{Volume}E\left(\frac{du}{dx}-z\frac{d^2v}{dx^2}+\frac{1}{2}\left(\frac{dv}{dx}\right)^2\right)\left(\delta\frac{du}{dx}-z\,\delta\frac{d^2v}{dx^2}+\left(\frac{dv}{dx}\right)\delta\left(\frac{dv}{dx}\right)\right)dV$$

Now, let's separate the linear and nonlinear terms such that:

$$\delta U=\delta U_L+\delta U_N$$

Where:

$$\delta U_L=\int_{Volume}E\left(\frac{du}{dx}\delta\frac{du}{dx}-z\frac{du}{dx}\delta\frac{d^2v}{dx^2}-z\frac{d^2v}{dx^2}\delta\frac{du}{dx}+z^2\frac{d^2v}{dx^2}\delta\frac{d^2v}{dx^2}\right)dV$$

Is the same expression we got for the linear problem in section 9.1. Meanwhile, the nonlinear term is:

$$\delta U_N = \int_{Volume} E\left(\frac{1}{2}\left(\frac{dv}{dx}\right)^2 \delta\frac{du}{dx} + z\frac{1}{2}\left(\frac{dv}{dx}\right)^2 \delta\frac{d^2v}{dx^2} + \frac{1}{2}\left(\frac{dv}{dx}\right)^3 \delta\frac{dv}{dx} + \frac{du}{dx}\frac{dv}{dx}\delta\frac{dv}{dx} - z\frac{d^2v}{dx^2}\frac{dv}{dx}\delta\frac{dv}{dx}\right)dV$$

In the above relation, if the modulus of elasticity is constant over any cross section and the z is measured from the centroid, the second and fifth terms will result in zero, and the relation reduces to:

$$\delta U_N = \int_0^L EA\left(\frac{1}{2}\left(\frac{dv}{dx}\right)^2 \delta\frac{du}{dx} + \frac{1}{2}\left(\frac{dv}{dx}\right)^3 \delta\frac{dv}{dx} + \frac{du}{dx}\frac{dv}{dx}\delta\frac{dv}{dx}\right)dx$$

All the linear terms have been dealt with in before, thus, here we will work with the nonlinear terms. To simplify the problem, we will assume that the axial displacements are equal to zero (Refer to the lecture notes on *Intermediate Mechanics of Materials* for details) which will reduce the nonlinear potential energy term to:

$$\delta U_N = \int_0^L EA\left(\frac{1}{2}\left(\frac{dv}{dx}\right)^3 \delta\frac{dv}{dx}\right)dx$$

Which may be rewritten as:

$$\delta U_N = \frac{1}{2}EA\int_0^L \delta\frac{dv}{dx}\left(\frac{dv}{dx}\frac{dv}{dx}\right)\frac{dv}{dx}dx$$

Using the beam interpolation function, we get:

$$\delta U_N = \lfloor \delta v \rfloor \frac{1}{2}EA\int_0^L \{N'\}\left(\lfloor v \rfloor \{N'\}\lfloor N' \rfloor \{v\}\right)\lfloor N' \rfloor dx\{v\} = \lfloor \delta v \rfloor [N_2]\{v\}$$

Where [N2] is a second order nonlinear matrix in the nodal displacements. Evaluating this matrix analytically is not impossible, rather, it will produce a cumbersome expression that can not be easily implemented in the code, thus, the nonlinear matrix is usually evaluated using numerical integration. Adding this term to the element equation, we get:

$$\left([K] - P[K_G] + [N_2]\right)\{v\} = \{0\}$$

Note that for all load values less than the buckling load, the [N2] matrix will be zero because the beam has no deflection. For the axial load values greater than the buckling load, the nonlinear term will prevent the deflections from blowing up to infinity and as the load increases, the nonlinear term will grow larger (stronger). To solve the above equation, we follow the following algorithm:

1. Select the value of P greater than the buckling load

2. Evaluate the linear stiffness term: $[K_L]=[K]-P[K_G]$
3. Introduce the function $\Psi(v)=([K_L]+[N_2])\{v\}$
4. Assume any initial guess for the displacement vector {v}
5. For the function $\Psi(v)=([K_L]+[N_2])\{v\}$ to be equal to zero, we will use Newton-Raphson iterations to find the value of {v} that maintains the function equal to zero.
6. Increment the value of P until you reach the maximum value then go to step 2

For step 5 in the above algorithm we will use Newton-Raphson method which follows the following algorithm:

1. Using an initial guess of the displacement vector, evaluate the function $\Psi(v_k)=([K_L]+[N_2]_k)\{v_k\}$ and its derivative $\frac{d\Psi}{d\{v_k\}}=[K_L]+3[N_2]_k$
2. Evaluate the approximate solution for the displacement vector using the relation: $\{v_{k+1}\}=\{v_k\}-\left[\frac{d\Psi}{d\{v_k\}}\right]^{-1}\{\Psi(v_k)\}$
3. If the result is *almost* equal to the initial vector, then exit, otherwise use the new value of {v} as the initial value and go to step 1

A code was developed using Octave package to calculate the maximum deflection of the beam at different temperatures. The results for a fixed-fixed Aluminum beam with modulus of elasticity of 71 GPa, length 0.5 m, width 0.02 m, thickness 0.002 m, and thermal expansion coefficient $22.5*10^{-6}$. The results using 10 elements are presented in Figure 9.3.1.

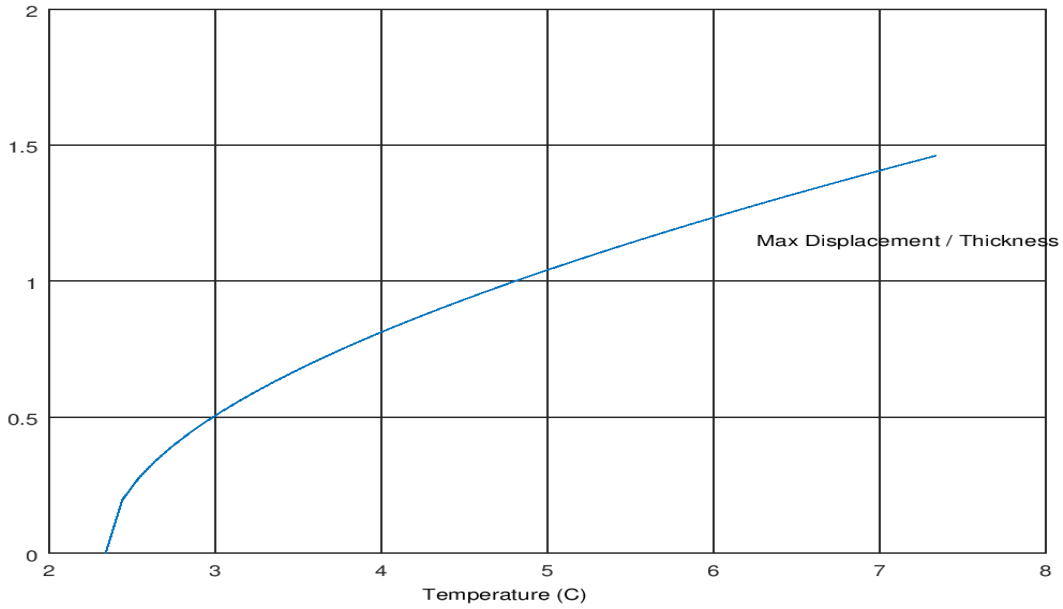

Figure 9.3.1: Post-buckling Deflection of Fixed-Fixed beam

10. In-plane Loading of a Plate

(Code for this section may be downloaded from GitHub through this link)

Now, we have enough background to extend our study to cover the plain elasticity problem. In this problem we are only concerned with the thin structures, such as thin plates, that are subjected to in-plane loading. In such a problem, the strain components, we are concerned with, become the axial strains in the plane of the plate and the shear strain component associated with them. All variables are assumed to constant across the thickness.

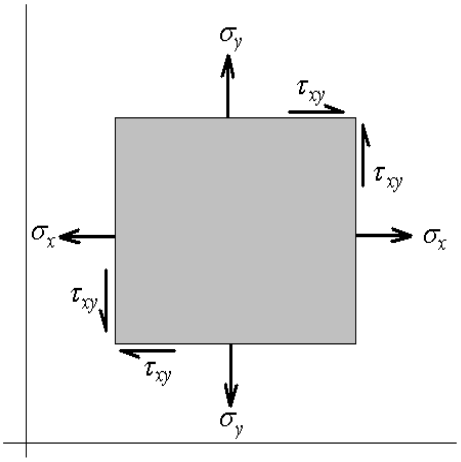

Figure 10.1: A sketch presenting a plain element with the stresses applied on it.

The above described stresses and strain are related through the following relations

$$\sigma_x = D\varepsilon_x + \nu D\varepsilon_y$$
$$\sigma_y = \nu D\varepsilon_x + D\varepsilon_y$$
$$\tau_{xy} = 2G\varepsilon_{xy}$$

Where

$$D = \frac{E}{1-\nu^2}$$
$$G = \frac{E}{2(1+\nu)}$$

In matrix form

$$\begin{Bmatrix} \sigma_x \\ \sigma_y \\ \tau_{xy} \end{Bmatrix} = \begin{bmatrix} D & \nu D & 0 \\ \nu D & D & 0 \\ 0 & 0 & 2G \end{bmatrix} \begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_{xy} \end{Bmatrix}$$

Or

$$\{\sigma\} = [Q]\{\varepsilon\}$$

10.1 Strain-Displacement Relations

The strain displacement relation in the 2-D problem is slightly different taking into account the displacement in the y-direction as well

$$\varepsilon_x = \frac{du}{dx}$$

$$\varepsilon_y = \frac{dv}{dy}$$

$$\varepsilon_{xy} = \frac{1}{2}\left(\frac{du}{dy} + \frac{dv}{dx}\right)$$

Or, in matrix form

$$\begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \varepsilon_{xy} \end{Bmatrix} = \begin{Bmatrix} \dfrac{du}{dx} \\ \dfrac{dv}{dy} \\ \dfrac{1}{2}\left(\dfrac{du}{dy} + \dfrac{dv}{dx}\right) \end{Bmatrix}$$

10.2 Strain Energy

The strain energy should take all stresses and strains into account. Thus, we get the expression as

$$U = \int_{Volume} \frac{1}{2}\varepsilon\sigma dV = \int_{Volume} \frac{1}{2}\{\varepsilon\}^T[Q]\{\varepsilon\}dV$$

For constant thickness, and since all the variables are constant across the thickness, we may simplify the integral over the volume to become an integral over the area

$$U = h\int_{Area} \frac{1}{2}\{\varepsilon\}^T[Q]\{\varepsilon\}dA$$

For the approximation of the displacement function u(x,y) over the element, use the 2-D interpolation function described in section 6.1. In the 2-D elasticity problem, we displacements in both the x and y-directions at every point of the plate. For a rectangular element, you get 8 DOF per element. The displacement "***vector***" may be written as:

$$\begin{Bmatrix} u(x,y) \\ v(x,y) \end{Bmatrix} = \begin{bmatrix} N_1, N_2, N_3, N_4 & 0,0,0,0 \\ 0,0,0,0 & N_1, N_2, N_3, N_4 \end{bmatrix} \begin{Bmatrix} u_1 \\ \vdots \\ u_4 \\ v_1 \\ \vdots \\ v_4 \end{Bmatrix}$$

And the strain-displacement relations become:

$$\begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{Bmatrix} = \begin{Bmatrix} \dfrac{du}{dx} \\ \dfrac{dv}{dy} \\ \dfrac{du}{dy} + \dfrac{dv}{dx} \end{Bmatrix} = \begin{bmatrix} N_{1x}, N_{2x}, N_{3x}, N_{4x} & 0,0,0,0 \\ 0,0,0,0 & N_{1y}, N_{2y}, N_{3y}, N_{4y} \\ N_{1y}, N_{2y}, N_{3y}, N_{4y} & N_{1x}, N_{2x}, N_{3x}, N_{4x} \end{bmatrix} \begin{Bmatrix} u_1 \\ \vdots \\ u_4 \\ v_1 \\ \vdots \\ v_4 \end{Bmatrix}$$

or

$$\{\varepsilon_m\} = [B_m]\{w_m\}$$

Note: The subscript m is intended to differentiate the membrane (in-plane) strain/deflection as contrasted to the bending (out-of-plane) strain/deflection discussed in the plate bending problems.

Which gives the strain energy:

$$U = h \int_{Area} \frac{1}{2} \{\varepsilon\}^T [Q] \{\varepsilon\} dA$$

Or:

$$U = h \int_{Area} \frac{1}{2} \{w_m\}^T [B_m]^T [Q][B_m] \{w_m\} dA$$

Taking the variation of the stain energy:

$$\delta U = h \int_{Area} \{\delta w_m\}^T [B_m]^T [Q][B_m] \{w_m\} dA = \{\delta w_m\}^T [k_m] \{w_m\}$$

The stiffness matrix may be evaluated analytically using Mathematica to get:

$$k_m = \frac{h}{48\,a\,b} \begin{pmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{pmatrix}$$

Where:

$$k_{11}=\begin{pmatrix} 16b^2D+8a^2G & -16b^2D+4a^2G & -8b^2D-4a^2G & 8b^2D-8a^2G \\ -16b^2D+4a^2G & 16b^2D+8a^2G & 8b^2D-8a^2G & -8b^2D-4a^2G \\ -8b^2D-4a^2G & 8b^2D-8a^2G & 16b^2D+8a^2G & -16b^2D+4a^2G \\ 8b^2D-8a^2G & -8b^2D-4a^2G & -16b^2D+4a^2G & 16b^2D+8a^2G \end{pmatrix}=k_{22}$$

Let's define the new variables:

$R=12Dv+6G$ and $J=12Dv-6G$

And we can write:

$$k_{12}=ab\begin{pmatrix} R & J & -R & -J \\ -J & -R & J & R \\ -R & -J & R & J \\ J & R & -J & -R \end{pmatrix}=k_{21}^T$$

The Maxima code used to derive the above stiffness matrix is listed in section 12.2. The Octave code that evaluates the matrix using numerical integration is described in section 13.10.

10.3 External Work

$$W_{external}=\int_{Area}\begin{Bmatrix} u \\ v \end{Bmatrix}^T\begin{Bmatrix} f_x \\ f_y \end{Bmatrix}dA+\sum_i\begin{Bmatrix} u(x_i,y_i) \\ v(x_i,y_i) \end{Bmatrix}^T\begin{Bmatrix} P_{xi} \\ P_{yi} \end{Bmatrix}$$

Where f_x, f_y are the distributed forces, in the x and y-directions respectively, applied on the surface and boundaries of the structure, while P_{xi}, P_{yi} are the externally applies concentrated forces, in the x and y-directions respectively, applied at points (x_i, y_i) on the surface or boundaries of the structure.

For most practical problems, the externally applied loads, in a plane elasticity problem, are applied on the boundaries. Using the interpolation relations we developed earlier, we get:

$$W_{external}=\int_{Area}\{w_m\}^T[N_m]^T\{f\}dA+\sum_i\{w_m\}^T[N_m(x_i,y_i)]^T\{P\}$$

Applying the variation, we get:

$$\delta W_{external}=\delta\{w_m\}^T\left(\int_{Area}[N_m]^T\{f\}dA+\sum_i[N_m(x_i,y_i)]^T\{P\}\right)$$

10.4 Element Equations

Using the Hamilton's principle, we get:

$$\delta \Pi = \delta W_{ex} - \delta U = 0$$

$$[k_m]\{w_m\} = \int_{Area} [N_m]^T \{f\} dA + \sum_i [N_m(x_i, y_i)]^T \{P_i\}$$

$$[k_m]\{w_m\} = \{f_m\} + \{P_m\}$$

10.5 Examples

10.5.1 Single element with corner-applied load

Let's consider a rectangular plate fixed at two corners on the same side.

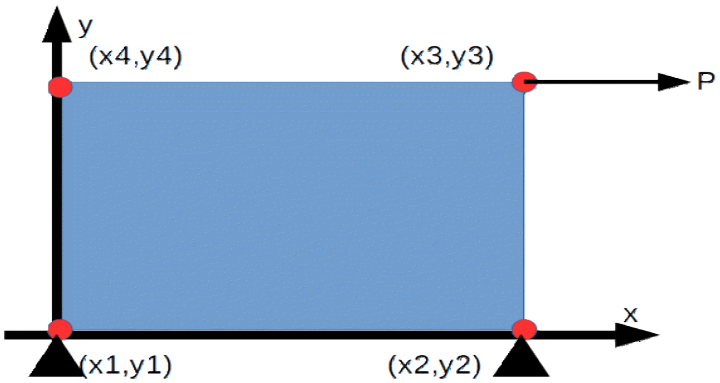

Figure 10.5.1. Plate with corner load

For generality, we may present the stiffness matrix as:

$$[k_m] = \begin{bmatrix} k_{11} & k_{12} & k_{13} & k_{14} & k_{15} & k_{16} & k_{17} & k_{18} \\ k_{12} & k_{22} & k_{23} & k_{24} & k_{25} & k_{26} & k_{27} & k_{28} \\ k_{13} & k_{23} & k_{33} & k_{34} & k_{35} & k_{36} & k_{37} & k_{38} \\ k_{14} & k_{24} & k_{34} & k_{44} & k_{45} & k_{46} & k_{47} & k_{48} \\ k_{15} & k_{25} & k_{35} & k_{45} & k_{55} & k_{56} & k_{57} & k_{58} \\ k_{16} & k_{26} & k_{36} & k_{46} & k_{56} & k_{66} & k_{67} & k_{68} \\ k_{17} & k_{27} & k_{37} & k_{47} & k_{75} & k_{76} & k_{77} & k_{78} \\ k_{18} & k_{28} & k_{38} & k_{48} & k_{58} & k_{68} & k_{78} & k_{88} \end{bmatrix}$$

But the node numbering on the elements level is different from that in the global level! (revise section 6). Thus, the stiffness matrix in the global coordinates with simply be created by swapping the 3rd and 4th, and the 7th and 8th rows and columns receptively to obtain the matrix below:

$$[k_m]_{Golbal}=\begin{bmatrix} k_{11} & k_{12} & k_{14} & k_{13} & k_{15} & k_{16} & k_{18} & k_{17} \\ k_{12} & k_{22} & k_{24} & k_{23} & k_{25} & k_{26} & k_{28} & k_{27} \\ k_{14} & k_{24} & k_{44} & k_{34} & k_{45} & k_{46} & k_{48} & k_{47} \\ k_{13} & k_{23} & k_{34} & k_{33} & k_{35} & k_{36} & k_{38} & k_{37} \\ k_{15} & k_{25} & k_{45} & k_{35} & k_{55} & k_{56} & k_{58} & k_{57} \\ k_{16} & k_{26} & k_{46} & k_{36} & k_{56} & k_{66} & k_{68} & k_{67} \\ k_{18} & k_{28} & k_{48} & k_{38} & k_{58} & k_{68} & k_{88} & k_{78} \\ k_{17} & k_{27} & k_{47} & k_{37} & k_{75} & k_{76} & k_{78} & k_{77} \end{bmatrix}$$

The load vector may be evaluated using the relation:

$$\{P\}=\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T \begin{Bmatrix} P \\ 0 \end{Bmatrix}=\begin{Bmatrix} 0 \\ 0 \\ 0 \\ P \\ 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix}$$

From the boundary conditions, we may conclude that:

$$u_1=u_2=v_1=v_2=0$$

Thus, the 1st , 2nd, 5th, and 6th columns of the stiffness matrix will be multiplied by zeroes, while the corresponding rows will create the auxiliary equations, thus, we may write the equations as:

The full equation:

$$\begin{bmatrix} k_{11} & k_{12} & k_{14} & k_{13} & k_{15} & k_{16} & k_{18} & k_{17} \\ k_{12} & k_{22} & k_{24} & k_{23} & k_{25} & k_{26} & k_{28} & k_{27} \\ k_{14} & k_{24} & k_{44} & k_{34} & k_{45} & k_{46} & k_{48} & k_{47} \\ k_{13} & k_{23} & k_{43} & k_{33} & k_{35} & k_{36} & k_{38} & k_{37} \\ k_{15} & k_{25} & k_{45} & k_{35} & k_{55} & k_{56} & k_{58} & k_{57} \\ k_{16} & k_{26} & k_{46} & k_{36} & k_{56} & k_{66} & k_{68} & k_{67} \\ k_{18} & k_{28} & k_{48} & k_{38} & k_{58} & k_{68} & k_{88} & k_{78} \\ k_{17} & k_{27} & k_{47} & k_{37} & k_{75} & k_{76} & k_{78} & k_{77} \end{bmatrix} \begin{Bmatrix} 0 \\ 0 \\ u_3 \\ u_4 \\ 0 \\ 0 \\ v_3 \\ v_4 \end{Bmatrix}=\begin{Bmatrix} 0 \\ 0 \\ 0 \\ P \\ 0 \\ 0 \\ 0 \\ 0 \end{Bmatrix}$$

The reduced equation:

$$\begin{bmatrix} k_{44} & k_{34} & k_{48} & k_{47} \\ k_{34} & k_{33} & k_{38} & k_{37} \\ k_{48} & k_{38} & k_{88} & k_{78} \\ k_{47} & k_{37} & k_{78} & k_{77} \end{bmatrix} \begin{Bmatrix} u_3 \\ u_4 \\ v_3 \\ v_4 \end{Bmatrix} = \begin{Bmatrix} 0 \\ P \\ 0 \\ 0 \end{Bmatrix}$$

The Auxiliary equation

$$\begin{bmatrix} k_{14} & k_{13} & k_{18} & k_{17} \\ k_{24} & k_{23} & k_{28} & k_{27} \\ k_{45} & k_{35} & k_{58} & k_{57} \\ k_{46} & k_{36} & k_{68} & k_{67} \end{bmatrix} \begin{Bmatrix} u_3 \\ u_4 \\ v_3 \\ v_4 \end{Bmatrix} = \begin{Bmatrix} R_{x1} \\ R_{x2} \\ R_{y1} \\ R_{y2} \end{Bmatrix}$$

If we use numbers (see section 13.10 for the program code), lets an Aluminum square plate with thickness of 0.5 mm and side length 1 m. The data as per the code will be:

Applyin load of 10 N, the deflections of the

$$\begin{Bmatrix} u_3 \\ u_4 \\ v_3 \\ v_4 \end{Bmatrix} = \begin{Bmatrix} 1.92 \\ 2.32 \\ 0.60 \\ -0.71 \end{Bmatrix} \mu m$$

While the reactions are:

$$\begin{Bmatrix} R_{x1} \\ R_{x2} \\ R_{y1} \\ R_{y2} \end{Bmatrix} = \begin{Bmatrix} -6.99 \\ -3.01 \\ -10 \\ 10 \end{Bmatrix} N$$

10.6 Thermal Strain

In mechanics of materials, the thermal strain is expressed as linear relation in the *change* in temperature of the material, with the constant of linearity defined as the strain per unit temperature and known as the *thermal expansion coefficient*. Thus the strain relation is given by:

$$\epsilon_x = D\sigma_x - \nu D\sigma_y + \alpha T$$
$$\epsilon_y = -\nu D\sigma_x + D\sigma_y + \alpha T$$

Where T denotes the *change* in temperature and α is the thermal expansion coefficient. Thus, the stress-strain relations become:

$$\begin{Bmatrix}\sigma_x\\ \sigma_y\\ \tau_{xy}\end{Bmatrix}=[Q]\left(\begin{Bmatrix}\epsilon_x\\ \epsilon_y\\ \gamma_{xy}\end{Bmatrix}-\begin{Bmatrix}\alpha\\ \alpha\\ 0\end{Bmatrix}T\right)$$

From which, the strain energy relation may be written as:

$$U=\frac{h}{2}\int_{Area}\left(\{\epsilon_m\}^T[Q]\{\epsilon_m\}-\{\epsilon_m\}^T[Q]\{\alpha\}T\right)dA$$

$$\delta U=\frac{h}{2}\int_{Area}\left(2\{\delta\epsilon_m\}^T[Q]\{\epsilon_m\}-\{\delta\epsilon_m\}^T[Q]\{\alpha\}T\right)dA$$

$$\delta U=\{\delta\epsilon_m\}^T\left([k_m]\{w_m\}-\{N_T\}\right)$$

Where the thermal load is:

$$\{N_T\}=\frac{h}{2}\int_{Area}\left([Q]\{\alpha\}T\right)dA$$

Finally, the element equation for the plate becomes:

$$[k_m]\{w_m\}=\{f_m\}+\{P_m\}+\{N_T\}$$

11. Mathematica Code for Some Problems

NOTE: Feel free to copy, modify, and/or use all the codes provided in this text.

11.1 The Beam Program

The following program produces the beam stiffness matrix as described in section 4.3.

```
Hw={1,x,x*x,x*x*x};
Hwx=D[Hw,x];
Hwxx=D[Hwx,x];

x=0;
Tb1=Hw;
Tb2=Hwx;
x=Le;
Tb3=Hw;
Tb4=Hwx;
Clear[x];

TB={Tb1,Tb2,Tb3,Tb4};
TBINV=Inverse[TB];

Kb=Inner[Times,Transpose[{Hwxx}],{Hwxx}];
Kb=Simplify[Integrate[Kb,{x,0,Le}]];
Kg=Transpose[TBINV].Kb.TBINV
```

11.2 The Laplace Equation

The following program produces the element matrix for the Laplace Equation described in section 6.2.

```
Hw={1,x,y,x*y}
Hwx=D[Hw,x]
Hwy=D[Hw,y]

x=0;y=0;T1=Hw;
x=a;y=0;T2=Hw;
x=a;y=b;T3=Hw;
x=0;y=b;T4=Hw;
Clear[x];Clear[y];

TB={T1,T2,T3,T4};
TBINV=Simplify[Inverse[TB]];
```

```
Kx=Inner[Times,Transpose[{Hwx}],{Hwx}]
Ky=Inner[Times,Transpose[{Hwy}],{Hwy}]
Kb=Simplify[Integrate[Integrate[Kx+Ky,{x,0,a}],{y,0,b}]]
Kg=Simplify[Transpose[TBINV].Kb.TBINV]//MatrixForm
```

12. Maxima Code for Some Problems

NOTE: Feel free to copy, modify, and/or use all the codes provided in this text.

Maxima is an open source symbolic manipulator. You may download it from
http://andrejv.github.io/wxmaxima/download.html

12.1 Generating the Beam Stiffness Matrix

The following codes describes how to obtain the beam stiffness matrix as described in section 4.3

```
H:matrix([1,x,x^2,x^3]);
Hx:diff(H,x);
Hxx:diff(H,x,2);

x:0;
Tb1:H,numer;
Tb2:Hx,numer;
x:L;
Tb3:H,numer;
Tb4:Hx,numer;
TT:matrix(Tb1[1],Tb2[1],Tb3[1],Tb4[1]);
T1:invert(TT);
kill(x);

Ke:transpose(T1).integrate(transpose(Hxx).Hxx,x,0,L).T1;
```

12.2 In-Plane Loading Stiffness Matrix

The code below, evaluates the stiffness matrix for the in-plane loading problem described in section 10. The Octave code is listed in in section 13.10.

```
Hu:matrix([1,x,y,x*y,0,0,0,0]);
Hv:matrix([0,0,0,0,1,x,y,x*y]);

Hux:diff(Hu,x);
Huy:diff(Hu,y);
Hvx:diff(Hv,x);
Hvy:diff(Hv,y);
```

```
x:0;y:0;
Tb1:Hu,numer;
Tb5:Hv,numer;
x:a;y:0;
Tb2:Hu,numer;
Tb6:Hv,numer;
x:a;y:b;
Tb3:Hu,numer;
Tb7:Hv,numer;
x:0;y:b;
Tb4:Hu,numer;
Tb8:Hv,numer;

TT:matrix(Tb1[1],Tb2[1],Tb3[1],Tb4[1],Tb5[1],Tb6[1],Tb7[1],Tb8[1])
;
T1:invert(TT);
kill(x);kill(y);

Cm:matrix(Hux[1],Hvy[1],Huy[1]+Hvx[1]);
Qq:matrix([D,nu*D,0],[nu*D,D,0],[0,0,2*G]);

Km:Thickness.transpose(T1).integrate(integrate(transpose(Cm).Qq.Cm
,x,0,a),y,0,b).T1;
```

13. Octave Code for Some Problems

NOTE: Feel free to copy, modify, and/or use all the codes provided in this text.

Octave is an open source package that provides functionalities similar to those of Matlab. You may download the package from https://www.gnu.org/software/octave/

13.1 The Truss Problem

The program below is written for the problem described in section 3.4.2. To use the same code for solving other truss problems, you may change the data section in the beginning of the program.

```
%Problem Data
EE=71e9;
AA=0.01*0.01;
NN=6;
NE=9;

% X,Y,Fx,Fy, u, v
Nodes=[0       ,0          ,0   ,0   ,1  ,2 ;
       2.0     ,0          ,0   ,0   ,3  ,4 ;
       2.5     ,0.5*sqrt(3),-100,0   ,5  ,6 ;
       1.5     ,0.5*sqrt(3),0   ,0   ,7  ,8 ;
       0.5     ,0.5*sqrt(3),0   ,0   ,9  ,10;
       1.0     ,0          ,0   ,0   ,11 ,12];

%      Node#1, Node#2, Modulus of Elasticity , Area
Elements=[1 ,5 ,EE ,AA;
          1 ,6 ,EE ,AA;
          5 ,4 ,EE ,AA;
          4 ,3 ,EE ,AA;
          6 ,2 ,EE ,AA;
          2 ,3 ,EE ,AA;
          2 ,4 ,EE ,AA;
          6 ,4 ,EE ,AA;
          6 ,5 ,EE ,AA];

BCs=[1,2,4]; %Fixed degrees of freedom
%Creating Empty Global matrix and force vector
KGlobal=zeros(2*NN);
FGlobal=zeros(2*NN,1);

%Looping on all elements
for ii=1:NE
    %Identifying the current element nodes and data
```

```
    Node1=Elements(ii,1);
    Node2=Elements(ii,2);
    Stiff=Elements(ii,3);
     Area=Elements(ii,4);
    %Getting the nodes' coordinates
       x1=Nodes(Node1,1);
       x2=Nodes(Node2,1);
       y1=Nodes(Node1,2);
       y2=Nodes(Node2,2);
    %Getting the nodes' degrees of freedom
       u1=Nodes(Node1,5);
       v1=Nodes(Node1,6);
       u2=Nodes(Node2,5);
       v2=Nodes(Node2,6);
       UV=[u1,v1,u2,v2];
    %Creating the stiffness matrix of each element
    LL=sqrt((x2-x1)^2+(y2-y1)^2); %Element length
    CC=(x2-x1)/LL; %Cos(Theta)
    SS=(y2-y1)/LL; %Sin(Theta)
    %Rotation MAtrix
    TT(:,:,ii)=[CC,SS,0 ,0  ;
        -SS,CC ,0 ,0  ;
        0 ,0  ,CC,SS;
        0 ,0  ,-SS,CC];
    %Element stiffness matrix in local coordinates
    Kl=[1,0,-1,0; 0,0,0,0; -1,0,1,0; 0,0,0,0]*Stiff*Area/LL;
    %Element stiffness matrix in GLOBAL coordinates
    KK(:,:,ii)=TT(:,:,ii)'*Kl*TT(:,:,ii);
    %Assembling the global stiffness matrix
    KGlobal(UV,UV)=KGlobal(UV,UV)+KK(:,:,ii);
end
%Filling the global force vector from nodes' data
for ii=1:NN
    FGlobal(Nodes(ii,5))=Nodes(ii,3);
    FGlobal(Nodes(ii,6))=Nodes(ii,4);
end

%Separating Auxiliary equations
KAux=KGlobal(BCs,:);
KAux(:,BCs)=[];
%Getting reduced stiffness and force vector
%   by applying boundary conditions
KReduced=KGlobal;
KReduced(BCs,:)=[];
KReduced(:,BCs)=[];
FReduced=FGlobal;
FReduced(BCs)=[];

%Solving for the displacements
```

```
Displacements=KReduced\FReduced
%Evaluating the reactions from the auxiliary equations
Reactions=KAux*Displacements

%To obtain the element forces ...
BCsC=[1:2*NN]';  %Complementary boundary conditions
BCsC(BCs)=[];
Displ=zeros(2*NN,1);
%Storing the resulting displacements in their respective location
Displ(BCsC)=Displacements;

%Looping on the elements
for ii=1:NE
    Node1=Elements(ii,1);
    Node2=Elements(ii,2);
       u1=Nodes(Node1,5);
       v1=Nodes(Node1,6);
       u2=Nodes(Node2,5);
       v2=Nodes(Node2,6);
       UV=[u1,v1,u2,v2];
    %Evaluating the local force vector
    ff(:,ii)=TT(:,:,ii)*KK(:,:,ii)*Displ(UV);
end
%Element forces are tension when f1 is negative
ElementForces=-ff(1,:)'
```

13.2 The beam problem

In the below code, the program that may obtain the displacement and reactions for a cantilever beam with tip load is listed. The procedure of the program is described in section 4.

```
%Clearing the memory and display
clear all
clc
%Problem Data
NE=2; %number of elements
Length=2.0; %beam length
Width=0.02; %beam width
Thickness=0.01; %beam thickness
Modulus=71e9; %Modulus of Elasticity Aluminum (GPa)
Rho=2700; %Density (Kg/m^3)
Alpha=22.5e-6; %Thermal Expansion coefficientt
%Cross-section area
Area=Width*Thickness;
%Second moment of area
Imoment=Width*Thickness*Thickness*Thickness/12;
Le=Length/NE; %Element Length
%Element stiffness matrix
Ke=Modulus*Imoment*[12,6*Le,-12,6*Le; ...
          6*Le,4*Le*Le,-6*Le,2*Le*Le; ...
          -12,-6*Le,12,-6*Le; ...
          6*Le,2*Le*Le,-6*Le,4*Le*Le]/Le/Le/Le;
%Global stiffness and mass matrix assembly
%Initializing an empty matrix
KGlobal=zeros(2*(NE+1),2*(NE+1));
%Assembling the global matrix
for ii=1:NE
    KGlobal(2*ii-1:2*(ii+1),2*ii-1:2*(ii+1))= ...
                KGlobal(2*ii-1:2*(ii+1),2*ii-1:2*(ii+1))+Ke;
end
%For a cantilever beam the first and second degree of freedom are
fixed
%Obtaining the auxiliary equations
KAux=KGlobal(1:2, 3:2*(NE+1));
%Applying the boundary conditions
KGlobal(1:2,:)=[];
KGlobal(:,1:2)=[];
%force Vector
FGlobal=zeros(2*NE,1); %This is the empty force fector
FGlobal(2*NE-1)=1; %Adding a single point load at the tip
%Obtainning the solution displacement field
WW=inv(KGlobal)*FGlobal
%Obtaining the support reaction
```

```
Reactions= KAux*WW
```

13.3 The Frame Problem

The program below is written for the problem described in section 5. The program is extremely similar to that of section 13.1. To use the same code for solving other frame problems, you may change the data section in the beginning of the program.

```
clear all
clc
close all

%Problem Data
EE=71e9;
AA=0.01*0.01;
II=AA*0.01*0.01/12;
NN=6;
NE=9;

% X,Y,Fx,Fy,M, u, v, v'
Nodes=[0       ,0           ,0   ,0   ,0   ,1  ,2 ,3 ;
       2.0     ,0           ,0   ,0   ,0   ,4  ,5 ,6 ;
       2.5     ,0.5*sqrt(3),-100,0   ,0   ,7  ,8 ,9 ;
       1.5     ,0.5*sqrt(3),0   ,0   ,0   ,10 ,11,12;
       0.5     ,0.5*sqrt(3),0   ,0   ,0   ,13 ,14,15;
       1.0     ,0           ,0   ,0   ,0   ,16 ,17,18];

% Node#1, Node#2, Modulus of Elasticity , Area, Second moment of
area
Elements=[1 ,5 ,EE ,AA,II;
          1 ,6 ,EE ,AA,II;
          5 ,4 ,EE ,AA,II;
          4 ,3 ,EE ,AA,II;
          6 ,2 ,EE ,AA,II;
          2 ,3 ,EE ,AA,II;
          2 ,4 ,EE ,AA,II;
          6 ,4 ,EE ,AA,II;
          6 ,5 ,EE ,AA,II];

BCs=[1,2,5]; %Fixed degrees of freedom
%Creating Empty Global matrix and force vector
KGlobal=zeros(3*NN);
FGlobal=zeros(3*NN,1);

%Looping on all elements
for ii=1:NE
    %Identifying the current element nodes and data
    Node1=Elements(ii,1);
    Node2=Elements(ii,2);
```

```
    Stiff=Elements(ii,3);
     Area=Elements(ii,4);
      IIe=Elements(ii,5);
    %Getting the nodes' coordinates
       x1=Nodes(Node1,1);
       x2=Nodes(Node2,1);
       y1=Nodes(Node1,2);
       y2=Nodes(Node2,2);
    %Getting the nodes' degrees of freedom
       u1=Nodes(Node1,6);
       v1=Nodes(Node1,7);
       t1=Nodes(Node1,8);
       u2=Nodes(Node2,6);
       v2=Nodes(Node2,7);
       t2=Nodes(Node2,8);
       UV=[u1,v1,t1,u2,v2,t2];
    %Creating the stiffness matrix of each element
    LL=sqrt((x2-x1)^2+(y2-y1)^2); %Element length
    CC=(x2-x1)/LL; %Cos(Theta)
    SS=(y2-y1)/LL; %Sin(Theta)
    %Rotation MAtrix
    TT(:,:,ii)=[CC,SS,0 , 0, 0 ,0  ;
                -SS, CC,0 , 0, 0 ,0  ;
                0 , 0 ,1 , 0, 0 ,0  ;
                0 , 0 ,0 ,CC,SS,0  ;
                0 , 0 ,0 ,-SS, CC,0
                0 , 0 ,0 , 0, 0 ,1  ];
    %Element stiffness matrix in local coordinates
    Kl=[Area, 0           , 0       ,-Area, 0           , 0
;
         0  , 12*IIe/LL/LL, 6*IIe/LL, 0   , -12*IIe/LL/LL,
6*IIe/LL ;
         0  , 6*IIe/LL    , 4*IIe   , 0   , -6*IIe/LL    , 2*IIe
;
       -Area, 0           , 0       , Area, 0           , 0
;
         0  ,-12*IIe/LL/LL,-6*IIe/LL, 0   ,
12*IIe/LL/LL,-6*IIe/LL ;
         0  , 6*IIe/LL    , 2*IIe   , 0   , -6*IIe/LL    , 4*IIe
]*Stiff/LL;
    %Element stiffness matrix in GLOBAL coordinates
    KK(:,:,ii)=TT(:,:,ii)'*Kl*TT(:,:,ii);
    %Assembling the global stiffness matrix
    KGlobal(UV,UV)=KGlobal(UV,UV)+KK(:,:,ii);
end
%Filling the global force vector from nodes' data
for ii=1:NN
    FGlobal(Nodes(ii,6))=Nodes(ii,3);
    FGlobal(Nodes(ii,7))=Nodes(ii,4);
```

```
    FGlobal(Nodes(ii,8))=Nodes(ii,5);
end

%Separating Auxiliary equations
KAux=KGlobal(BCs,:);
KAux(:,BCs)=[];
%Getting reduced stiffness and force vector
%   by applying boundary conditions
KReduced=KGlobal;
KReduced(BCs,:)=[];
KReduced(:,BCs)=[];
FReduced=FGlobal;
FReduced(BCs)=[];

%Solving for the displacements
Displacements=KReduced\FReduced
%Evaluating the reactions from the auxiliary equations
Reactions=KAux*Displacements

%To obtain the element forces ...
BCsC=[1:3*NN]';  %Complementary boundary conditions
BCsC(BCs)=[];
Displ=zeros(3*NN,1);
%Storing the resulting displacements in their respective location
Displ(BCsC)=Displacements;

%Looping on the elements
for ii=1:NE
    Node1=Elements(ii,1);
    Node2=Elements(ii,2);
       u1=Nodes(Node1,6);
       v1=Nodes(Node1,7);
       t1=Nodes(Node1,8);
       u2=Nodes(Node2,6);
       v2=Nodes(Node2,7);
       t2=Nodes(Node2,8);
       UV=[u1,v1,t1,u2,v2,t2];
    %Evaluating the loval force vector
    ff(:,ii)=TT(:,:,ii)*KK(:,:,ii)*Displ(UV);
end
%Element foces are tension when f1 is negative
ElementForces=-ff(1,:)'
```

13.4 The Laplace Problem – Rectangular Mesh

In the following, you will find a simple program that solves the Laplace problem in 2-D domain as described in section 6. The program creates the element matrices using numerical integration described in section 7.2, and assembles the global matrix using the techniques used in the program in section 13.3 to utilize the abilities of Octave (a different algorithm for assembly is described in section 6.3).

```
clear all
close all
clc
%Problem Data
LengthX=1;
LengthY=1;
Nx=100;
Ny=100;
Ne=Nx*Ny;
Nn=(Nx+1)*(Ny+1);
%Length of the elements in x and y-directions
Lx=LengthX/Nx;
Ly=LengthY/Ny;
%Evaluating the stiffness matrix
T1=CalcTRect(Lx,Ly);
KK=CalcRect(T1,Lx,Ly);
[Nodes,Elements,BCs]=GenerateMesh(Nx,Ny,Lx,Ly);
%Creating Empty Global matrix
KGlobal=zeros(Nn,Nn);
%Looping on all elements
for ii=1:Ne
    %Identifying the current element nodes and data
    Node1=Elements(ii,1);
    Node2=Elements(ii,2);
    Node3=Elements(ii,3);
    Node4=Elements(ii,4);
    UV=[Node1,Node2,Node3,Node4];
    %Assembling the global stiffness matrix
    KGlobal(UV,UV)=KGlobal(UV,UV)+KK;
end
BCsC=[1:Nn]';  %Complementary boundary conditions
BCsC(BCs)=[];
%The matrix that will be multiplied by the boundary values
KAux=KGlobal(:,BCs);
KAux(BCs,:)=[];
%Getting reduced stiffness and force vector
%   by applying boundary conditions
KReduced=KGlobal;
```

```
KReduced(BCs,:)=[];
KReduced(:,BCs)=[];
%The right-hand-side vector
FReduced=-KAux*Nodes(BCs,3);
%Solving for the function values
Displacements=KReduced\FReduced;
%Saving the solution values in the Nodes' registry
Nodes(BCsC,3)=Displacements
```

```
function [Nodes,Elements,BCs]=GenerateMesh(Nx,Ny,Lx,Ly)
Ne=Nx*Ny;
Nn=(Nx+1)*(Ny+1);
%Creating the Nodes' registry
Nodes=zeros(Nn,3); % x,y,function value
BCsCounter=0;
for ii=1:Nx+1
    Xx=(ii-1)*Lx; %x-coordinate
    for jj=1:Ny+1
        Yy=(jj-1)*Ly; %y-coordinate
        NN=(jj-1)*(Nx+1) + ii; %node number
        Nodes(NN,:)=[Xx,Yy,0]; %saving the data
        if ii==1 %setting the boundary values
            BCsCounter=BCsCounter+1;
            BCs(BCsCounter,1)=NN;
            Nodes(NN,3)=1;
        elseif or(or(jj==1,jj==Ny+1),ii==Nx+1)
            BCsCounter=BCsCounter+1;
            BCs(BCsCounter,1)=NN;
            Nodes(NN,3)=0;
        endif
    endfor
endfor
%Creating the Elements' registry
Elements=zeros(Ne,4);
for ii=1:Nx
    for jj=1:Ny
        NN=(jj-1)*Nx+ii;
        Elements(NN,1)=(jj-1)*(Nx+1) + ii; %Node1 number
        Elements(NN,2)=(jj-1)*(Nx+1) + ii+1; %Node2 number
        Elements(NN,3)=jj*(Nx+1) + ii+1; %Node3 number
        Elements(NN,4)=jj*(Nx+1) + ii; %Node4 number
    endfor
endfor
endfunction
```

13.5 Beam Stiffness Matrix Using Numerical Integration

In the following code, the beam stiffness matrix is evaluated using Gauss Quadrature. The function receives the parameters D (bending stiffness EI), TbInv (inverse of the transformation matrix), and LengthX (element length). Then the function uses 8-point Gauss Quadrature (3 points would have been enough for this particular problem) to evaluate the integration. The Gauss Quadrature data are set in a 2x8 matrix with the firs row having the weighing values and the second row having the Gauss domain coordinates. The algorithm used is described in section 7.1. The function calls another function “CalcHxx” in its body. CalcHxx is listed after the main body of the function for the purpose of demonstration as well as CalcH and CalcHx

```
function KB=CalcLinear(D,TbInv,LengthX)

GaussConstants=zeros(2,8);
GaussConstants(2, 1) = -0.9602898565;
GaussConstants(2, 2) = -0.7966664774;
GaussConstants(2, 3) = -0.5255324099;
GaussConstants(2, 4) = -0.1834346424;
GaussConstants(2, 5) = 0.1834346424;
GaussConstants(2, 6) = 0.5255324099;
GaussConstants(2, 7) = 0.7966664774;
GaussConstants(2, 8) = 0.9602898564;

GaussConstants(1, 1) = 0.1012285362;
GaussConstants(1, 2) = 0.2223810344;
GaussConstants(1, 3) = 0.3137066458;
GaussConstants(1, 4) = 0.3626837833;
GaussConstants(1, 5) = 0.3626837833;
GaussConstants(1, 6) = 0.3137066458;
GaussConstants(1, 7) = 0.2223810344;
GaussConstants(1, 8) = 0.1012285362;
            KB=zeros(4,4);

%Start the numerical integrration procedure
      for Xi=1:8
         X = LengthX * (GaussConstants(2, Xi) + 1) / 2;
            %*************************
            cb=CalcHxx(X);
            %*************************
            Kb= cb'*D*cb;
            %performing the weighted summation
            KB=KB+GaussConstants(1,Xi)*Kb;
            %End of Calculation loop body
      end
      KB= TbInv'*KB*TbInv;
```

```
      %Multiplying the resulting matreces by Jacobian
      KB = KB * LengthX / 2;
```

```
function H=CalcH(x)

H=zeros(1,4);

f1=1;
f2=x;
f3=x*x;
f4=x*x*x;

H(1)=f1;
H(2)=f2;
H(3)=f3;
H(4)=f4;
```

```
function H=CalcHx(x)

H=zeros(1,4);

f1=0;
f2=1;
f3=2*x;
f4=3*x*x;

H(1)=f1;
H(2)=f2;
H(3)=f3;
H(4)=f4;
```

```
function H=CalcHxx(x)

H=zeros(1,4);

f1=0;
f2=0;
f3=2;
f4=6*x;

H(1)=f1;
H(2)=f2;
H(3)=f3;
H(4)=f4;
```

13.6 2-D Element Matrix Using Gauss Quadrature

In the following code, the matrix for a 2-D rectangular element for the Laplace problem is evaluated using Gauss Quadrature. The function receives the parameters TbInv (inverse of the transformation matrix), LengthX (element length in the x-direction), and LengthY (element length in the y-direction). Then the function uses 8-point Gauss Quadrature (3 points would have been enough for this particular problem) to evaluate the integration. The Gauss Quadrature data are set in a 2x8 matrix with the firs row having the weighing values and the second row having the Gauss domain coordinates. The algorithm used is described in section 7.2. The function calls other functions "CalcHx and CalcHy" in its body. CalcHx and CalcHy are listed after the main body of the function for the purpose of demonstration as well as CalcH.

```
function KB=CalcRect(TbInv,LengthX,LengthY)

GaussConstants=zeros(2,8);
GaussConstants(2, 1) = -0.9602898565;
GaussConstants(2, 2) = -0.7966664774;
GaussConstants(2, 3) = -0.5255324099;
GaussConstants(2, 4) = -0.1834346424;
GaussConstants(2, 5) = 0.1834346424;
GaussConstants(2, 6) = 0.5255324099;
GaussConstants(2, 7) = 0.7966664774;
GaussConstants(2, 8) = 0.9602898564;

GaussConstants(1, 1) = 0.1012285362;
GaussConstants(1, 2) = 0.2223810344;
GaussConstants(1, 3) = 0.3137066458;
GaussConstants(1, 4) = 0.3626837833;
GaussConstants(1, 5) = 0.3626837833;
GaussConstants(1, 6) = 0.3137066458;
GaussConstants(1, 7) = 0.2223810344;
GaussConstants(1, 8) = 0.1012285362;
            KB=zeros(4,4);

%Start the numerical integrration procedure
   for Xi=1:8
     X = LengthX * (GaussConstants(2, Xi) + 1) / 2;
     for Yi=1:8
         Y = LengthY * (GaussConstants(2, Yi) + 1) / 2;
            %*************************
            cx=CalcHx(X,Y);
            cy=CalcHy(X,Y);
            %*************************
            Kb= (cx'*cx)+(cy'*cy);
            %performing the weighted summation
            KB=KB+GaussConstants(1,Xi)*GaussConstants(1,Yi)*Kb;
```

```
            %End of Calculation loop body
      end
   end
   KB= TbInv'*KB*TbInv;
   %Multiplying the resulting matreces by Jacobian
   KB = KB * LengthX * LengthY/ 4;
```

```
function H=CalcH(x,y)

H=zeros(1,4);

f1=1;
f2=x;
g1=1;
g2=y;

H(1)=f1*g1;
H(2)=f2*g1;
H(3)=f1*g2;
H(4)=f2*g2;
```

```
function H=CalcHx(x,y)

H=zeros(1,4);

f1=0;
f2=1;
g1=1;
g2=y;

H(1)=f1*g1;
H(2)=f2*g1;
H(3)=f1*g2;
H(4)=f2*g2;
```

```
function H=CalcHy(x,y)

H=zeros(1,4);

f1=1;
f2=x;
g1=0;
g2=1;

H(1)=f1*g1;
H(2)=f2*g1;
```

```
H(3)=f1*g2;
H(4)=f2*g2;
```

13.7 2-D Laplace Problem using Quadrilateral Elements

In the following Octave code, you will find the *different code* compared to that listed in section 13.6. The differences are:

1- The element matrix is evaluated for each element in the assembly loop

2- The interpolation polynomial transformation matrix changes from one element to the other

3- When evaluating the element matrix, the derivatives of the physical coordinated with respect to the Gauss coordinates is evaluated at each point.

The procedure is described in section 7.3.

```
%Looping on all elements
for ii=1:Ne
    %Identifying the current element nodes and data
    Node1=Elements(ii,1);
    Node2=Elements(ii,2);
    Node3=Elements(ii,3);
    Node4=Elements(ii,4);
    UV=[Node1,Node2,Node3,Node4];

Xnodes=[Nodes(Node1,1);Nodes(Node2,1);Nodes(Node3,1);Nodes(Node4,1
)];

Ynodes=[Nodes(Node1,2);Nodes(Node2,2);Nodes(Node3,2);Nodes(Node4,2
)];
    T1=CalcTxy(Xnodes,Ynodes);
    KK=CalcQuad(T1,Xnodes,Ynodes);
    %Assembling the global stiffness matrix
    KGlobal(UV,UV)=KGlobal(UV,UV)+KK;
end
```

```
function TbInv=CalcTxy (Xv,Yv)
    T1=CalcH(Xv(1),Yv(1));
    T2=CalcH(Xv(2),Yv(2));
    T3=CalcH(Xv(3),Yv(3));
    T4=CalcH(Xv(4),Yv(4));
    TT=[T1;T2;T3;T4];
    TbInv=inv(TT);
endfunction
```

```
function KB=CalcQuad(T1,Xv,Yv)

GaussConstants=zeros(2,8);
GaussConstants(2, 1) = -0.9602898565;
GaussConstants(2, 2) = -0.7966664774;
GaussConstants(2, 3) = -0.5255324099;
GaussConstants(2, 4) = -0.1834346424;
GaussConstants(2, 5) = 0.1834346424;
GaussConstants(2, 6) = 0.5255324099;
GaussConstants(2, 7) = 0.7966664774;
GaussConstants(2, 8) = 0.9602898564;

GaussConstants(1, 1) = 0.1012285362;
GaussConstants(1, 2) = 0.2223810344;
GaussConstants(1, 3) = 0.3137066458;
GaussConstants(1, 4) = 0.3626837833;
GaussConstants(1, 5) = 0.3626837833;
GaussConstants(1, 6) = 0.3137066458;
GaussConstants(1, 7) = 0.2223810344;
GaussConstants(1, 8) = 0.1012285362;
            KB=zeros(4,4);
Txy=inv([1,-1,-1,1;1,1,-1,-1;1,1,1,1;1,-1,1,-1]);

%Start the numerical integrration procedure
   for Xi=1:8
     xx=GaussConstants(2, Xi);
     for Yi=1:8
         yy=GaussConstants(2, Yi);
         HT=CalcH(xx,yy)*Txy;
         HTx=CalcHx(xx,yy)*Txy;
         HTy=CalcHy(xx,yy)*Txy;
         X = HT*Xv;
         Y = HT*Yv;
         Xx= HTx*Xv;
         Xy= HTy*Xv;
         Yx= HTx*Yv;
         Yy= HTy*Yv;
            %*************************
            cx=CalcHx(X,Y);
            cy=CalcHy(X,Y);
            %*************************
            Kb= ((cx'*cx)+(cy'*cy));
            %performing the weighted summation

KB=KB+GaussConstants(1,Xi)*GaussConstants(1,Yi)*Kb*(Xx*Yy+Xy*Yx);
            %End of Calculation loop body
      end
   end
   KB= T1'*KB*T1;
```

13.8 Column Buckling Program

The code below may be used to find the buckling load for a column as described in section 9.1.

```
%Clearing the memory and display
clear all
clc
%Problem Data
NE=5; %number of elements
Length=1.0; %beam length
Width=0.02; %beam width
Thickness=0.001; %beam thickness
Modulus=71e9; %Modulus of Elasticity Aluminum (GPa)
%Cross-section area
Area=Width*Thickness;
%Second moment of area
Imoment=Width*Thickness*Thickness*Thickness/12;
Le=Length/NE; %Element Length
%Element stiffness matrix
Ke=Modulus*Imoment*[12     ,6*Le    ,-12  ,6*Le; ...
                    6*Le   ,4*Le*Le ,-6*Le,2*Le*Le; ...
                    -12    ,-6*Le   ,12   ,-6*Le; ...
                    6*Le   ,2*Le*Le ,-6*Le,4*Le*Le]/Le/Le/Le;
%Element Geometricstiffness matrix
Kg=[36  ,3*Le   ,-36  ,3*Le; ...
    3*Le,4*Le*Le,-3*Le,-Le*Le; ...
    -36 ,-3*Le  ,36   ,-3*Le; ...
    3*Le,-Le*Le ,-3*Le,4*Le*Le]/30/Le;
%Global matrices assembly
%Initializing empty matrices
KGlobal=zeros(2*(NE+1),2*(NE+1));
KGGlobal=zeros(2*(NE+1),2*(NE+1));
%Assembling the global matrix
for ii=1:NE
    KGlobal(2*ii-1:2*(ii+1),2*ii-1:2*(ii+1))= ...
                  KGlobal(2*ii-1:2*(ii+1),2*ii-1:2*(ii+1))+Ke;
    KGGlobal(2*ii-1:2*(ii+1),2*ii-1:2*(ii+1))= ...
                  KGGlobal(2*ii-1:2*(ii+1),2*ii-1:2*(ii+1))+Kg;
end
%Applying the boundary conditions
BCS=[1,2*(NE+1)-1]; %Simply Supported BCS
KGlobal(BCS,:)=[];
KGlobal(:,BCS)=[];
KGGlobal(BCS,:)=[];
KGGlobal(:,BCS)=[];
%Evaluating the eigenvalues
sort(eig(inv(KGGlobal)*Kglobal))
```

13.9 Dynamics of Beams

The code below may be used to find the natural frequencies and frequency response of beams as described in section 9.2.5.

```
%Clearing the memory and display
clear all
clc
%Problem Data
NE=5; %number of elements
Length=1.0; %beam length
Width=0.02; %beam width
Thickness=0.001; %beam thickness
Modulus=71e9; %Modulus of Elasticity Aluminum (GPa)
Rho=2700; %Density (Kg/m^3)
%Cross-section area
Area=Width*Thickness;
%Second moment of area
Imoment=Width*Thickness*Thickness*Thickness/12;
Le=Length/NE; %Element Length
%Element stiffness matrix
Ke=Modulus*Imoment*[12     ,6*Le    ,-12  ,6*Le; ...
                    6*Le   ,4*Le*Le ,-6*Le,2*Le*Le; ...
                    -12    ,-6*Le   ,12   ,-6*Le; ...
                    6*Le   ,2*Le*Le ,-6*Le,4*Le*Le]/Le/Le/Le;
%Element Mass matrix
Me=Rho*Area*Le*[156  ,22*Le    ,54    ,-13*Le; ...
                22*Le, 4*Le*Le ,13*Le ,-3*Le*Le; ...
                54   ,13*Le    ,156   ,-22*Le; ...
               -13*Le,-3*Le*Le ,-22*Le,4*Le*Le]/420;
%Global matrices assembly
%Initializing empty matrices
KGlobal=zeros(2*(NE+1),2*(NE+1));
MGlobal=zeros(2*(NE+1),2*(NE+1));
FGlobal=zeros(2*(NE+1),1); %Global Force Vector
%Assembling the global matrix
for ii=1:NE
    KGlobal(2*ii-1:2*(ii+1),2*ii-1:2*(ii+1))= ...
                  KGlobal(2*ii-1:2*(ii+1),2*ii-1:2*(ii+1))+Ke;
    MGlobal(2*ii-1:2*(ii+1),2*ii-1:2*(ii+1))= ...
                  MGlobal(2*ii-1:2*(ii+1),2*ii-1:2*(ii+1))+Me;
end
%Applying the boundary conditions
BCS=[1,2*(NE+1)-1]; %Simply Supported BCS
KGlobal(BCS,:)=[];
KGlobal(:,BCS)=[];
```

```
MGlobal(BCS,:)=[];
MGlobal(:,BCS)=[];
FGlobal(BCS)=[];
%Evaluating the Natural Frequencies
vvv=sort(sqrt(eig(inv(MGlobal)*KGlobal)))
```

Adding a few lines to obtain the frequency response

```
%Evaluating the frequency response
%The excitation will be through
% a concentrated harmonic moment at the
% left side (DOF# 1)
%The measurement will be measured from
% the displacement of the nede just before the
% right support (DOF # (2*NE - 1))
FGlobal(1)=1;
FREQ=[0];
for ii = 0:1000
   FREQ(ii+1,1)=0.5*ii;
   KD=KGlobal - FREQ(ii+1,1)*FREQ(ii+1,1)*MGlobal;
   RESP=inv(KD)*FGlobal;
   Res(ii+1,1)=20*log(abs(RESP(2*NE-1)));
endfor
plot(FREQ,Res)
grid
```

13.10 In-plane plate loading

(You may download the full set of functions fo this problem from GitHub at this link)

The following code evaluate the stiffness matrix for plate in-plane loading problem described in section 10. The Maxima codes is listed in section 12.2.

(The code in the frame below is saved under the name MainRect.m in the GitHub repository)

```
clear all
close all
clc
%Problem Data
Stiff=71e9;
Nu=0.3;
Thickness=0.0005;
GStiff=Stiff/2/(1+Nu);
LengthX=1;
LengthY=1;
%Number of elements in each direction
Nx=1;
Ny=1;
Ne=Nx*Ny; %Total Number of elements
Nn=(Nx+1)*(Ny+1);%Number of nodes
%Length of the elements in x and y-directions
Lx=LengthX/Nx;
Ly=LengthY/Ny;
%The transformation matrix
T1=CalcTRect(Lx,Ly);
%The PLATE stiffness matrix
Dd=Stiff/(1-Nu*Nu);
Qq=[Dd,Nu*Dd,0; Nu*Dd,Dd,0; 0,0,2*GStiff];
%Evaluating the ELEMENT stiffness matrix
KK=Thickness*T1'*CalcLinear(Qq,Lx,Ly)*T1;
%Creating the element/node registry
[Nodes,Elements,BCs]=GenerateMesh(Nx,Ny,Lx,Ly);
BCs=[1;Nx+1;Nn+1;Nn+1+Nx]
%Creating Empty Global matrix
KGlobal=zeros(2*Nn,2*Nn);
%Looping on all elements
for ii=1:Ne
  %Looping for each node-row
  for jj=1:4
    %Storing the node number
    Nodej=Elements(ii,jj);
    %Storing the DOF's of the node
    Uj=Nodes(Nodej,3);
```

```
    Vj=Nodes(Nodej,4);
    %Looping for each node-column
    for kk=1:4
      %Storing the node number
      Nodek=Elements(ii,kk);
      %Storing the node DOF's
      Uk=Nodes(Nodek,3);
      Vk=Nodes(Nodek,4);
      %Assembling the global stiffness matrix
      KGlobal(Uj,Uk)=KGlobal(Uj,Uk)+KK(jj,kk);
      KGlobal(Vj,Vk)=KGlobal(Vj,Vk)+KK(jj+4,kk+4);
      KGlobal(Uj,Vk)=KGlobal(Uj,Vk)+KK(jj,kk+4);
      KGlobal(Vj,Uk)=KGlobal(Vj,Uk)+KK(jj+4,kk);
    endfor
  endfor
end
BCsC=[1:2*Nn]';  %Complementary boundary conditions
BCsC(BCs)=[];
%The matrix that will be multiplied by the boundary values
KAux=KGlobal(:,BCsC);
KAux(BCsC,:)=[];
%Getting reduced stiffness and force vector
%   by applying boundary conditions
KReduced=KGlobal;
KReduced(BCs,:)=[];
KReduced(:,BCs)=[];
%The right-hand-side vector
FReduced=zeros(2*Nn,1);
FReduced(Nn)=10;
FReduced(2*Nn)=0;
FReduced(BCs)=[];
%Solving for the function values
Displacements=KReduced\FReduced
Reactions=KAux*Displacements
```

The code below evaluate the element stiffness matrix (some of the functions called are also included.)

```
function KB=CalcLinear(Qq,LengthX,LengthY)

GaussPoints=5;
GaussConstants=GetGC(GaussPoints);

            KB=zeros(8,8);

%Start the numerical integrration procedure
   for Xi=1:GaussPoints
     X = LengthX * (GaussConstants(2, Xi) + 1) / 2;
```

```
     for Yi=1:GaussPoints
         Y = LengthY * (GaussConstants(2, Yi) + 1) / 2;
            %*************************
            cm=CalcCm(X,Y);
            %*************************
            %performing the weighted summation

KB=KB+GaussConstants(1,Xi)*GaussConstants(1,Yi)*cm'*Qq*cm;
            %End of Calculation loop body
      end
   end
```

```
function Cm=CalcCm(x,y)
    Hux=CalcHux(x,y);
    Huy=CalcHuy(x,y);
    Hvx=CalcHvx(x,y);
    Hvy=CalcHvy(x,y);
    Cm=[Hux;Hvy;0.5*(Huy+Hvx)];
endfunction

function Hu=CalcHux(x,y)
    Hm=CalcHmx(x,y);
    Hu=[Hm,0,0,0,0];
endfunction

function Hu=CalcHuy(x,y)
    Hm=CalcHmy(x,y);
    Hu=[Hm,0,0,0,0];
endfunction

function Hu=CalcHvx(x,y)
    Hm=CalcHmx(x,y);
    Hu=[0,0,0,0,Hm];
endfunction

function Hu=CalcHvy(x,y)
    Hm=CalcHmy(x,y);
    Hu=[0,0,0,0,Hm];
endfunction

function Hm=CalcHmx(x,y)
    f0=0;
```

```
    f1=1;
    g0=1;
    g1=y;
    Hm=[f0*g0,f1*g0,f0*g1,f1*g1];
endfunction

function Hm=CalcHmy(x,y)
    f0=1;
    f1=x;
    g0=0;
    g1=1;
    Hm=[f0*g0,f1*g0,f0*g1,f1*g1];
endfunction

function Hm=CalcHm(x,y)
    f0=1;
    f1=x;
    g0=1;
    g1=y;
    Hm=[f0*g0,f1*g0,f0*g1,f1*g1];
endfunction
```

14. References and Bibliography

[1] Mohammad Tawfik, *In Search for the Super Element: Algorithms to Generate Higher-Order Elements*, ResearchGate.net, 2017, DOI: 10.13140/RG.2.2.24039.75682

[2] J. N. Reddy, *An Introduction to the Finite Element Method*, 3rd ed., McGraw Hill, 2006

[3] Steven Chapra and Raymond Conale, *Numerical Methods for Engineers*, 5th ed., McGraw Hill, 2006

[4] Daniel Zwillinger (editor), *Standard Mathematical Tables and Formulae*, 30th ed., CRC Press, 2000

[5] C. A. J. Fletcher, *Computational Galerkin Methods*, Springer-Verlag, 1984

[6] M. Tawfik, A Spectral Finite Element Model for Thin Plate Vibration, International Congress on Sound and Vibration (ICSV14), Cairns, Australia, 9-12 July 2007. DOI: 10.13140/2.1.1123.6167

[7] M. Tawfik, Higher Order and Spectral Finite Element Model for Thin Plate Vibration – Much Difference?, 5th International Conference on Mathematics and Information Sciences (ICMIS2016), Zewail City of Science and Technology 11-13- Feb. 2016

[8] I. Babuska, B. Szabo, and I. Katz, The p-Version of the Finite Element Method, SIAM Journal of Numerical Analysis, Vol. 18, No. 33, pp. 515-545, June 1981

[9] L. Demkpvicz, J. Kurtz, D. Pardo, M. Paszynski, W. Rachowicz, and A. Zdunek, Computing with hp-Adaptive Finite Elements, Chapman and Hall, 2008

[10] J. Korelc and P. Wriggers, Automation of Finite Element Methods, Springer Verlag, 2016, DOI 10.1007/978-3-319-39005-5

[11] A. Bespalov and N. Heuer, A New H(div) Conforming p-Interpolation Operator in Two Dimensions, ESAIM: Mathematical Modeling and Numerical Analysis, Vol. 45, pp. 255-275, 2011, DOI:10.1051/m2an/2010039

[12] M. Tawfik, Fundamentals of Numerical Analysis, book draft, ResearchGate.net, 2017, DOI: 10.13140/RG.2.2.25680.81925

[13] S. C. Chapra and R. P. Canale, Numerical Methods for Engineers, 5th ed., McGraw Hill, 2006

[14] L. Demkowicz, J. Gopalakrishnan, and J. Schöberl, Polynomial Extension Operators. P art I, Mathematics and Statistics Faculty Publications and Presentations. 61, 2008 http://pdxscholar .library .pdx.edu/mth_fac/61

[15] M. Costable, M. Dauge, and L. Demkovicz, Polynomial Extension Operators for H1, H(curl), H(div) – Spaces on a Cube, Mathematics of Computation, Vol. 77, No. 264, pp. 1967-1999, 2008