1.1 Introduction

A finite element analysis of a deep beam for different mesh discretization is performed here using 1-point and 2×2 Gauss point integration. The effects of mesh properties and integration points are discussed and finally conclusions are drawn for appropriateness of a particular mesh type and general capability of the finite element analyses to simulate the analytical behavior.

The dimensions of the beam and the applied loads are shown in Figure 1.

![Figure 1: Problem statement](image)

Five mesh regimes are used for this problem, which are shown in Figure 2. The nodes and element connectivity files have been provided in file for Quad4 (v) mesh regime.
A program was written in matlab to perform calculations for the response quantities. The program takes input the nodes locations, elements connectivity, loads and boundary conditions and gives output as the nodal and element response quantities. Four functions were written to supplement the code:

1. Stiffness Function to obtain stiffness of an isoparametric element
2. ApplyBC function to enforce boundary conditions
3. Extrapolation function to get nodal results from Gaussian points
4. GetStressStrain function to get stress-strain of elements

The main body of program uses these functions to get the required response quantities. The complete Matlab program is attached in the Appendix.

1.3 Analysis Results

1.3.1 Stiffness Matrices
Ideally, a quad element should have three eigenvalues for three rigid body modes. The stiffness matrix obtained for Quad4 (i) using 1-point gauss integration had more than three zero eigenvalues which suggests that it contains spurious zero energy modes. The stiffness matrix was ill-conditioned for this case and results could not be obtained.

All other stiffness matrices obtained for rest of the cases were found to be ok.

1.3.2 Vertical Deflection
Vertical deflection was obtained for all five mesh regimes using 1-point Gauss integration and 2×2 Gauss integration. The analytical deflection at the tip of the beam is given by the expression

\[ \Delta_{\text{max}} = \frac{1}{3} \frac{PL^3}{EI} + \frac{6}{5} \frac{PL}{GA} \]  

The second part of the above expression \( \frac{6}{5} \frac{PL}{GA} \) is contributed by shear deformation of beam.

Maximum values of the deflection obtained from different cases are summarized in Table 1. Values obtained for 1-point Gauss point integration are the values at the Gauss integration points and have not been extrapolated to the nodes. However, values for 2×2 Gauss integration have been extrapolated using extrapolation matrix and the results are reported for the nodes.

<table>
<thead>
<tr>
<th></th>
<th>Quad1</th>
<th>Quad2</th>
<th>Quad3</th>
<th>Quad4</th>
<th>Quad5</th>
<th>Analytical</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Gauss Point</td>
<td>NaN</td>
<td>2.7089</td>
<td>2.1803</td>
<td>2.7164</td>
<td>2.0851</td>
<td>1.03</td>
</tr>
<tr>
<td>2×2 Gauss</td>
<td>0.843</td>
<td>0.8717</td>
<td>0.8812</td>
<td>0.9682</td>
<td>1.0137</td>
<td>1.03</td>
</tr>
</tbody>
</table>
The verticals deflection of the top surface obtained as function of distance from the fixed support is presented in Figure 3.

![Figure 3: Vertical deflection at the top surface of the beam](image)

(a) 2×2 Gauss integration  
(b) 1-Point Gauss integration

It can be seen that values obtained from 1-point Gauss integration are erroneous compared to 2×2 Gauss integration. Moreover, the Quad4 (v) mesh regime was found to be most accurate for predictions of displacements.

An important conclusion can be drawn by comparing the results of Quad4 (iii) and Quad (iv), where both have same number of meshes but distribution of meshes is different. If total number of meshes is kept same:

- Increasing number of meshes in the vertical direction gives more accurate result for the deflection if 1-point gauss integration is used.
- If 2×2 Gauss integration is used, increasing the number of mesh in horizontal direction gives more accurate prediction of the vertical deflection
1.3.3 Stresses at the Fixed Support

Stresses were calculated at the fixed support. Here values only for $2 \times 2$ Gauss integration are reported.

Variations of normal stresses with height at the fixed obtained for five cases are shown in Figure 4.

![Figure 4: Normal stress ($\sigma_{xx}$) obtained for five mesh regimes](image)

Values of normal stresses are in good agreement, and changing mesh regimes didn’t have significant effect on the normal stress. This is due to the fact that normal stress varies linearly
along the depth of the beam and increasing number of meshes doesn’t have much effect on the result.

Variations of shear stresses along the height of the beam are shown in Figure 5.

![Figure 5: Normal stress (τ_{xy}) obtained for five mesh regimes](image)

The analytical distribution of shear stress along the depth of a rectangular beam is parabolic. It can be seen from the figure above that accuracy of the prediction increases with the number of meshes. The Quad4 (v) case gives the best results out of five cases.

One interesting case can be noted for the cases Quad4 (iii) and Quad (iv) where the total number of meshes are the same but the distribution differs. Increasing number of meshes in horizontal direction gives more accurate prediction of shear stresses when 2×2 Gauss integration is used. The trend is similar to that obtained in deflection.
Appendix A: Main Program

% Program written to calculate stiffness matrix using Gauss-Quadrature
% Written by Manish Kumar for CIE 617: Advance Finite Element Analysis
% Date 10/16/2012

clear;
clc;

% Define material parameter
nu=0.25;
t=1;
E=10000;

% Define Constitutive matrix for plane stress condition
C=(E/(1-nu*nu))*[1 nu 0; nu 1 0; 0 0 (1-nu)/2];

% Load nodes and element connectivity matrix
load Nodes.out; load Elements.out;
N=Nodes(:,2:size(Nodes,2));
E=Elements(:,2:size(Elements,2));
nn=size(N,1); % no of nodes
sdim=size(N,2); % spatial dimension
ne=size(E,1); % no of elements
ns=size(E,2); % no of nodes in the isoparametric element 4-node, 8-node etc

% force vector
f=zeros(2*nn,1);
freeDofs=[18 36]; % dof at which loads are applied
f(freeDofs)=20/size(freeDofs,2); % 20 kips load divided into no of nodes

% boundary conditions
fixedDofs=[1 2 19 20];
fixedDofValues=zeros(1,size(fixedDofs,2))';

outputEle=[1]; % elements where stress-strains are desired
fixedNodes=[10 11 12 13 14 15 16 17 18];

K1G=zeros(2*nn,2*nn);
K2G=zeros(2*nn,2*nn);

for i=1:ne
    X=[];
    for ns1=1:ns
        X=[X;N(E(i,ns1),:)'];
    end
    K1p=stiffness(X,C,t,1);
    K2p=stiffness(X,C,t,2);
    for j=1:ns;
        for p=1:sdim
            for k=1:sdim
                for l=1:sdim
                    K1G(sdim*E(i,j)-sdim+p,2*E(i,k)+l-sdim) = K1G(sdim*E(i,j)-sdim+p,2*E(i,k)+l-sdim)+K1p(sdim*j-sdim+p, sdim*k-sdim+l);
                    K2G(sdim*E(i,j)-sdim+p,2*E(i,k)+l-sdim) = K2G(sdim*E(i,j)-sdim+p,2*E(i,k)+l-sdim)+K2p(sdim*j-sdim+p, sdim*k-sdim+l);
                end
            end
        end
    end
end
%Enforce boundary conditions
[K1G,f1]=applyBC(K1G,f,fixedDofs,fixedDofValues);
[K2G,f2]=applyBC(K2G,f,fixedDofs,fixedDofValues);

U1G=K1G\f1;
U2G=(K2G)\f2;

%Calculate stress at fixed supports
stressGP=[];strainGP=[];
stressNodes=[];strainNodes=[];

for i=outputEle %if output is required over whole beam, iterate from i=1:ne
    x=[];u1=[];u2=[];
    for ns1=1:ns
        x=[x;N(E(i,ns1),:)']
    end

    for ns1=1:ns
        for l=1:sdim
            u1=[u1;U1G(sdim*E(i,ns1)+l-sdim)];
            u2=[u2;U2G(sdim*E(i,ns1)+l-sdim)];
        end
    end

    [stress1 strain1]=getStressStrain(u1,x,C,1);
    [stress2 strain2]=getStressStrain(u2,x,C,2);
    [stress2N strain2N]=Extrapolation(stress2, strain2); %nodal only
determined for 2x2 Gauss

    for ns1=1:ns
        stress1GP(E(i,ns1),:)=stress1';
        strain1GP(E(i,ns1),:)=strain1';
        stress2GP(E(i,ns1),:)=stress2(ns1,:);
        strain2GP(E(i,ns1),:)=strain2(ns1,:);
        stressNodes2(E(i,ns1),:)=stress2N(ns1,:);
        strainNodes2(E(i,ns1),:)=strain2N(ns1,:);
    end
end

%-----------------------------------Results-----------------------------------%

%Calculate stresses at the prescribed nodes, here fixed nodes
stress1=[N(fixedNodes,2) stress1GP(fixedNodes,:)]; %format: [y-coordinate sigmax simgay sigmaxy]
strain1=[N(fixedNodes,2) strain1GP(fixedNodes,:)]; %format: [y-coordinate ex ey exy]

stress2=[N(fixedNodes,2) stressNodes2(fixedNodes,:)]; %format: [y-coordinate sigmax simgay sigmaxy]
strain2=[N(fixedNodes,2) strainNodes2(fixedNodes,:)]; %format: [y-coordinate ex ey exy]

dispTopx=[N(outputNodes,2) U1G(2*outputNodes-1) U2G(2*outputNodes-1)];
dispTopy=[N(outputNodes,1) U1G(2*outputNodes) U2G(2*outputNodes)];
%dispTopx2=[N(outputNodes,2) U2G(2*outputNodes-1)];
%dispTopy2=[N(outputNodes,1) U2G(2*outputNodes)];
%Calculate maximum deflection
w1=mean(U1G(freeDofs));
w2=mean(U2G(freeDofs));
Appendix B: Stiffness

function [K]=stiffness(X, C, t, n)

syms r s;

s1=(1/4)*(1-r)*(1-s);
s2=(1/4)*(1+r)*(1-s);
s3=(1/4)*(1+r)*(1+s);
s4=(1/4)*(1-r)*(1+s);

H=[s1 0 s2 0 s3 0 s4 0; 0 s1 0 s2 0 s3 0 s4];

Xco=H(1,:)*(X);
Yco=H(2,:)*(X);
J=[diff(Xco,r) diff(Yco,r); diff(Xco, s) diff(Yco, s)];

Bu=(J^-1)*[diff(H(1,:),r);diff(H(1,:),s)];
Bv=(J^-1)*[diff(H(2,:),r);diff(H(2,:),s)];
B=[Bu(1,:); Bv(2,:); Bu(2,:)+Bv(1,:)];

F=B'*C*B*det(J)*t;

if n==2
    K=1*subs(F, [r, s], [-0.57735, -0.57735])+1*subs(F, [r, s], [0.57735, -0.57735])+1*subs(F, [r, s], [0.57735, 0.57735])+1*subs(F, [r, s], [-0.57735, 0.57735]);
elseif n==1
    K=2*subs(F, [r, s], [0, 0]);
end
Appendix C: ApplyBC

```matlab
function [K,f]=applyBC(K,f,bcdoef,bcval)

n=length(bcdoef);
sdof=size(K);

for i=1:n
    c=bcdoef(i);
    for j=1:sdof
        K(c,j)=0;
    end
    K(c,c)=1;
    f(c)=bcval(i);
end
```
Appendix D: getStressStrain

function [stress, strain]= getStressStrain(U,X,C,n)

syms r s;

s1=(1/4)*(1-r)*(1-s);
s2=(1/4)*(1+r)*(1-s);
s3=(1/4)*(1+r)*(1+s);
s4=(1/4)*(1-r)*(1+s);

H=[s1 0 s2 0 s3 0 s4 0; 0 s1 0 s2 0 s3 0 s4];

Xco=H(1,:)*(X);
Yco=H(2,:)*(X);

J=[diff(Xco,r) diff(Yco,r); diff(Xco, s) diff(Yco, s)];

Bu=(J^(-1))*[diff(H(1,:),r);diff(H(1,:),s)];
Bv=(J^(-1))*[diff(H(2,:),r);diff(H(2,:),s)];
B=[Bu(1,:); Bv(2,:); Bu(2,:)+Bv(1,:)];

if n==2
    B1=subs(B, [r, s], [-0.57735, -0.57735]);
    B2=subs(B, [r, s], [0.57735, -0.57735]);
    B3=subs(B, [r, s], [0.57735, 0.57735]);
    B4=subs(B, [r, s], [-0.57735, 0.57735]);

    strain(1,:)=(B1*U)'; stress(1,:)=(C*B1*U)';
    strain(2,:)=(B2*U)'; stress(2,:)=(C*B2*U)';
    strain(3,:)=(B3*U)'; stress(3,:)=(C*B3*U)';
    strain(4,:)=(B4*U)'; stress(4,:)=(C*B4*U)';
elseif n==1
    B11=subs(B, [r, s], [0, 0]);
    strain=B11*U; stress=C*B11*U;
end
Appendix E: Extrapolation

% nodal stress-strain from gaussian point stress-strain

function [sigma, e] = Extrapolation(stress, strain)

explmt = [1+sqrt(3)/2   -1/2       1-sqrt(3)/2   -1/2; 
          -1/2       1+sqrt(3)/2   -1/2       1-sqrt(3)/2 ;
         1-sqrt(3)/2   -1/2       1+sqrt(3)/2   -1/2;
        -1/2       1-sqrt(3)/2   -1/2       1+sqrt(3)/2 ];

for i=1:3
    e(:,i) = explmt*strain(:,i) ;
    sigma(:,i) = explmt*stress(:,i) ;
end