Volumetric Locking and its Alleviation in Finite Element Analysis

Department of Mechanical and Industrial Engineering

Indian Institute of Technology Roorkee

Ankur Gaur (19538004)
Arun Kumar (19538005)
Prateek S Pawar (19538009)

Abstract

This project work involves the implementation of finite element methods to alleviate the problem of volumetric locking in structural elements. This included the investigation why and where volumetric locking occurs and what are the methods to reduce it. Because Locking can occur for multiple reasons. In this analysis locking due to the governing equations which are trying to solve are poorly conditioned, which leads to an ill-conditioned system of finite element equations is examined. This analysis is carried on a 2-Dimensional plate element (Plane Stress Condition). Formulation of finite element equations for a 2-Dimensional plate is carried out and using computational tool (MALAB), displacement which is main parameter which get effected due to volumetric locking is examined. Results obtained through computations are compared and analyzed through comparing with analytical solutions. Analytical solutions are obtained using ANSYS software.

Contents

List of Figures

Chapter 1

Introduction

1.1. Hydrostatic State of Stress:

An ideal fluid cannot sustain any shearing forces and the normal force on any surface is compressive in nature. This can be represented by

$$\mathrm{T}^{\mathrm{n}} = -\mathrm{pn}, \qquad \mathrm{p} \geq 0$$

$\forall,\quad T^n =$ Resultant stress vector on any face;

The rectangular components of T^n are obtained by taking the projections of T^n along the *x*, *y* and *z* axes. If n_x, n_y and n_z are the direction cosines of ***n***, then

$$T_x^n = -\mathrm{p}n_x\,; \quad T_y^n = -\mathrm{p}n_y; \quad T_z^n = -\mathrm{p}n_z \qquad \ldots\ldots(1.1)$$

From Cauchy's stress equation, can be written as

$$T_x^n = n_x\sigma_x + n_y\tau_{yx} + n_z\tau_{zx}$$

$$T_y^n = n_x\tau_{xy} + n_y\sigma_y + n_z\tau_{zy}$$

$$T_z^n = n_x\tau_{xz} + n_y\tau_{yz} + n_z\sigma_z$$

Since all shear stress components are zero, then from above equation can be written as

$$T_x^n = n_x\sigma_x\,,\ T_y^n = n_y\sigma_y\,,\quad T_z^n = n_z\sigma_z \qquad \ldots\ldots(1.2)$$

Comparing Equations (1.1) and (1.2)

$$\sigma_x = \sigma_y = \sigma_z = -p$$

Since plane ***n*** was chosen arbitrarily, can be concluded that the resultant stress vector on any plane is normal and is equal to $-p$. This is the type of stress that a small sphere would experience when immersed in a liquid. Hence, the state of stress at a point where the resultant stress vector on any plane is normal to the plane and has the same magnitude is known as a hydrostatic or an isotropic state of stress. The word isotropy means 'independent of orientation' or 'same in all directions'.

An arbitrary state of stress can be resolved into a hydrostatic state and a state of pure shear. Let the given state referred to a coordinate system be

$$[\tau_{ij}] = \begin{bmatrix} \sigma_x & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & \sigma_y & \tau_{yz} \\ \tau_{xz} & \tau_{yz} & \sigma_z \end{bmatrix}$$

Let

$$p = 1/3(\sigma_x + \sigma_y + \sigma_z) \qquad \text{......(1.3)}$$

The given state can be resolved into two different states, as shown:

$$\begin{bmatrix} \sigma_x & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & \sigma_y & \tau_{yz} \\ \tau_{xz} & \tau_{yz} & \sigma_z \end{bmatrix} = \begin{bmatrix} p & 0 & 0 \\ 0 & p & 0 \\ 0 & 0 & p \end{bmatrix} + \begin{bmatrix} \sigma_x - p & \tau_{xy} & \tau_{xz} \\ \tau_{xy} & \sigma_y - p & \tau_{yz} \\ \tau_{xz} & \tau_{yz} & \sigma_z - p \end{bmatrix}$$

The first state on the right-hand side of the above equation is a hydrostatic state. The second state is a state of pure shear since the first invariant for this state is

$$\mathrm{I}_1 = (\sigma_x - p + \sigma_y - p + \sigma_z - p)$$

From equation (1.3) , $\mathrm{I}_1 = 0$;

If the given state is referred to the principal axes, the decomposition into a hydrostatic state and a pure shear state can once again be done as above, i.e.

$$\begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix} = \begin{bmatrix} p & 0 & 0 \\ 0 & p & 0 \\ 0 & 0 & p \end{bmatrix} + \begin{bmatrix} \sigma_1 - p & 0 & 0 \\ 0 & \sigma_2 - p & 0 \\ 0 & 0 & \sigma_3 - p \end{bmatrix}$$

The pure shear state of stress is also known as the deviatoric state of stress or simply as stress deviator.

1.2 Locking Phenomenon:

Determining interpolation functions for displacement field for simple finite element using approximation function or using Lagrange polynomials work well for most applications, but there are situations in which the simple element formulations can give very inaccurate results. In this section illustration of some unexpected difficulties that can arise in apparently perfectly well-designed finite element solutions to boundary value problems will be discussed.

Finite elements are said to “lock” if they exhibit an unphysically stiff response to deformation. Locking can occur for many different reasons. The most common causes include the following: (1) the governing equations which are trying to solve are poorly conditioned, which leads to an ill-conditioned system of finite element equations; (2) the element interpolation functions are unable to approximate accurately the strain distribution in the solid, so the solution converges very slowly as the mesh size is reduced; (3) in certain element formulations (especially beam, plate, and shell elements), displacements and their derivatives are interpolated separately. Locking can occur in these elements if the interpolation functions for displacements and their derivatives are not consistent.

In this analysis locking due to the governing equations which are trying to solve are poorly conditioned, which leads to an ill-conditioned system of finite element equations is examined.

Chapter 2

Governing Equations Formulation

2.1 Problem Statement:

Consider a 2-Dimensional plate as shown:

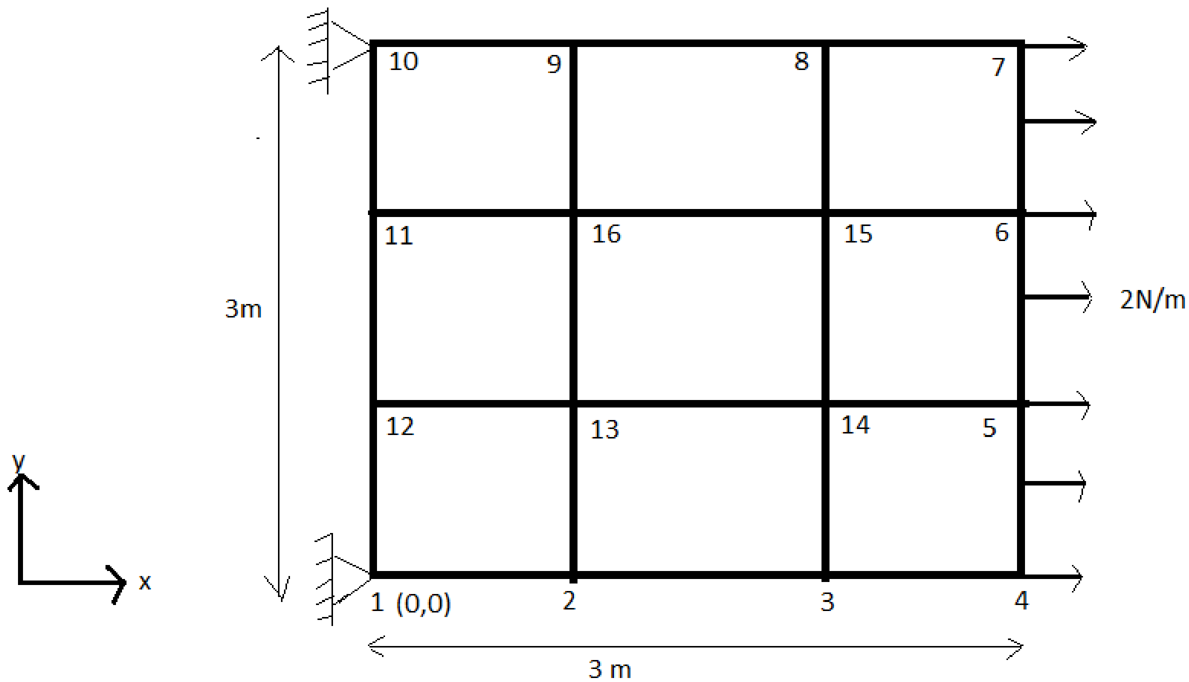

Figure : 1 : 2-D plate with mesh and Loading

Thickness of the element is considered is t = 1mm which allows to treat this problem as plane stress problem. Origin of coordinate system is at node1. Coordinates of other nodes can be calculated.

$$\text{Modulus of Elasticity (E)} = 2\text{x}10^{11} \text{ Pa}$$

$$\text{Poisson Ratio } (\nu) = 0.49999$$

Iso-parametrization of Quadrilateral Element:

Transforming quadrilateral element from *x*-*y* coordinates to $\xi - \eta$ coordinate system as shown:

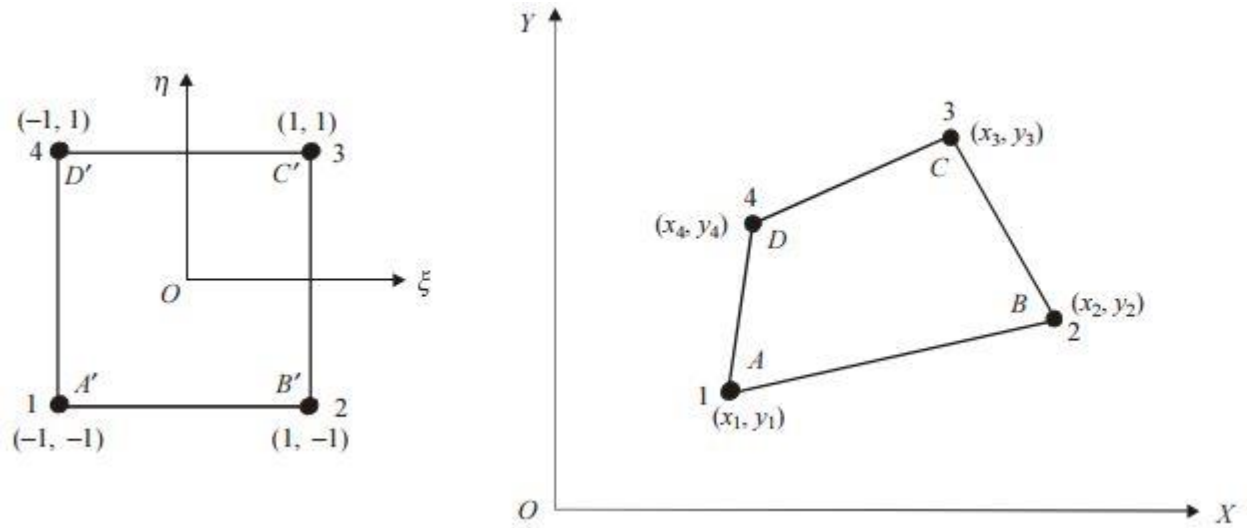

Figure: 2: iso-parametric Element

Interpolation *x*-*y* as function of $\xi - \eta$ as:

$$x = a_0 + a_1\xi + a_2\eta + a_3\xi\eta$$

$$y = a_0 + a_1\xi + a_2\eta + a_3\xi\eta$$

Then can be written as

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 1 & -1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}$$

Or,

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} = \begin{bmatrix} 1 & -1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

Then can be written as

$$y(\xi,\eta) = N_1x_1 + N_2x_2 + N_3x_3 + N_4x_4$$

$$y(\xi,\eta) = N_1x_1 + N_2x_2 + N_3x_3 + N_4x_4$$

Where Interpolation function can written as

$$N_i = \frac{1}{4}[(1+\xi\xi_i)(1+\eta\eta_i)] \quad ; \quad \text{for i = 1 to 4}$$

Then displacement(u) can be written as

$$u = \sum N_i\, u_i$$

$$v = \sum N_i\, v_i \quad ; \ \forall \ u_i\, , v_i = \text{Nodal Displacements in x-y directions}$$

- Brief more on Volumetric Locking:
  When displacements analysed for a 2D or 3D loaded structural element using finite element solution with standard four-noded plane strain quadrilateral element, it is observed that analytical and finite element solution agree well for ν = 0.3, but the finite element solution grossly underestimates the displacements as Poisson's ratio is increased toward 0.5 (material is incompressible limit ν = 0.5) In this limit, the finite element displacements tend to zero, this is known as "volumetric locking."

Chapter 3

3.1 Finite Element Formulation : Fully integration Method

The governing equation for strain energy of any structural problem can be given as:

$$U = \int_v (\varepsilon\sigma) dv$$

Where

$$\{\varepsilon\} = \begin{Bmatrix} \varepsilon_x \\ \varepsilon_y \\ \gamma_{xy} \end{Bmatrix} = \begin{Bmatrix} \dfrac{\partial u}{\partial x} \\ \dfrac{\partial v}{\partial y} \\ \dfrac{\partial u}{\partial y} + \dfrac{\partial v}{\partial x} \end{Bmatrix}$$

$$\sigma = [D]\varepsilon$$

And

$$[D] = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu & 0 \\ \nu & 1 & 0 \\ 0 & 0 & (1-\nu)/2 \end{bmatrix}$$

Here the displacement is function of both iso-parametric coordinates. For any general function f of x and y;

$$\frac{\partial f}{\partial x} = \frac{\partial f}{\partial \xi}\frac{\partial \xi}{\partial x} + \frac{\partial f}{\partial \eta}\frac{\partial \eta}{\partial x}$$

$$\frac{\partial f}{\partial y} = \frac{\partial f}{\partial \xi}\frac{\partial \xi}{\partial y} + \frac{\partial f}{\partial \eta}\frac{\partial \eta}{\partial y}$$

Then

$$\frac{\partial f}{\partial \xi} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial \xi}$$

$$\frac{\partial f}{\partial \eta} = \frac{\partial f}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial f}{\partial y}\frac{\partial y}{\partial \eta}$$

For strains:

$$\begin{Bmatrix} \dfrac{\partial}{\partial \xi} \\ \dfrac{\partial}{\partial \eta} \end{Bmatrix} = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} \begin{Bmatrix} \dfrac{\partial}{\partial x} \\ \dfrac{\partial}{\partial y} \end{Bmatrix}$$

$$\begin{Bmatrix} \dfrac{\partial}{\partial \xi} \\ \dfrac{\partial}{\partial \eta} \end{Bmatrix} = [J] \begin{Bmatrix} \dfrac{\partial}{\partial x} \\ \dfrac{\partial}{\partial y} \end{Bmatrix}$$

Where Jacobian matrix J is defined as

$$[J] = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix}$$

$$[J] = \begin{bmatrix} \sum \dfrac{\partial N_i}{\partial \xi} x_i & \sum \dfrac{\partial N_i}{\partial \xi} y_i \\ \sum \dfrac{\partial N_i}{\partial \eta} x_i & \sum \dfrac{\partial N_i}{\partial \eta} x_i \end{bmatrix}$$

For a 4-Node Quadrilateral Element Jacobian can be written as:

$$[J]=\begin{bmatrix} \left(\frac{1-\eta}{4}\right)(x_2-x_1)+\left(\frac{1+\eta}{4}\right)(x_3-x_4) & \left(\frac{1-\eta}{4}\right)(y_2-y_1)+\left(\frac{1+\eta}{4}\right)(y_3-y_4) \\ \left(\frac{1-\xi}{4}\right)(x_4-x_1)+\left(\frac{1+\xi}{4}\right)(x_3-x_2) & \left(\frac{1-\xi}{4}\right)(y_4-y_1)+\left(\frac{1+\xi}{4}\right)(y_3-y_2) \end{bmatrix}$$

For determining strains can be written as:

$$\begin{Bmatrix} \frac{\partial}{\partial x} \\ \frac{\partial}{\partial y} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{Bmatrix} = \frac{1}{|J|}\begin{bmatrix} J_{22} & -J_{12} \\ -J_{21} & J_{11} \end{bmatrix}\begin{Bmatrix} \frac{\partial}{\partial \xi} \\ \frac{\partial}{\partial \eta} \end{Bmatrix}$$

Then strains can be written as:

$$\{\varepsilon\} = \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial v}{\partial y} \\ \frac{\partial u}{\partial y}+\frac{\partial v}{\partial x} \end{Bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \end{Bmatrix}$$

where,

$$\begin{Bmatrix} \frac{\partial u}{\partial x} \\ \frac{\partial u}{\partial y} \\ \frac{\partial v}{\partial x} \\ \frac{\partial v}{\partial y} \end{Bmatrix} = \begin{bmatrix} \frac{J_{22}}{|J|} & \frac{-J_{12}}{|J|} & 0 & 0 \\ \frac{-J_{21}}{|J|} & \frac{J_{11}}{|J|} & 0 & 0 \\ 0 & 0 & \frac{J_{22}}{|J|} & \frac{-J_{12}}{|J|} \\ 0 & 0 & \frac{-J_{21}}{|J|} & \frac{J_{11}}{|J|} \end{bmatrix} \begin{Bmatrix} \frac{\partial u}{\partial \xi} \\ \frac{\partial u}{\partial \eta} \\ \frac{\partial v}{\partial \xi} \\ \frac{\partial v}{\partial \eta} \end{Bmatrix} = [B_2][B_3]\{\mathrm{u}\}$$

Then

$$\begin{Bmatrix} \frac{\partial u}{\partial \xi} \\ \frac{\partial u}{\partial \eta} \\ \frac{\partial v}{\partial \xi} \\ \frac{\partial v}{\partial \eta} \end{Bmatrix} = \begin{bmatrix} \frac{\partial N_1}{\partial \xi} & 0 & \frac{\partial N_2}{\partial \xi} & 0 & \frac{\partial N_3}{\partial \xi} & 0 & \frac{\partial N_4}{\partial \xi} & 0 \\ \frac{\partial N_1}{\partial \eta} & 0 & \frac{\partial N_2}{\partial \eta} & 0 & \frac{\partial N_3}{\partial \eta} & 0 & \frac{\partial N_4}{\partial \eta} & 0 \\ 0 & \frac{\partial N_1}{\partial \xi} & 0 & \frac{\partial N_2}{\partial \xi} & 0 & \frac{\partial N_3}{\partial \xi} & 0 & \frac{\partial N_4}{\partial \xi} \\ 0 & \frac{\partial N_1}{\partial \eta} & 0 & \frac{\partial N_2}{\partial \eta} & 0 & \frac{\partial N_3}{\partial \eta} & 0 & \frac{\partial N_4}{\partial \eta} \end{bmatrix} \begin{Bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \\ u_4 \\ v_4 \end{Bmatrix} =$$

$[\mathrm{B}_3]\{\mathrm{u}\}$

where

$$\frac{\partial N_1}{\partial \xi} = \frac{-(1-\eta)}{4}\ ;\ \frac{\partial N_2}{\partial \xi} = \frac{(1-\eta)}{4};\frac{\partial N_3}{\partial \xi} = \frac{(1+\eta)}{4}\ ;\ \frac{\partial N_4}{\partial \xi} = \frac{-(1+\eta)}{4}$$

$$\frac{\partial N_1}{\partial \eta} = \frac{-(1-\xi)}{4};\ \frac{\partial N_2}{\partial \eta} = \frac{-(1+\xi)}{4};\ \frac{\partial N_3}{\partial \eta} = \frac{(1+\xi)}{4}\ ;\ \frac{\partial N_4}{\partial \eta} = \frac{(1-\xi)}{4}$$

The above three equations can be combined as:

$$\{\varepsilon\} = [B]\{u\}$$

where, $$[B] = [B_1][B_2][B_3]$$

The stiffness matrix is given as:

$$[k] = \int_v [B]^T[D][B]dv = \iint [B]^T[D][B]tdxdy = \int_{-1}^{1}\int_{-1}^{1}[B]^T[D][B]t|J|d\xi d\eta$$

This integration can be evaluated using 2x2 Gauss Quadrature method. Considering given problem with "$\boldsymbol{\nu}$" as 0.4999 like in rubber material, the result obtained will be as followed:

As from below figure it can be concluded that the displacements obtained using fully integrated elements are less than actual values which is expected because of volumetric locking phenomenon.

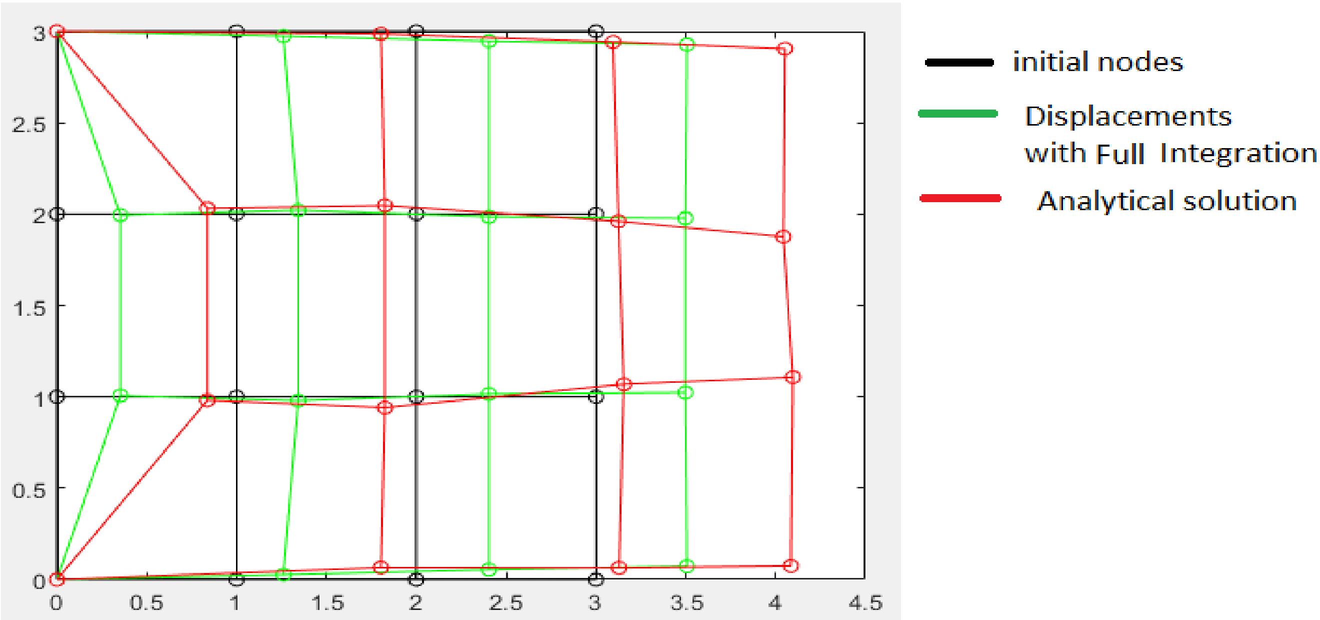

Figure:3: Displacement on Nodes: Full Integration Method

3.2 Finite Element Formulation: Reduced Point Integration

This method is used to reduce effect of volumetric locking by integrating deviatoric part of strain energy with 2x2 point gauss quadrature integration method and volumetric part of strain energy using single gauss point quadrature integration method. Strain energy can be written as

$$\int_v \left(\varepsilon\sigma - \frac{\varepsilon_{qq}\sigma_{kk}}{3}\right) dv + \int_v \left(\frac{\varepsilon_{qq}\sigma_{kk}}{3}\right) dv$$

In above mentioned equation initial term in bracket represents deviatoric part of stress and other term represents volumetric part stress.

Above equation can be formulated as:

$$\int_v \left(N'{}_1{}^T[D_1]N'{}_1 - \frac{1}{3}N'{}_2{}^T[D_2]N'{}_2\right) dv + \int_v \left(\frac{1}{3}N'{}_2{}^T[D_2]N'{}_2\right) dv$$

Here D1, N1 are same as fully integrated method and D2 and N2 are as follows:

$$[D_2] = \frac{E}{1-\nu^2}\begin{bmatrix} 1 & \nu \\ \nu & 1 \end{bmatrix}$$

$$N'_2 = \begin{bmatrix} N_{1\xi} & 0 & N_{2\xi} & 0 & N_{3\xi} & 0 & N_{4\xi} & 0 \\ 0 & N_{1\eta} & 0 & N_{2\eta} & 0 & N_{3\eta} & 0 & N_{4\eta} \end{bmatrix}$$

Solving the given problem using reduced point integration deflection are obtained as shown below:

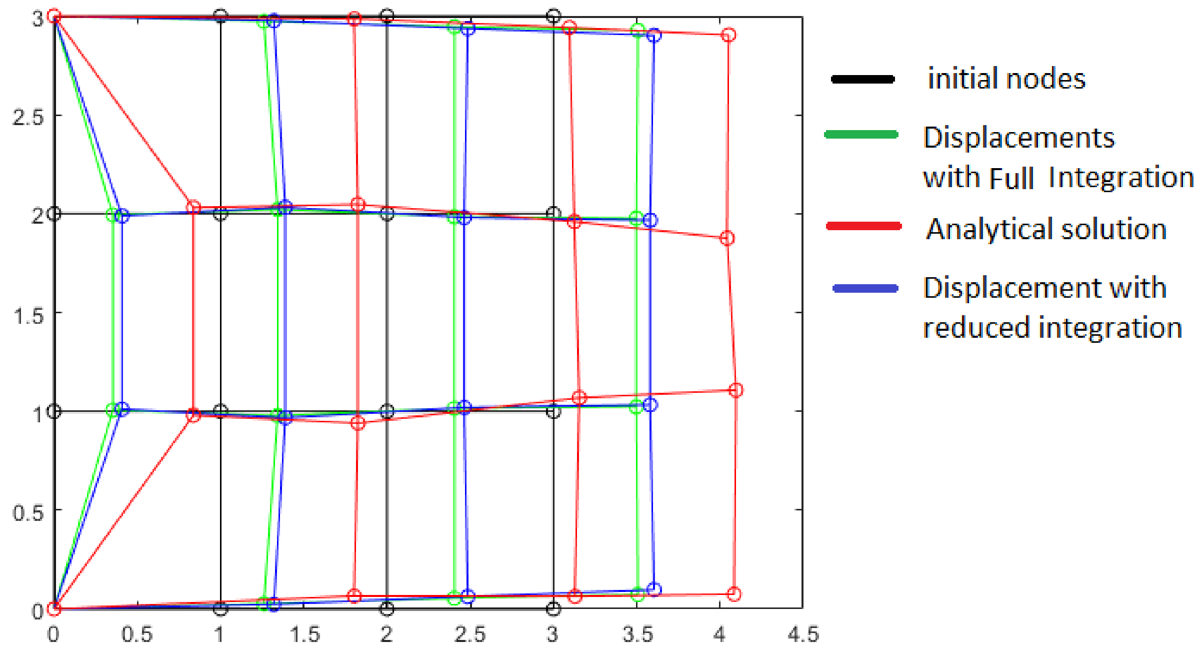

Figure:4 : Displacement on nodes : Reduced Integration Method

References:

[1]. Bower, Allan F, “Applied mechanics of solids”, CRC Press (2010)., Ch. 8, pp. 530-540.

[2]. Seshu, P., “Textbook of Finite Element Analysis ”, PHI Learning Private Limited (2012)., Ch. 4, pp. 145-231.

[2]. Srinath L S, “Advanced Mechanics of solids”, Tata McGraw-Hill (2009), Ch. 1, pp. 1-63.

Annexure 1: Matlab Code for Full integration Method

```
clc
clear all
close all
E=2e11;
nu=0.499;
t=0.001;
r=[1; 2 ;19 ;20];
D3=zeros(3,3);
D3(1,1)=1; D3(1,2)=nu; D3(1,3)=0;
D3(2,1)=nu; D3(2,2)=1; D3(2,3)=0;
D3(3,1)=0; D3(3,2)=0; D3(3,3)=(1-nu)/2;
D3=D3*E/(1-nu*nu);
for i=1:1:3
    for j=1:1:3
        fprintf('i=%d and j=%d',i,j);
        A1=B3((1)/sqrt(3),(1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        A2=B3((1)/sqrt(3),(-1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        A3=B3((-1)/sqrt(3),(1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        A4=B3((-1)/sqrt(3),(-1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);

        jd1=JD((1)/sqrt(3),(1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        jd2=JD((1)/sqrt(3),(-1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        jd3=JD((-1)/sqrt(3),(1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        jd4=JD((-1)/sqrt(3),(-1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);

        k=(A1'*D3*A1)*t*jd1 ...
            +(A2'*D3*A2)*t*jd2 ...
            +(A3'*D3*A3)*t*jd3 ...
            +(A4'*D3*A4)*t*jd4;
K(i,j,:,:)=k(:,:);
        end
end
k1(:,:)=K(1,1,:,:);k2(:,:)=K(2,1,:,:);k3(:,:)=K(3,1,:,:);k4(:,:)=K(1,2,:,:);
k5(:,:)=K(2,2,:,:);k6(:,:)=K(3,2,:,:);k7(:,:)=K(1,3,:,:);k8(:,:)=K(2,3,:,:);
k9(:,:)=K(3,3,:,:);
```

```
Kasse=zeros(32);
Kasse=KA(Kasse,k1,1,2,13,12);
Kasse=KA(Kasse,k2,2,3,14,13);
Kasse=KA(Kasse,k3,3,4,5,14);
Kasse=KA(Kasse,k4,12,13,16,11);
Kasse=KA(Kasse,k5,13,14,15,16);
Kasse=KA(Kasse,k6,14,5,6,15);
Kasse=KA(Kasse,k7,11,16,9,10);
Kasse=KA(Kasse,k8,16,15,8,9);
Kasse=KA(Kasse,k9,15,6,7,8);
Kasse;
Kasse=del(Kasse,r)

 F=zeros(32,1);
 F(7,1)=1.0;
 F(9,1)=2.0;
 F(11,1)=2.0;
 F(13,1)=1.0;
 F=fdel(F,r)
 u=zeros(28,1);
 u=Kasse\F;
 u
 function fnew=fdel(f,r)
 n=size(r);
 for i=1:1:n
     f(r(i)-i+1,:)=[];
  end
fnew=f;
 end
 function knew=del(k,r)
 n=size(r)
 for i=1:1:n
     k(r(i)-i+1,:)=[];
     k(:,r(i)-i+1)=[];
 end
knew=k;
 end
```

```
function ka=KA(Kasse,k,a,b,c,d)
v=[a b c d];
ka=zeros(32);
for i=1:1:4
    for j=1:1:4
        ka(2*v(i)-1:2*v(i),2*v(j)-1:2*v(j))=k(2*i-1:2*i,2*j-1:2*j);
    end
end
ka=ka+Kasse;
end
function jd=JD(zi,it,x1,y1,x2,y2,x3,y3,x4,y4)
J=zeros(2,2);
J(1,1)=(1-it)/4*(x2-x1)+(1+it)/4*(x3-x4);
J(1,2)=(1-it)/4*(y2-y1)+(1+it)/4*(y3-y4);
J(2,1)=(1-zi)/4*(x4-x1)+(1+zi)/4*(x3-x2);
J(2,2)=(1-zi)/4*(y4-y1)+(1+zi)/4*(y3-y2);
jd=det(J);
end
function tb=B3(zi,it,x1,y1,x2,y2,x3,y3,x4,y4)
J=zeros(2,2);
J(1,1)=(1-it)/4*(x2-x1)+(1+it)/4*(x3-x4);
J(1,2)=(1-it)/4*(y2-y1)+(1+it)/4*(y3-y4);
J(2,1)=(1-zi)/4*(x4-x1)+(1+zi)/4*(x3-x2);
J(2,2)=(1-zi)/4*(y4-y1)+(1+zi)/4*(y3-y2);
jd=det(J);
b1=[1, 0, 0 ,0;0, 0, 0, 1;0, 1,1 ,0];
b2=zeros(4,4);
b2(1,1)=J(2,2)/jd;
b2(1,2)=(-1)*J(1,2)/jd;
b2(1,3)=0; b2(1,4)=0;
b2(2,1)=(-1)*J(2,1)/jd;
b2(2,2)=J(1,1)/jd;
b2(2,3)=0; b2(2,4)=0;

b2(3,3)=J(2,2)/jd;
b2(3,4)=(-1)*J(1,2)/jd;
b2(3,1)=0; b2(3,2)=0;
```

```
b2(4,3)=(-1)*J(2,1)/jd;
b2(4,4)=J(1,1)/jd;
b2(4,1)=0; b2(4,2)=0;

b3=zeros(4,8);
dzi=zeros(4,1);
dit=zeros(4,1);

dzi(1)=((-1)*(1-it)/4);
dzi(2)=((1-it)/4);
dzi(3)=((1+it)/4);
dzi(4)=((-1)*(1+it)/4);

dit(1)=((-1)*(1-zi)/4);
dit(2)=((-1)*(1+zi)/4);
dit(3)=((1+zi)/4);
dit(4)=((1-zi)/4);

b3(1,1:2:8)=dzi;
b3(2,1:2:8)=dit;
b3(3,2:2:8)=dzi;
b3(4,2:2:8)=dit;

tb=b1*b2*b3;
end
```

Annexure 2: Matlab Code for Reduced integration Method

```
clc
clear all
close all
E=2e11;
nu=0.499;
t=0.001;
r=[1; 2 ;19 ;20];
D3=zeros(3,3);
D3(1,1)=1; D3(1,2)=nu; D3(1,3)=0;
D3(2,1)=nu; D3(2,2)=1; D3(2,3)=0;
D3(3,1)=0; D3(3,2)=0; D3(3,3)=(1-nu)/2;
D3=D3*E/(1-nu*nu);
D2=zeros(2,2);
D2(1,1)=1; D2(1,2)=nu;
D2(2,1)=nu; D2(2,2)=1;
D2=D2*E/(1-nu*nu);

for i=1:1:3
    for j=1:1:3
        fprintf('i=%d and j=%d',i,j);
        A1=B3((1)/sqrt(3),(1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        A2=B3((1)/sqrt(3),(-1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        A3=B3((-1)/sqrt(3),(1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        A4=B3((-1)/sqrt(3),(-1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);

        b1=B2((1)/sqrt(3),(1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        b2=B2((1)/sqrt(3),(-1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        b3=B2((-1)/sqrt(3),(1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        b4=B2((-1)/sqrt(3),(-1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);

        jd1=JD((1)/sqrt(3),(1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        jd2=JD((1)/sqrt(3),(-1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        jd3=JD((-1)/sqrt(3),(1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
        jd4=JD((-1)/sqrt(3),(-1)/sqrt(3),i-1,j-1,i,j-1,i,j,i-1,j);
```

```
 c=B2(0,0,i-1,j-1,i,j-1,i,j,i-1,j);
       jdc=JD(0,0,i-1,j-1,i,j-1,i,j,i-1,j);

       k=(A1'*D3*A1-b1'*D2*b1/3)*t*jd1 ...
           +(A2'*D3*A2-b2'*D2*b2/3)*t*jd2 ...
           +(A3'*D3*A3-b3'*D2*b3/3)*t*jd3 ...
           +(A4'*D3*A4-b4'*D2*b4/3)*t*jd4 ...
           + (1/3*c'*D2*c*t*jdc*2);
              K(i,j,:,:)=k(:,:);
        end
end
k1(:,:)=K(1,1,:,:);k2(:,:)=K(2,1,:,:);k3(:,:)=K(3,1,:,:);k4(:,:)=K(1,2,:,:);
k5(:,:)=K(2,2,:,:);k6(:,:)=K(3,2,:,:);k7(:,:)=K(1,3,:,:);k8(:,:)=K(2,3,:,:);
k9(:,:)=K(3,3,:,:);
Kasse=zeros(32);

Kasse=KA(Kasse,k1,1,2,13,12);
Kasse=KA(Kasse,k2,2,3,14,13);
Kasse=KA(Kasse,k3,3,4,5,14);
Kasse=KA(Kasse,k4,12,13,16,11);
Kasse=KA(Kasse,k5,13,14,15,16);
Kasse=KA(Kasse,k6,14,5,6,15);
Kasse=KA(Kasse,k7,11,16,9,10);
Kasse=KA(Kasse,k8,16,15,8,9);
Kasse=KA(Kasse,k9,15,6,7,8);
Kasse;

Kasse=del(Kasse,r)

 F=zeros(32,1);
 F(7,1)=1.0;
 F(9,1)=2.0;
 F(11,1)=2.0;
 F(13,1)=1.0;
 F=fdel(F,r)
  u=zeros(28,1);
 u=Kasse\F;
```

```
u
function fnew=fdel(f,r)
n=size(r);
for i=1:1:n
    f(r(i)-i+1,:)=[];
 end
fnew=f;
end

function knew=del(k,r)
n=size(r);
for i=1:1:n
    k(r(i)-i+1,:)=[];
    k(:,r(i)-i+1)=[];
end
knew=k;
end
function ka=KA(Kasse,k,a,b,c,d)
v=[a b c d];
ka=zeros(32);
for i=1:1:4
    for j=1:1:4
        ka(2*v(i)-1:2*v(i),2*v(j)-1:2*v(j))=k(2*i-1:2*i,2*j-1:2*j);
    end
end
ka=ka+Kasse;
end
function jd=JD(zi,it,x1,y1,x2,y2,x3,y3,x4,y4)
J=zeros(2,2);
J(1,1)=(1-it)/4*(x2-x1)+(1+it)/4*(x3-x4);
J(1,2)=(1-it)/4*(y2-y1)+(1+it)/4*(y3-y4);
J(2,1)=(1-zi)/4*(x4-x1)+(1+zi)/4*(x3-x2);
J(2,2)=(1-zi)/4*(y4-y1)+(1+zi)/4*(y3-y2);
jd=det(J);
end

function tb=B3(zi,it,x1,y1,x2,y2,x3,y3,x4,y4)
J=zeros(2,2);
```

```
J(1,1)=(1-it)/4*(x2-x1)+(1+it)/4*(x3-x4);
J(1,2)=(1-it)/4*(y2-y1)+(1+it)/4*(y3-y4);
J(2,1)=(1-zi)/4*(x4-x1)+(1+zi)/4*(x3-x2);
J(2,2)=(1-zi)/4*(y4-y1)+(1+zi)/4*(y3-y2);

jd=det(J);
b1=[1, 0, 0 ,0;0, 0, 0, 1;0, 1,1 ,0];
b2=zeros(4,4);
b2(1,1)=J(2,2)/jd;
b2(1,2)=(-1)*J(1,2)/jd;
b2(1,3)=0; b2(1,4)=0;
b2(2,1)=(-1)*J(2,1)/jd;
b2(2,2)=J(1,1)/jd;
b2(2,3)=0; b2(2,4)=0;

b2(3,3)=J(2,2)/jd;
b2(3,4)=(-1)*J(1,2)/jd;
b2(3,1)=0; b2(3,2)=0;
b2(4,3)=(-1)*J(2,1)/jd;
b2(4,4)=J(1,1)/jd;
b2(4,1)=0; b2(4,2)=0;

b3=zeros(4,8);
dzi=zeros(4,1);
dit=zeros(4,1);

dzi(1)=((-1)*(1-it)/4);
dzi(2)=((1-it)/4);
dzi(3)=((1+it)/4);
dzi(4)=((-1)*(1+it)/4);

dit(1)=((-1)*(1-zi)/4);
dit(2)=((-1)*(1+zi)/4);
dit(3)=((1+zi)/4);
dit(4)=((1-zi)/4);
```

```
b3(1,1:2:8)=dzi;
b3(2,1:2:8)=dit;
b3(3,2:2:8)=dzi;
b3(4,2:2:8)=dit;
tb=b1*b2*b3;
end
function tb=B2(zi,it,x1,y1,x2,y2,x3,y3,x4,y4)
J=zeros(2,2);
J(1,1)=(1-it)/4*(x2-x1)+(1+it)/4*(x3-x4);
J(1,2)=(1-it)/4*(y2-y1)+(1+it)/4*(y3-y4);
J(2,1)=(1-zi)/4*(x4-x1)+(1+zi)/4*(x3-x2);
J(2,2)=(1-zi)/4*(y4-y1)+(1+zi)/4*(y3-y2);
jd=det(J);

b2=zeros(2,4);
b2(1,1)=J(2,2)/jd;
b2(1,2)=(-1)*J(1,2)/jd;
b2(1,3)=0; b2(1,4)=0;

b2(2,3)=(-1)*J(2,1)/jd;
b2(2,4)=J(1,1)/jd;
b2(2,1)=0; b2(2,2)=0;

b3=zeros(4,8);
dzi=zeros(4,1);
dit=zeros(4,1);

dzi(1)=((-1)*(1-it)/4);
dzi(2)=((1-it)/4);
dzi(3)=((1+it)/4);
dzi(4)=((-1)*(1+it)/4);

dit(1)=((-1)*(1-zi)/4);
dit(2)=((-1)*(1+zi)/4);
dit(3)=((1+zi)/4);
dit(4)=((1-zi)/4);
```

```
b3(1,1:2:8)=dzi;
b3(2,1:2:8)=dit;
b3(3,2:2:8)=dzi;
b3(4,2:2:8)=dit;

tb=b2*b3;
end
```