3.0 DEVELOPMENT OF ANALYSIS SOFTWARE IN C++

Finite element analysis involves three stages of activities namely:

**Pre processing**, that involves the preparation of data such as nodal coordinate connectivity, boundary conditions and loading and material properties.

**Processing**, that involves stiffness generation, stiffness modification, solution of equations resulting in the evaluation of nodal variables, other derived quantities such as stresses, strains etc.

**Post processing**, that deals with the presentation of results such as deformed configuration, stress and strain distributions etc.

In this chapter, main finite element analysis program is developed in Visual C++. Its accuracy is ascertained by solving simple examples and verifying the results obtained with worked out results, using basic strength of materials solution.[20,87] Once, basic program is validated, it can be applied to the required domain and problem evaluated.

3.1 DESCRIPTION OF THE FINITE ELEMENT PROGRAM

A basic, general purpose finite element program for the three dimensional analysis of a solid continuum is developed using OOP using the FORTRAN subroutine QUAD4 for a four noded plane element.[24]
3.1.1 Functions used in the Program

Main program requires various input data for analysis, which are listed below:

**Geometry data**, which includes the number of nodes, number of elements, nodal coordinates and element connectivity.

**Material property data**, for each element which consists of the Modulus of elasticity, Poisson's ratio and the Unit weight of material.

**Displacement boundary conditions data**, that includes the list of nodes with partial or complete displacement restraints, and the list of restraint degrees of freedom at each such node.

**Load data**, that reads all types of loads applied to the structure such as concentrated nodal loads, distributed surface loads; *water pressure, silt pressure, and load due to road traffic*, and distributed body forces; *static loads due to self weight and ground acceleration*.

Mesh generator program provides all the above data and the output of the mesh generator can directly be fed in the input for the finite element analysis program. A pseudo code for the input to the finite element analysis program developed is modified for a solid continuum as given below.[87]

```plaintext
Input nNodes, nElems
For i = 1, nNodes
    input k, x(k), y(k), z(k)
End For

Element connectivity data

Input the node number corresponding to the node numbers of the basic element in the same order for all the elements.

For i = 1, number of elements
```
Input k, node (1, k), node (2, k), node (n, k)

where, n is the number of nodes of the basic element chosen.

**Displacement boundary conditions**

Destination array is used to store the addresses to which the elements of the matrices $[k]$ and $\{r^p\}$ must be posted. This destination array is first initialized with zeros. The destination array is a rectangular matrix which has many columns as the number of nodes and as many rows as the number of degrees of freedom per node. The displacement boundary conditions are fed into this array as 1’s and 0’s.

**Initialise destination array**

For $i = 1,$ number of nodes

For $j = 1, 2, 3$

Destn $(i, j) = 0$

The number of nodes with displacement boundary condition prescribed is then input. Corresponding to each of this node, the boundary condition should be input as follows:

For $i = 1, n\text{DisplRestraints}$

Input k, destn(1, k), destn(2, k), destn(3, k)

Input ‘1’ for degrees of freedom which are restrained and ‘0’ for degrees of freedom which are free.

The destination array is now scanned column wise and each column from top to bottom. Each ‘1’ is converted into ‘0’ and ‘0’ is replaced by a counter. In this way the elements of the destination array represent the degree of freedom number. The last number in the destination array represents the number of equations, i.e., the total number of degrees of freedom of the finite element model.
For updating the destination array the following procedure is followed.

```
neq = 0;
for i = 1, nNodes
  for j = 1, 3
    if destn (i, j) = 0;
      neq = neq + 1;
      id(i, j) = neq;
    else: id(i, j) = 0;
    destn(i, j) = 0;
  endif
endf
```

**Assembling of element stiffness matrices**

The `nod` array and the `destn` array are used to assemble the element stiffness matrices and the load vectors to get the global stiffness matrix and the global load vector respectively. Before assembly, these global arrays are initialized.

Concentrated nodal loads, if any acting in the degrees of freedom direction, it is fed directly into the global load vector as,

```text
For i = 1, neq
  R(i) = 0;
For j = 1, neq
  R(i, j) = 0;
```

Input nNconc,

where, nconc is the number of nodes at which concentrated loads are applied.

```
For i = 1, nNconc
  input k, Fx, Fy, Fz;
  dof1 = destn(k, 1)
  dof2 = destn(k, 2)
  dof3 = destn(k, 3)
  R(dof1) = Fx
  R(dof2) = Fy
  R(dof3) = Fz
```

**Assembly of global matrices**

Invoke element subprogram to get element stiffness matrix and the load vector.
For \( n = 1, n_{\text{Elems}} \)

call elem stiffness;

\[
\text{nod } 1 = \text{nod}(1, n);
\]

\[
\text{nod } 2 = \text{nod}(2, n);
\]

\[
\text{nod } 3 = \text{nod}(3, n);
\]

\[
\ldots \quad \ldots \quad \ldots
\]

\[
\text{nod } i = \text{nod}(i, n);
\]

\[
kk(1) = \text{destn}(1, \text{nod } 1)
\]

\[
kk(2) = \text{destn}(2, \text{nod } 1)
\]

\[
kk(3) = \text{destn}(3, \text{nod } 1)
\]

\[
kk(4) = \text{destn}(1, \text{nod } 2)
\]

\[
kk(5) = \text{destn}(2, \text{nod } 2)
\]

\[
kk(6) = \text{destn}(3, \text{nod } 2)
\]

\[
kk(7) = \text{destn}(1, \text{nod } 3)
\]

\[
kk(8) = \text{destn}(2, \text{nod } 3)
\]

\[
kk(9) = \text{destn}(3, \text{nod } 3)
\]

\[
\ldots \quad \ldots \quad \ldots
\]

\[
kk(3i - 2) = \text{destn}(1, \text{nod } i)
\]

\[
kk(3i - 1) = \text{destn}(2, \text{nod } i)
\]

\[
kk(3i) = \text{destn}(3, \text{nod } i)
\]

For \( i = 1, 6 \)

\[
\text{if } kk(i) > 0;
\]

\[
\text{krow } = kk(i);
\]

\[
R(krow) = R(krow) + r(i)
\]

endif

For \( j = 1, 6 \)

\[
\text{if } kk(j) > 0;
\]

\[
kcol = kk(j);
\]

\[
R(krow, kcol) = R(krow, kcol) + ke(i, j)
\]

\[
\sum_{e=1}^{n_{\text{Elems}}} [k^e]
\]

\section{Solution of Equations}

We have,

\[
[K] \{U\} = \{R\} \tag{3.1}
\]

where \([K]\) – global stiffness matrix

\[
\{R\} – \text{global load vector}
\]

These matrices are obtained by a process called “assembly” that can be mathematically represented as:
\[ \{ R \} = \sum_{e=1}^{nElems} \{ r^e \} \]

This summation implies a special type of matrix addition called ‘assembly’.

Eq (3.1) can be partitioned as:

\[
\begin{bmatrix}
K_{ff} & K_{fr} \\
K_{rf} & K_{rr}
\end{bmatrix}
\begin{bmatrix}
U_f \\
U_r
\end{bmatrix}
= \begin{bmatrix}
R_f \\
R_r
\end{bmatrix} \quad (3.2)
\]

Showing the restrained and free (subscripts \( r \) and \( f \)) degrees of freedom. In general \( \{ U_r \} = 0 \); the null vector equation becomes

\[ [K_{ff}] \{ U_f \} = \{ R_f \} \quad (3.3) \]

Thus only \( K_{ff} \) part of the stiffness matrix; the reduced stiffness matrix, need be stored. \( \{ R_r \} \) are applied loads and \( \{ R_f \} \) are the support reaction. The support reactions are obtained after solving equations above for the unknown free degrees of freedom \( \{ U_f \} \). The element nodal displacements are extracted from \( \{ U \} \) and are used to calculate \( \{ r^e \} \) knowing \( \{ u^e \} \).

\[ [k^e] \{ u^e \} = \{ r^e \} \]

Once the global stiffness matrix and the global load vector have been assembled we arrive at a system of linear algebraic equations as given by \([K_{ff}] \{ U_f \} = \{ R_f \}\)

whose solution gives the unknown free degrees of freedom \( \{ U_f \} \) of the problem for which Gauss elimination method is adopted in the program and outlined in 3.3.

### 3.2  **LINEAR AND QUADRATIC ELEMENTS**

Depending on the profile of the surface; the degree of curve, the analysis can be carried out using linear quadratic hexahedral elements as explained in 2.2.3.
3.3 COMPUTATION OF DEFLECTION

The global stiffness matrix obtained will be symmetric and banded for structural mechanics problems. The symmetry and bandedness of the stiffness matrix can be utilized to save the computational time and memory requirement in the present work. The developed finite element program gives global displacements in the three dimensions as detailed above by Gauss elimination method.[21,84]

```c
void gauss(void)
{
    // Converting the FORTRAN statements from R D Cook et al (p595)
    int lim;
    int semiband;
    semiband = neq;
    // Treat the case of one or more independent equations
    if (semiband <= 1)
    {
        for (i=1; i<=neq; ++i)
            global_disp[i]=global_load[i]/ global_stiffness[neq][1];
        return;
    }

    // Forward reduction of stiffness matrix
    for (n=1; n<=neq-1; ++n)
    {
        lim = min(semiband, neq + 1 - n);
        for (l=2; l<=lim; ++l)
        {
            dum=global_stiffness[n][l]/global_stiffness[n][1];
            i = n + l - 1;
            j = 0;
            for (k=l; k<=lim; ++k)
            {
                j ++;
                global_stiffness[i][j]-=dum*global_stiffness[n][k];
            }
            global_stiffness[n][l] = dum;
        }
    }

    // Forward reduction of load vector
    for (n=1; n<=neq-1; ++n)
    {
        lim = min(semiband, neq + 1 - n);
        for (l=2; l<=lim; ++l)
        {
            i = n + l - 1;
            global_load[i]-= global_stiffness[n][l]*global_load[n];
        }
        global_load[n] = global_load[n]/global_stiffness[n][1];
    }
}
```
global_load[neq] = global_load[neq]/global_stiffness[neq][1];

// Back substitution phase
for (n=neq-1; n>=1; --n)
{
    lim = min(semiband, neq + 1 - n);
    for (l=2; l<=lim; ++l)
    {
        k = n + l - 1;
        global_load[n]-=global_stiffness[n][l] * global_load[k];
    }
    for (i=1; i<=neq; ++i)
    {
        global_disp[i] = global_load[i];
    }
}
return;

3.4 COMPUTATION OF STRESSES AND STRAINS

Once nodal displacements of an element are known, it is possible to obtain the displacements at any point on the element using interpolation. Then, strain at any point on the element can be obtained by making use of the strain-displacement relations, and the stresses by using constitutive relations. As the strain-displacement relation involves partial differentiation and as numerical differentiation always introduces additional errors, the strains; hence the stresses too, are evaluated with less accuracy. As a general observation, it is stated that stresses computed at the nodes are least accurate. On the other hand, in most of the finite element analyses, it is convenient and important to obtain the nodal values of the stresses. Stresses are evaluated with maximum accuracy at a few points on the element called optimal points also known as the Barlow points, which are located in the interior of the element. The stress values at the finite element nodes can be obtained by extrapolating from the values at the optimal sampling points after averaging the values obtained at each node from the neighboring elements. For Isoparametric elements these sampling points are located at the Gauss points of one
order less than that required for full integration of the element stiffness matrix. There exists a standard Gauss rule for each type of element. For example, for a plane four-noded quadrilateral element a 2x2 Gauss rule is sufficient, and for an 8 or 9-noded quadratic element, a 3x3 rule is needed. For these elements, the Barlow points are located at the Gauss points corresponding to 1x1 rules for the four-noded element and 2x2 rules for the 8 or 9-noded element. Stresses at Gauss points can be interpolated or extrapolated to other points in the element.[21,88] This concept is extended to three dimensional isoparametric elements and applied in this project. Since, on finer discretisation the nodal values of stresses show higher values, the elemental stresses are found based on the average value of Gauss points for each element. The stresses at the centroid of the element found by inputting (0,0,0) for the local coordinates are compared with this. The analysis is done with 4x4x4 Gauss rule so that the stress values corresponding to 3x3x3 Gauss points are obtained for each element. Thus, the most optimal 27 stress and strain values for each element are averaged to get more reliable element stresses and strains.

```cpp
// Computation of element stresses at Gauss points
fout << " Sig_x    Sig_y    Sig_z    Sig_xy    Sig_yz    Sig_xz \n   ";
double pxi, pet, pze;
double stress[7], strain[7];
for ( n=1; n<= no_of elems; ++n) {
    for (int ii=1; ii<=20; ++ii) {
        xl[ii] = x[nod[ii][n]];
yl[ii] = y[nod[ii][n]];
zl[ii] = z[nod[ii][n]];
    }
    for (i=1; i<=ngauss-1; ++i) {
        pxi = place[i][ngauss-1];
        for (j=1; j<=ngauss-1; ++j) {
            pet = place[j][ngauss-1];
            for (k=1; k<=ngauss-1; ++k) {
                pze = place[k][ngauss-1];
                quad20_stress (n, pxi, pet, pze); // pxi=pet=pze=0 gives stress at centroid
            }
        }
    }
}
```
fout << "\n" << n << " (" << i << ", " << j << ", " << k << ")"
for (int kk=1; kk<=6; ++kk)
    fout << stress[kk] << " ";
}
}

// Compute strains at element guassian points
fout << " Eps_x  Eps_y  Eps_z  Eps_xy  Eps_yz  Eps_xz\n";
double pxi, pet, pze;
for (n=1; n<=no_of_elems; ++n)
{
    for (int ii=1; ii<=20; ++ii)
    {
        xl[ii] = x[nod[ii][n]];   
        yl[ii] = y[nod[ii][n]]; 
        zl[ii] = z[nod[ii][n]]; 
    }
    for (i=1; i<=ngauss - 1; ++i)
    {
        pxi = place[i][ngauss-1];
        for (j=1; j<=ngauss - 1; ++j)
        {
            pet = place[j][ngauss-1];
            for (k=1; k<=ngauss - 1; ++k)
            {
                pze = place[k][ngauss-1];
                quad20_stress(n, pxi, pet, pze);
            }
        }
    }
}

void quad20_stress (int elem, double pxi, double pet, double pze)
{
    int i, j, k;
    double u[61], DB[7][61];

    //Get element displacement vector
    int dof, dof1, node;
    dof1 = 0;
    for (i=1; i<=20; ++i)
    {
        node = nod[i][elem];
        for (j=1; j<=3; ++j)
        {
            dof1 ++;
            dof = id[j][node];
            if (dof != 0)
            u[dof1] = global_disp[dof];
        }
    }
}
// Get B matrix
  shape_fun (pxi, pet, pze);

// [D]*[B]
for (i=1; i<=6; ++i)
{
  for (j=1; j<=60; ++j)
  {
    DB[i][j] = 0;
    for (k=1; k<=6; ++k)
      DB[i][j] += D[i][k] * B[k][j];
  }
}

// Compute stress as DB * u and strains as B * u
for (i=1; i<=6; ++i)
{
  stress[i] = 0;
  strain[i] = 0;
  for (j=1; j<=60; ++j)
  {
    stress[i] += DB[i][j] * u[j];
    strain[i] += B[i][j] * u[j];
  }
}

3.5 PROGRAM VALIDATION

Standard numerical examples are worked out with the basic program developed as explained above for the three dimensional finite element analysis of continuum using 8-noded and 20-noded hexahedral elements. Output from the Mesh generator program is directly fed as the input to the Main Analysis Program for ‘Nodal Input’ data. This general purpose program for the analysis of 3D solid continuum is applied for the solved problems for nodal loads and boundary conditions.[20,87]

3.5.1 Nodal Load Vector

Straight Cantilever Beam Example 1

A straight cantilever beam of length 1000, thickness 1, depth 100, modulus of elasticity = 200000; (all in consistent units) and Poisson’s ratio = 0.2 is analyzed for
various descretisation using the developed program of 20-noded elements for a vertical point load of 100 units at the free end.[87]

![Cantilever beam; point load at free end](image)

The theoretical solution from strength of materials for the beam is:

Maximum deflection = 2.0, Maximum bending stress at the support; $M/Z = 60$.

The solution arrived from the program is tabulated below in Table 3.1 and is found of very good conformity with the strength of materials solution. Deflected profiles are shown in Fig 3.2 and Fig 3.3.

Table 3.1: Deflection at free end of a cantilever due to vertical point load

<table>
<thead>
<tr>
<th>Discretisation</th>
<th>Number of elements</th>
<th>Number of nodes</th>
<th>Deflection</th>
<th>Max Gauss point stress</th>
<th>Max Gauss point strain</th>
</tr>
</thead>
<tbody>
<tr>
<td>1x1x1</td>
<td>1</td>
<td>20</td>
<td>-1.5343</td>
<td>26.5066</td>
<td>0.000000</td>
</tr>
<tr>
<td>4x1x1</td>
<td>4</td>
<td>56</td>
<td>-1.97449</td>
<td>44.807</td>
<td>0.0002207</td>
</tr>
<tr>
<td>10x1x4</td>
<td>40</td>
<td>353</td>
<td>-2.00584</td>
<td>57.849</td>
<td>0.0002699</td>
</tr>
</tbody>
</table>

![Deflected Profile of Cantilever 40 & 4 elements](image)
The program input and output for 1x1x1 discretisation is outlined below as an example.

```plaintext
if stream ("fem20.inp"); // Output of Mesh Generator Program

1 20 4 // no_of elems >> no_of_nodes >> ngauss;
200000 0.2 0 // E >> po >> gama;

// Nodal coordinates
1 0 0 0 2 0 0.5 0 3 0 1 0 4 0 0 50 5 0 1 50
6 0 0 100 7 0 0.5 100 8 0 1 100 9 500 0 0 10 500 1 0
11 500 0 100 12 500 1 100 13 1000 0 0 14 1000 0.5 0 15 1000 1 0
16 1000 0 50 17 1000 1 50 18 1000 0 100 19 1000 0.5 100 20 1000 1 100

// Nodal connectivity
1 3 1 13 15 8 6 18 20 2 9 14 10 7 11 19 12 5 4 16 17

8 // Nodes at which displacements restricted
1 1 1 1 1 1 1 2 1 1 1 3 1 1 1 4 1 1 1
5 1 1 1 1 1 1 7 1 1 1 8 1 1 1

8 // Concentrated global loads at nodes
13 0 0 -20 14 0 0 0 -5
15 0 0 -20 16 0 0 0 -5
17 0 0 -5 18 0 0 0 -20
19 0 0 -5 20 0 0 0 -20

of stream fout ("fem20.out")

Three Dimensional Problems with 20 Noded Quadratic Brick Elements

Number of elements = 1
Number of nodes = 20
Number of gaussian points = 4
E = 200000 po = 0.2 gama solid = 0 Ebar = 277778

D is:

222222 55555.6 55555.6 0 0 0
55555.6 222222 55555.6 0 0 0
55555.6 55555.6 222222 0 0 0
0 0 0 83333.3 0 0
0 0 0 0 83333.3 0
0 0 0 0 0 83333.3

Nodal Coordinates

Node  X  Y  Z
1   0  0  0
2   0  0.5 0
3   0  1  0
4   0  0  50
5   0  1  50
6   0  0 100
7   0  0.5 100
8   0  1 100
9   500 0  0
10  500 1  0
11  500 0 100

45
<table>
<thead>
<tr>
<th></th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
<th>18</th>
<th>19</th>
<th>20</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>500</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0</td>
<td>0.5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>50</td>
<td>0</td>
<td>50</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0</td>
<td>0.5</td>
<td>100</td>
<td>0</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>0</td>
<td>0.5</td>
<td>0</td>
<td>50</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>100</td>
</tr>
<tr>
<td></td>
<td>1000</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>50</td>
<td>0</td>
<td>100</td>
<td>0</td>
<td>100</td>
</tr>
</tbody>
</table>

**Element Connectivity**

1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  
3  1  13  15  8  6  18  20  2  9  14  10  7  11  19  12  5  4  16  17

**Number of nodes at which displacement is prescribed = 8**

1  1  1  1  2  1  1  1  3  1  1  1  4  1  1  1  5  1  1  1  6  1  1  1  7  1  1  1  8  1  1  1

**ID Array**

0  0  0  0  0  0  0  0  0  1  4  7  10  13  16  19  22  25  28  31  34  
0  0  0  0  0  0  0  0  0  1  2  5  8  11  14  17  20  23  26  29  32  35  
0  0  0  0  0  0  0  0  0  1  3  6  9  12  15  18  21  24  27  30  33  36  

**Number of degrees of freedom = 36**

**Semiband width = 36**

**No of Loaded Joints=8**

13  0  0  -20  14  0  0  -5  15  0  0  -20  16  0  0  -5  17  0  0  -5  18  0  0  -20  19  0  0  -5  20  0  0  -20

**Global Load Array**

<table>
<thead>
<tr>
<th>Node</th>
<th>load-X</th>
<th>load-Y</th>
<th>load-Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>0</td>
<td>0</td>
<td>-20</td>
</tr>
<tr>
<td>14</td>
<td>0</td>
<td>0</td>
<td>-5</td>
</tr>
<tr>
<td>15</td>
<td>0</td>
<td>0</td>
<td>-20</td>
</tr>
<tr>
<td>16</td>
<td>0</td>
<td>0</td>
<td>-5</td>
</tr>
<tr>
<td>17</td>
<td>0</td>
<td>0</td>
<td>-5</td>
</tr>
<tr>
<td>18</td>
<td>0</td>
<td>0</td>
<td>-20</td>
</tr>
<tr>
<td>19</td>
<td>0</td>
<td>0</td>
<td>-5</td>
</tr>
<tr>
<td>20</td>
<td>0</td>
<td>0</td>
<td>-20</td>
</tr>
</tbody>
</table>

**Global Displacement Array**

<table>
<thead>
<tr>
<th>Node</th>
<th>disp-X</th>
<th>disp-Y</th>
<th>disp-Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

46
<table>
<thead>
<tr>
<th>Element Gauss-Points</th>
<th>Sig_x</th>
<th>Sig_y</th>
<th>Sig_z</th>
<th>Sig_xy</th>
<th>Sig_yz</th>
<th>Sig_xz</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 1, 1)</td>
<td>-0.524467518</td>
<td>1.688209784</td>
<td>-1.001190341</td>
<td>0.00975507155</td>
<td>2.7039137</td>
<td></td>
</tr>
<tr>
<td>(1, 1, 2)</td>
<td>0.3690358e-007</td>
<td>-3.14168702e-008</td>
<td>-5.158379e-008</td>
<td>1.214081e-008</td>
<td>0.00975507</td>
<td>-2.649789028</td>
</tr>
<tr>
<td>(1, 1, 3)</td>
<td>2.07448378</td>
<td>0.5244674505</td>
<td>-1.688209816</td>
<td>0.0011903198</td>
<td>0.00975507</td>
<td>-2.703913707</td>
</tr>
<tr>
<td>(1, 2, 1)</td>
<td>-0.524467664</td>
<td>1.68820961</td>
<td>-1.1630845e-008</td>
<td>2.54658516e-006</td>
<td>-0.01140229467</td>
<td>1.075857606</td>
</tr>
<tr>
<td>(1, 2, 2)</td>
<td>0.8406597e-007</td>
<td>2.5872101e-008</td>
<td>4.09301e-008</td>
<td>-7.373196e-008</td>
<td>-0.01140229467</td>
<td>1.129892279</td>
</tr>
<tr>
<td>(1, 2, 3)</td>
<td>0.52446766</td>
<td>1.688209628</td>
<td>-9.4655509e-009</td>
<td>1.455191523e-010</td>
<td>-0.01140229451</td>
<td>1.075857601</td>
</tr>
<tr>
<td>(1, 3, 1)</td>
<td>-0.5244674497</td>
<td>1.688209802</td>
<td>0.0011903177</td>
<td>-0.009755070989</td>
<td>-2.703913533</td>
<td></td>
</tr>
<tr>
<td>(1, 3, 2)</td>
<td>3.411557e-008</td>
<td>4.214072e-009</td>
<td>1.211678e-008</td>
<td>-0.009755070989</td>
<td>-2.64978854</td>
<td></td>
</tr>
<tr>
<td>(1, 3, 3)</td>
<td>0.524467518</td>
<td>-1.688209794</td>
<td>-0.001190338902</td>
<td>-0.009755071063</td>
<td>-2.703913526</td>
<td></td>
</tr>
<tr>
<td>(2, 1, 1)</td>
<td>0.8406597e-007</td>
<td>1.68820957</td>
<td>0.0011903177</td>
<td>-0.009755070989</td>
<td>-2.703913533</td>
<td></td>
</tr>
<tr>
<td>(2, 1, 2)</td>
<td>2.384597e-007</td>
<td>2.5872101e-008</td>
<td>4.09301e-008</td>
<td>-7.373196e-008</td>
<td>-0.01140229467</td>
<td>1.129892279</td>
</tr>
<tr>
<td>(2, 1, 3)</td>
<td>0.52446766</td>
<td>1.688209628</td>
<td>-9.4655509e-009</td>
<td>1.455191523e-010</td>
<td>-0.01140229451</td>
<td>1.075857601</td>
</tr>
<tr>
<td>(2, 2, 1)</td>
<td>-6.06889e-010</td>
<td>5.88626e-011</td>
<td>1.5332807e-009</td>
<td>-8.731194e-011</td>
<td>1.298990622</td>
<td></td>
</tr>
<tr>
<td>(2, 2, 2)</td>
<td>-0.0008991945472</td>
<td>0.11344905e-009</td>
<td>5.31144905e-011</td>
<td>0.01140229451</td>
<td>1.075857606</td>
<td></td>
</tr>
<tr>
<td>(2, 2, 3)</td>
<td>-9.602843e-010</td>
<td>-2.025235e-010</td>
<td>1.198344e-009</td>
<td>-3.229194e-008</td>
<td>-0.01140229451</td>
<td>1.075857699</td>
</tr>
<tr>
<td>(2, 3, 1)</td>
<td>-6.06889e-010</td>
<td>5.88626e-011</td>
<td>1.5332807e-009</td>
<td>-8.731194e-011</td>
<td>1.298990622</td>
<td></td>
</tr>
<tr>
<td>(2, 3, 2)</td>
<td>-9.602843e-010</td>
<td>-2.025235e-010</td>
<td>1.198344e-009</td>
<td>-3.229194e-008</td>
<td>-0.01140229451</td>
<td>1.075857699</td>
</tr>
<tr>
<td>(2, 3, 3)</td>
<td>-0.0008991945472</td>
<td>0.11344905e-009</td>
<td>5.31144905e-011</td>
<td>0.01140229451</td>
<td>1.075857704</td>
<td></td>
</tr>
<tr>
<td>(3, 1, 1)</td>
<td>-5.737255881</td>
<td>0.002988726258</td>
<td>-0.00637928666</td>
<td>-2.703911209</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3, 1, 2)</td>
<td>0.3559575e-007</td>
<td>8.55790e-008</td>
<td>9.003905e-008</td>
<td>-3.230056e-008</td>
<td>-0.00637928666</td>
<td>-2.649876553</td>
</tr>
<tr>
<td>(3, 1, 3)</td>
<td>2.50667557</td>
<td>4.129964178</td>
<td>5.372558993</td>
<td>-0.002988767354</td>
<td>0.006379287134</td>
<td>-2.703911212</td>
</tr>
<tr>
<td>(3, 2, 1)</td>
<td>-5.737255881</td>
<td>0.002988726258</td>
<td>-0.00637928666</td>
<td>-2.703911209</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3, 2, 2)</td>
<td>0.3559575e-007</td>
<td>8.55790e-008</td>
<td>9.003905e-008</td>
<td>-3.230056e-008</td>
<td>-0.00637928666</td>
<td>-2.649876553</td>
</tr>
<tr>
<td>(3, 2, 3)</td>
<td>2.50667557</td>
<td>4.129964178</td>
<td>5.372558993</td>
<td>-0.002988767354</td>
<td>0.006379287134</td>
<td>-2.703911212</td>
</tr>
<tr>
<td>(3, 3, 1)</td>
<td>0.3559575e-007</td>
<td>8.55790e-008</td>
<td>9.003905e-008</td>
<td>-3.230056e-008</td>
<td>-0.00637928666</td>
<td>-2.506676553</td>
</tr>
<tr>
<td>47</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Stresses at Element Gauss Points
<table>
<thead>
<tr>
<th>Element Gauss-Points</th>
<th>Eps_x</th>
<th>Eps_y</th>
<th>Eps_z</th>
<th>Eps_xy</th>
<th>Eps_yz</th>
<th>Eps_xz</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1, 1, 1)</td>
<td>-0.000104887</td>
<td>1.634328e-005</td>
<td>2.9710353e-005</td>
<td>-1.428409e-008</td>
<td>1.170608e-007</td>
<td>-3.244696e-005</td>
</tr>
<tr>
<td>(1, 1, 2)</td>
<td>1.854653e-012</td>
<td>-5.15941e-013</td>
<td>1.456791e-013</td>
<td>1.170608e-007</td>
<td>-3.17985e-005</td>
<td></td>
</tr>
<tr>
<td>(1, 1, 3)</td>
<td>0.000104887</td>
<td>-1.634328e-005</td>
<td>2.9710353e-005</td>
<td>-1.428409e-008</td>
<td>1.170608e-007</td>
<td>-3.244696e-005</td>
</tr>
<tr>
<td>(1, 2, 1)</td>
<td>-0.000104887</td>
<td>1.634328e-005</td>
<td>2.9710353e-005</td>
<td>-1.428409e-008</td>
<td>1.170608e-007</td>
<td>-3.244696e-005</td>
</tr>
<tr>
<td>(1, 2, 2)</td>
<td>-3.447767e-015</td>
<td>7.536337e-015</td>
<td>1.457283e-014</td>
<td>0.1170608e-007</td>
<td>-3.17985e-005</td>
<td></td>
</tr>
<tr>
<td>(1, 2, 3)</td>
<td>0.000104887</td>
<td>-1.634328e-005</td>
<td>2.9710353e-005</td>
<td>-1.428409e-008</td>
<td>1.170608e-007</td>
<td>-3.244696e-005</td>
</tr>
<tr>
<td>(1, 3, 1)</td>
<td>-0.000104887</td>
<td>1.634328e-005</td>
<td>2.9710353e-005</td>
<td>-1.428409e-008</td>
<td>1.170608e-007</td>
<td>-3.244696e-005</td>
</tr>
<tr>
<td>(1, 3, 2)</td>
<td>-1.861580e-012</td>
<td>3.510144e-013</td>
<td>1.454219e-013</td>
<td>0.1170608e-007</td>
<td>-3.17985e-005</td>
<td></td>
</tr>
<tr>
<td>(1, 3, 3)</td>
<td>0.000104887</td>
<td>-1.634328e-005</td>
<td>2.9710353e-005</td>
<td>-1.428409e-008</td>
<td>1.170608e-007</td>
<td>-3.244696e-005</td>
</tr>
</tbody>
</table>

Fig 3.3: Deflected Profile of Cantilever 1 element
**Straight Cantilever Beam Example 2**

A straight cantilever beam of length 200 cm, breadth 20 cm, depth 30 cm, modulus of elasticity = 2000 kN/cm² and Poisson’s ratio = 0.0 is analysed for various discretisation using the developed program of 20 and 8-noded elements as given in Fig 3.4(a) and 3.4 (b).[20]

![Diagram of Straight Cantilever Beam](image)

**Fig 3.4: (a) Straight Cantilever 20 noded 3 elements**

![Diagram of Straight Cantilever Beam](image)

**Fig 3.4: (b) Straight Cantilever 08 noded 5 elements**
**Axial load of 10 kN at free end**

An axial load of 10 kN is applied at the free end of a cantilever beam as shown in Fig 3.5 and analysis done for various discretisation using the developed program and the results compared with the theoretical deflection at free end given by $\frac{PL}{AE}$; Table 3.2.[20] Stresses are given in Table 3.3.

![Cantilever beam; axial load at free end](image)

**Table 3.2: Deflection at free end of a cantilever due to axial load**

<table>
<thead>
<tr>
<th>Hexahedral element chosen</th>
<th>Number of elements</th>
<th>Number of nodes</th>
<th>Deflections at free end in cm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Program</td>
<td>Theory</td>
<td></td>
</tr>
<tr>
<td>8 noded</td>
<td>5</td>
<td>24</td>
<td>0.00166667</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>44</td>
<td>0.00166667</td>
</tr>
<tr>
<td>20 noded</td>
<td>2</td>
<td>32</td>
<td>0.00166667</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>68</td>
<td>0.00166667</td>
</tr>
</tbody>
</table>

**Table 3.3: Stresses at element Gauss points; 20 noded 5 elements**

<table>
<thead>
<tr>
<th>Element</th>
<th>Gauss point</th>
<th>Sig_x</th>
<th>Sig_y</th>
<th>Sig_z</th>
<th>Sig_xy</th>
<th>Sig_yz</th>
<th>Sig_xz</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>(2, 2, 2)</td>
<td>0.01666</td>
<td>1.036e-14</td>
<td>1.76e-14</td>
<td>1.17e-17</td>
<td>-4.74e-18</td>
<td>-1.69e-17</td>
</tr>
<tr>
<td>2</td>
<td>(2, 2, 2)</td>
<td>0.01666</td>
<td>-4.943e-14</td>
<td>-9.21e-14</td>
<td>-6.72e-19</td>
<td>4.97e-18</td>
<td>-8.22e-18</td>
</tr>
<tr>
<td>3</td>
<td>(2, 2, 2)</td>
<td>0.01666</td>
<td>2.004e-13</td>
<td>3.72e-013</td>
<td>-5.64e-18</td>
<td>1.24e-17</td>
<td>-1.20e-17</td>
</tr>
<tr>
<td>4</td>
<td>(2, 2, 2)</td>
<td>0.01666</td>
<td>-6.602e-13</td>
<td>-1.44e-01</td>
<td>-4.66e-18</td>
<td>-2.49e-18</td>
<td>-1.56e-17</td>
</tr>
<tr>
<td>5</td>
<td>(2, 2, 2)</td>
<td>0.01666</td>
<td>3.443e-12</td>
<td>6.19e-12</td>
<td>2.34e-17</td>
<td>1.68e-17</td>
<td>1.49e-18</td>
</tr>
</tbody>
</table>

Stress at fixed end = 0.0166 kN/cm² (theory)
**Point load of 10 kN at free end**

A concentrated load of 10 kN is applied at the free end of a cantilever beam shown in Fig 3.6 in the vertical direction and analysis done for various discretisation using the developed program of 8 and 20 noded elements and the results tabulated in Table 3.4 and 3.5 are compared with the theoretical deflection and stress at free end given by \( \frac{PL^3}{3EI} \). Deflected profile is given in Fig 3.7.

![Cantilever beam with point load at free end](image)

**Fig 3.6: Cantilever beam; point load at free end**

**Table 3.4: Deflection at free end of cantilever due to point load of 10 kN at free end**

<table>
<thead>
<tr>
<th>Hexahedral element chosen</th>
<th>Number of elements</th>
<th>Number of nodes</th>
<th>Deflections at free end in cm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Program</td>
<td>Theory</td>
<td></td>
</tr>
<tr>
<td>8 noded</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-0.158627</td>
<td>-0.296297</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>-0.245152</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>-0.283800</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>-0.289340</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 noded</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>-0.289921</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-0.299275</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Table 3.5: Stresses at element gauss points with 20 noded 5 elements

<table>
<thead>
<tr>
<th>Element No.</th>
<th>Distance from free end</th>
<th>Sig kN/cm²</th>
<th>Sig kN/cm²</th>
<th>Sig kN/cm²</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>180.0</td>
<td>0.598465</td>
<td>3.54059e-006</td>
<td>-6.498908e-006</td>
</tr>
<tr>
<td>2</td>
<td>140.0</td>
<td>0.474524</td>
<td>3.03625e-006</td>
<td>-2.493193e-005</td>
</tr>
<tr>
<td>3</td>
<td>100.0</td>
<td>0.350604</td>
<td>-2.17938e-005</td>
<td>0.000122288</td>
</tr>
<tr>
<td>4</td>
<td>60.0</td>
<td>0.226638</td>
<td>0.000122</td>
<td>-0.000482863</td>
</tr>
<tr>
<td>5</td>
<td>20.0</td>
<td>0.102690</td>
<td>-0.000610</td>
<td>0.001893046</td>
</tr>
</tbody>
</table>

Theoretical maximum stress = 0.6000 kN/cm²

Fig 3.7: Deflected Profile of Cantilever beam 5 elements

Curved Cantilever Beam Example 3

A concentrated load of 10 kN is applied at the free end of a curved cantilever beam Section 20 x 30 cm, radius 100 cm inside shown in Fig 3.8 towards the radial direction and analysis is done using the developed program of 20-noded elements and are compared with the deflection at the free end, as arrived.[21] The maximum deflection from the example is 0.25577 (with error 2.6%) whereas obtained from the program is tabulated below in Table 3.6 and found conforming. Deflected profile is shown in Fig 3.9.
Table 3.6: Deflection at the free end of the curved cantilever

<table>
<thead>
<tr>
<th>Node</th>
<th>Disp-x</th>
<th>Disp-y</th>
<th>Disp-z</th>
</tr>
</thead>
<tbody>
<tr>
<td>73</td>
<td>-0.0653267</td>
<td>-0.2347983</td>
<td>0.0001501</td>
</tr>
<tr>
<td>74</td>
<td>-0.0486148</td>
<td>-0.2515638</td>
<td>7.11e-05</td>
</tr>
<tr>
<td>75</td>
<td>-0.031837</td>
<td>-0.2684533</td>
<td>-2.86e-05</td>
</tr>
<tr>
<td>76</td>
<td>-0.0653232</td>
<td>-0.2345978</td>
<td>-2.72e-14</td>
</tr>
<tr>
<td>77</td>
<td>-0.0318144</td>
<td>-0.2682405</td>
<td>-2.28e-14</td>
</tr>
<tr>
<td>78</td>
<td>-0.0653267</td>
<td>-0.2347983</td>
<td>-0.0001501</td>
</tr>
<tr>
<td>79</td>
<td>-0.0486148</td>
<td>-0.2515638</td>
<td>-7.11e-05</td>
</tr>
<tr>
<td>80</td>
<td>-0.031837</td>
<td>-0.2684533</td>
<td>2.86e-05</td>
</tr>
</tbody>
</table>