www.ijecs.in
International Journal Of Engineering And Computer Science ISSN:2319-7242
Volume 3 Issue 4 April, 2014 Page No. 5400-5435

An explicit finite element integration scheme using automatic mesh generation technique for linear convex quadrilaterals over plane regions

H.T. Rathod[a*], Bharath Rathod[b], Shivaram K.T[c], H. Y. Shrivalli[d], Tara Rathod[e], K. Sugantha Devi[f]

[a] Department of Mathematics, Central College Campus,
Bangalore University, Bangalore- 560001
E-mail: htrathod2010@gmail.com

[b] Xavier Institute of Management and Entrepreneurship,
Electronic City Phase-II, Hosur Road, Bangalore- 560100
E-mail: rathodbharath@gmail.com

[c] Department of Mathematics, Dayananda Sagar College of Engineering
Shavige Malleswara Hills, Kumar Swamy Layout, Bangalore- 560078
E-mail: shivaramktshiv@gmail.com

[d] Department of Mathematics,
B.M.S college of Engineering,
Basavanagudi, Bangalore-560019,
India
E-mail: hys.maths@bmsce.ac.in

[e] Department of Statistics, Jnana Bharathi Campus,
Bangalore University, Bangalore- 560056
E-mail: rathodtara12@gmail.com

[f] Department of Mathematics, Dr. T. Thimmaiah Institute of Technology, Oorgam Post,
Kolar Gold Field, Kolar District, Karnataka state, Pin- 563120, India.
Email: suganthadevik@yahoo.co.in

Abstract :

This paper presents an explicit finite element integration scheme to compute the stiffness matrices for plane problems using symbolic mathematics. Stiffness matrices are expressed as double integrals of the products of global derivatives over the all quadrilateral plane region. These matrices can be shown to depend on material and geometric properties matrix and the rational functions with polynomial numerators and linear denominator in bivariates over a 2-square. We have computed the integrals of these rational functions over a 2-square by explicit integration using the symbolic mathematics capabilities of MATLAB. The proposed explicit finite element integration scheme is illustrated by computing the Prandtl stress function values and the torisonal constant for the square cross section by using the four node linear convex quadrilateral finite elements.An automatic all quadrilateral mesh generation techniques which is recently proposed by the authours is also integrated in the appended application programs written in MATLAB.

Key words: Explicit Integration, Gauss Legendre Quadrature, Quadrilateral Element, Prandtl's Stress Function for torsion, Symbolic mathematics,all quadrilateral mesh generation technique.

1. Introduction :

In recent years, the finite element method (FEM) has emerged as a powerful tool for the approximate solution of differential equations governing diverse physical phenomena. Today, finite element analysis is an integral and major component in many fields of engineering design and manufacturing. Its use in industry and research is extensive, and indeed without it many practical problem in science, engineering and emerging technologies such as nanotechnology, biotechnology, aerospace, chemical. etc would be incapable of solution [1,2,3]. In FEM, various integrals are to be determined numerically in the evaluation of stiffness matrix, mass matrix, body force vector, etc. The algebraic integration needed to derive explicit finite element relations for second order continuum mechanics problems generally defies our analytic skill and in most cases, it appears to be a prohibitive task. Hence, from a practical point of view, numerical integration scheme is not only necessary but very important as well. Among various numerical integration schemes, Gaussian quadrature, which can evaluate exactly the $(2n-1)^{th}$ order polynomial with n Gaussian integration points, is mostly used in view of the accuracy and efficiency of calculation. However, the integrands of global derivative products in stiffness matrix computations of practical applications are not always simple polynomials but rational expressions which the Gaussian quadrature cannot evaluate exactly [7-15]. The integration points have to be increased in order improve the integration accuracy but it is also desirable to make these evaluations by using as few Gaussian points as possible, from the point of view of the computational efficiency, Thus it is important task to strike a proper balance between accuracy and economy in computation. Therefore analytical integration is essential to generate a smaller error as well as to save the computational costs of Gaussian quadrature commonly applied for science, engineering and technical problems. In explicit integration of stiffness matrix, complications arise from two main sources, firstly the large number of integrations that need to be performed and secondly, in methods which use isoparametric elements, the presence of determinant of the Jacobian matrix (we refer this as Jacobian here after) in the denominator of the element matrix integrands. This problem is considered in the recent work [16] for the linear convex quadrilateral proposes a new discretisation method and use of pre computed universal numeric arrays which do not depend on element size and shape. In this method a linear polygon is discretized into a set of linear triangles and then each of these triangles is further discretised into three linear convex quadrilateral elements by joining the centroid to the mid-point of sides. These quadrilateral elements are then mapped into 2-squares ($-1\leq \xi,\eta \leq 1$) in the natural space (ξ,η) to obtain the same expression of the Jacobian, namely $c(4+\xi+\eta)$ where c is some appropriate constant which depends on the geometric data for the triangle.

In the present paper, we propose a similar discretisation method for linear polygon in Cartesian two space (x,y). This discrtisation is carried in two steps, We first discretise the linear polygon into a set of linear triangles in the Cartesian space (x,y) and these linear triangles are then mapped into a standard triangle in a local space (u,v). We further discretise the standard triangles into three linear quadrilaterals by joining the centroid to the midpoints of triangles in (u,v) space which are finally mapped into 2-square in the local (ξ,η) space. We then establish a derivative product relation between the linear convex quadrilaterals in the Cartesian space, (x,y) which are interior to an arbitrary triangle and the linear quadrilaterals in the local space (u,v) interior to the standard triangle. In this procedure, all computations in the local space (u,v) for product of global derivative integrals are free from geometric properties and hence they are pure numbers. We then propose a numerical scheme to integrate the products of global derivatives. We have shown that the matrix product of global derivative integrals is expressible as matrix triple product comprising of geometric properties matrices and the product of local derivative integrals matrix. We have obtained explicit integration of the product of local derivatives which is now possible by use of symbolic integration commands available in leading mathematical softwares MATLAB, MAPLE, MATHEMATIKA etc. In this paper, we have used the MATLAB symbolic mathematics to compute the integrals of the products of local derivatives in (u, v) space .The proposed explicit integration scheme is shown as a useful technique in the formation of element stiffness matrices for second order boundary problems governed by partial differential equations.

2. Explicit form of the Jacobian For a linear Convex Quadrilateral:

Let us first consider an arbitrary four noded linear convex quadrilateral element in the global Cartesian system (u, v) as in Fig 1a. which is mapped in to a 2-square in the local parametric system (ξ,η) as in Fig 1b. is given by

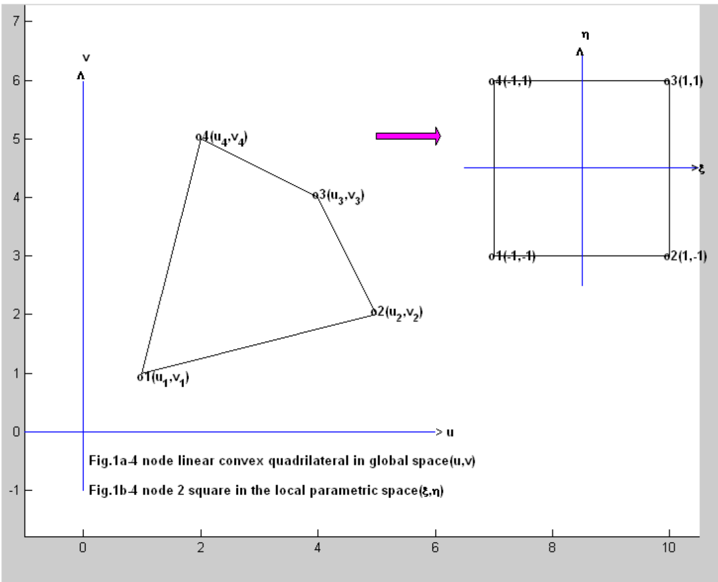


Fig.1a-4 node linear convex quadrilateral in global space(u,v)
Fig.1b-4 node 2 square in the local parametric space(ξ,η)

$\begin{pmatrix} u \\ v \end{pmatrix} = \sum_{k=1}^{4} \begin{pmatrix} u_k \\ v_k \end{pmatrix} \; N_k(\xi, \eta)$ ------------------------ (1)

Where (u_k, v_k) , (k=1,2,3,4) are the vertices of the quadrilateral element in (u, v) plane and $N_k(\xi, \eta)$ denotes the shape function of node k, and they are expressed as [1-3]

$N_k(\xi, \eta) = \frac{1}{4}\left(1 + \xi_k \xi\right)\left(1 + \eta_k \eta\right)$ ----------------------- (2a)

Where $\{ (\xi_k, \eta_k), k = 1,2,3,4\}$ = {(-1,-1),(1,-1),(1,1),(-1,1)} ----------------------- (2b)

From the Eq.(1) and Eq.(2), we have

$\frac{\partial u}{\partial \xi} = \sum_{k=1}^{4} u_k \frac{\partial N_k}{\partial \xi} = \frac{1}{4}[\,(-u_1 + u_2 + u_3 - u_4) + (u_1 - u_2 + u_3 - u_4)\,\eta\,]$ ------------ (3a)

$\frac{\partial u}{\partial \eta} = \sum_{k=1}^{4} u_k \frac{\partial N_k}{\partial \eta} = \frac{1}{4}[\,(-u_1 - u_2 + u_3 + u_4) + (u_1 - u_2 + u_3 - u_4)\,\xi\,]$ ------------ (3b)

Similarly

$\frac{\partial v}{\partial \xi} = \frac{1}{4}[\,(-v_1 + v_2 + v_3 - v_4) + (v_1 - v_2 + v_3 - v_4)\,\eta\,]$ ------------ (3c)

$\frac{\partial v}{\partial \eta} = \frac{1}{4}[\,(-v_1 - v_2 + v_3 + v_4) + (v_1 - v_2 + v_3 - v_4)\,\xi\,]$ ------------ (3d)

Hence the Jacobian, J can be expressed as [1, 2, 3]

$J = \frac{\partial(u,v)}{\partial(\xi,\eta)} = \frac{\partial u}{\partial \xi}\frac{\partial v}{\partial \eta} - \frac{\partial u}{\partial \eta}\frac{\partial v}{\partial \xi} = \alpha + \beta\xi + \gamma\,\eta$ -------------- (4a)

Where

$\alpha = \frac{1}{8}[\,(u_4 - u_2)(v_1 - v_3) + (u_3 - u_1)(v_4 - v_2)\,]$

$\beta = \frac{1}{8}[\,(u_4 - u_3)(v_2 - v_1) + (u_1 - u_2)(v_4 - v_3)\,]$

$\gamma = \frac{1}{8}[\,(u_4 - u_1)(v_2 - v_3) + (u_3 - u_2)(v_4 - v_1)\,]$ -------------------- (4b)

3. Global Derivatives:

If N_i denotes the basis functions of node i of any order of the element e, then the chain rule of differentiation from Eq.(1) we can write the global derivative as in [1, 2, 3]

$\begin{pmatrix} \frac{\partial N_i}{\partial u} \\ \frac{\partial N_i}{\partial v} \end{pmatrix} = \frac{1}{J}\begin{bmatrix} \frac{\partial v}{\partial \eta} & -\frac{\partial v}{\partial \xi} \\ -\frac{\partial u}{\partial \eta} & \frac{\partial u}{\partial \xi} \end{bmatrix}\begin{bmatrix} \frac{\partial N_i}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} \end{bmatrix}$ ---------------------------- (5)

Where $\frac{\partial u}{\partial \xi}, \frac{\partial u}{\partial \eta}, \frac{\partial v}{\partial \xi}$ and $\frac{\partial v}{\partial \eta}$ are defined as in Eqs.(3a)–(3d) while J is defined in Eq.(4) , (i, j = 1,2,3,. . . . . . . , nde) , nde = the number of nodes per element

4. Discretisation of an arbitrary triangle:

A linear convex polygon in the (x, y) plane can be always discretised into a finite number of linear triangles. However, we would like to study a particular discretization of these triangles further into linear convex quadrilaterals. This is stated in the following Lemma [6].

Lemma 1. Let Δ PQR be an arbitrary triangle with the vertices $P(x_p, y_p)$, $Q(x_q, y_q)$ and $R(x_r, y_r)$ and S, T, U be the midpoints of sides PQ, QR and RP respectively and let Z be its centroid. We can obtain three linear convex quadrilaterals ZTRU, ZUPS and ZSQT from triangle Δ PQR as shown in Fig2. If we map each of these quadrilaterals into 2-squares in which the nodes are oriented in counter clockwise from Z then Jacobian J for each element e is given by

$$J = J^e = \frac{1}{48} \Delta\, pqr\, (4 + \xi + \eta), \qquad e = 1,2,3 \qquad \text{--------------- (6)}$$

Where Δ pqr is the area of the triangle Δ PQR

$$2\Delta\, pqr = \begin{vmatrix} 1 & x_p & y_p \\ 1 & x_q & y_q \\ 1 & x_r & y_r \end{vmatrix} = \left[(x_p - x_r)(y_q - y_r) - (x_q - x_r)(y_p - y_r) \right] \qquad \text{-------------- (7)}$$

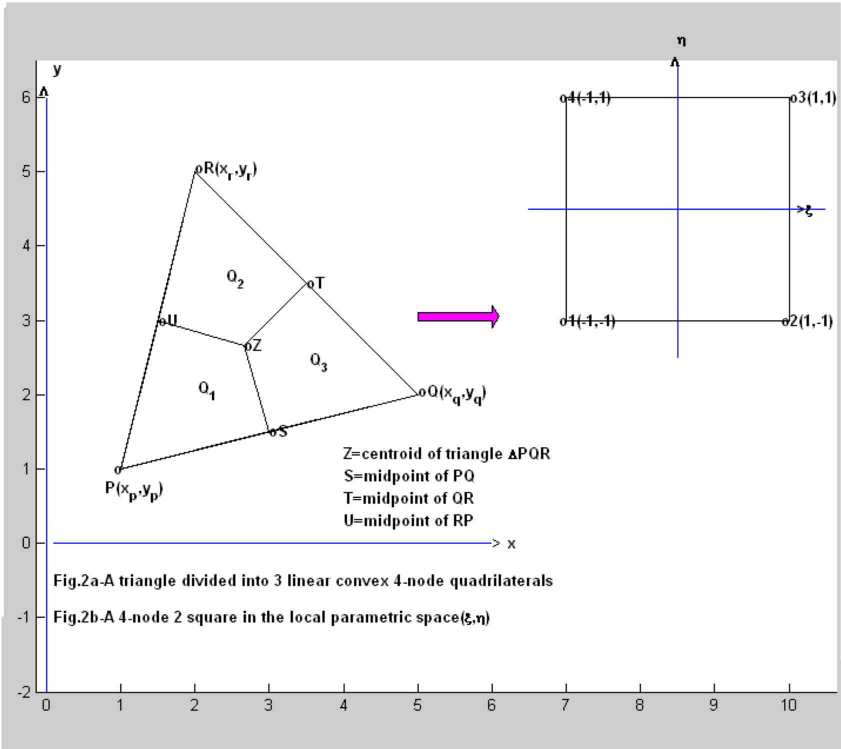


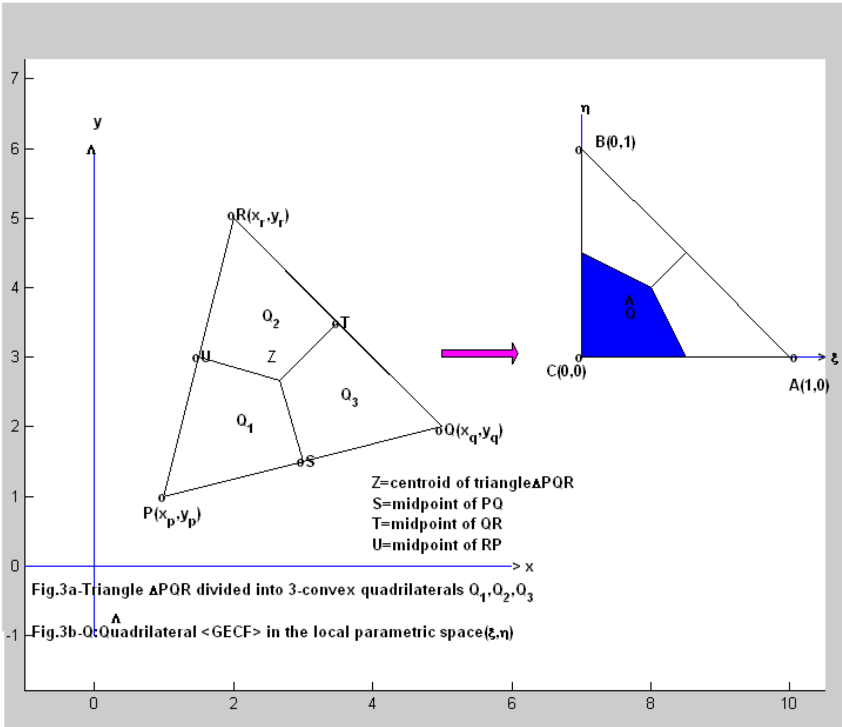
Fig.2a-A triangle divided into 3 linear convex 4-node quadrilaterals

Fig.2b-A 4-node 2 square in the local parametric space(ξ,η)

Proof : Proof is straight forward and given in [16]

We now prove a new result in the form of following Lemma :

Lemma 2. Let Δ PQR be an arbitrary triangle with the vertices $P(x_p, y_p)$, $Q(x_q, y_q)$ and $R(x_r, y_r)$, let S, T, U be the midpoints of sides PQ, QR, and RP and let Z be the centroid of Δ PQR, Then we obtain three quadrilaterals Q_1, Q_2, Q_3 spanning the vertices <ZUPS> , <ZSQT> and <ZTRU> . these quadrilaterals can be mapped into the quadrilateral spanning vertices GECF with G(1/3, 1/3), E(0, ½) , C(0, 0) and F(1/2, 0) of the right isosceles triangle Δ ABC with spanning vertices A(1, 0), B(0, 1) and C(0, 0) in the (u, v) space as shown in Fig 3a and Fig 3b



Proof : The sum of the quadrilaterals $Q_1 + Q_2 + Q_3 = \Delta$ PQR as shown in Fig 2a & Fig 3a. The linear transformations

$$\begin{pmatrix} x^{(1)} \\ y^{(1)} \end{pmatrix} = \begin{pmatrix} x_p \\ y_p \end{pmatrix} w + \begin{pmatrix} x_q \\ y_q \end{pmatrix} u + \begin{pmatrix} x_r \\ y_r \end{pmatrix} v \qquad \text{---------------------------- (8)}$$

$$\begin{pmatrix} x^{(2)} \\ y^{(2)} \end{pmatrix} = \begin{pmatrix} x_q \\ y_q \end{pmatrix} w + \begin{pmatrix} x_r \\ y_r \end{pmatrix} u + \begin{pmatrix} x_p \\ y_p \end{pmatrix} v \qquad \text{--------------------------- (9)}$$

$$\begin{pmatrix} x^{(3)} \\ y^{(3)} \end{pmatrix} = \begin{pmatrix} x_r \\ y_r \end{pmatrix} w + \begin{pmatrix} x_p \\ y_p \end{pmatrix} u + \begin{pmatrix} x_q \\ y_q \end{pmatrix} v \qquad \text{------------------------- (10)}$$

with $w = 1 - u - v$ ------------------------- (11)

map the arbitrary triangle Δ PQR into a right isosceles triangle A(1, 0) , B(0, 1) and C(0, 0) in the uv–plane We can now verify that quadrilateral Q_1 spanned by vertices Z$\left(\frac{x_p+x_q+x_r}{3}, \frac{y_p+y_q+y_r}{3}\right)$, U$\left(\frac{x_r+x_p}{2}, \frac{y_r+y_q}{2}\right)$, P(x_p, y_p) ,S$\left(\frac{x_p+x_q}{2}, \frac{y_p+y_q}{2}\right)$ in xy- plane is mapped into the quadrilateral spanning the vertices G(1/3, 1/3) , E(0, ½), C(0, 0) and F(1/2, 0) by use of the transformation given in Eq.(8),

Similarly, we see that the quadrilateral Q_2 spanned by vertices Z,S,Q,T is mapped into the quadrilateral spanned by vertices G(1/3, 1/3), E(0, ½), C(0, 0) and F(½, 0) by use of the transformation of Eq.(9), Finally the quadrilateral Q_3 in the xy- plane is mapped into the quadrilateral GECF in uv- plane by use of the linear transformation of Eq.(10),

This completes the proof.

We have shown in the present section that an arbitrary triangle can be discretised into three linear convex quadrilaterals. Further, each of these quadrilaterals can be mapped into a unique quadrilateral in uv-plane spanned by vertices (1/3, 1/3), (0, ½), (0, 0) and (½, 0) .

5. Composite Integration over an arbitrary triangle:

We shall now establish a composite integration formula for an arbitrary triangular region Δ PQR shown in Fig 2a or Fig 3a We have for an arbitrary smooth function $\phi(x, y)$

$$II_{\Delta PQR} = \iint_{\Delta PQR} \phi(x, y)\, dxdy = \sum_{e=1}^{3} \iint_{Q_e} \phi(x, y)\, dxdy \qquad \text{--------------------- (12)}$$

$= \iint_{\hat{Q}} \sum_{e=1}^{3}[\phi\,(x^{(e)}(u,v), y^{(e)}(u,v)) \frac{\partial\left(x^{(e)}(u,v), y^{(e)}(u,v)\right)}{\partial(u,v)}]\, dudv$

$= (2\,\Delta_{pqr}) \iint_{\hat{Q}} \{ \sum_{e=1}^{3}[\phi\,(x^{(e)}(u,v),\ y^{(e)}(u,v))\,]\}\, dudv$ -------------------- (13)

Where $(x^{(e)}(u,v),\ y^{(e)}(u,v)), e = 1,2,3)$ are the transformations of Eqs.(8)–(10) and $\hat{Q}$ is the quadrilateral in uv- plane spanned by vertices G(1/3 ,1/3) , E(0, ½) , C(0, 0) and F(1/2,0) , and Δ_{pqr} is the area of triangle Δ PQR, Now using the transformations defined in Eqs.(1)–(2) we obtain

$II_{\Delta PQR} = (2\,\Delta_{pqr}) \iint_{\hat{Q}} \{ \sum_{e=1}^{3}[\phi\,(x^{(e)}(u,v),\ y^{(e)}(u,v)) \frac{\partial(u,v)}{\partial(\xi,\eta)}\}\, d\xi\, d\eta$ --------------- (14)

In Eq.(14) we have used the transformation

$u(\xi,\ \eta) = \frac{1}{3}N_1(\xi,\ \eta) + \frac{1}{2}N_4(\xi,\ \eta)$

$v(\xi,\ \eta) = \frac{1}{3}N_1(\xi,\ \eta) + \frac{1}{2}N_2(\xi,\ \eta)$ ---------------------- (15)

to map the quadrilateral $\hat{Q}$ into a 2 – square in $\xi\eta$ – plane.

We can now obtain from Eqs.(14)–(15)

$II_{\Delta PQR} = (2\,\Delta_{pqr}) \int_{-1}^{1}\int_{-1}^{1} [\sum_{e=1}^{3} \left(\frac{4+\xi+\eta}{96}\right)\phi(x^{(e)}(u,v),\ y^{(e)}(u,v))\,]d\xi\, d\eta$ ------------- (16)

We can evaluate Eq.(16) either analytically or numerically depending on the form of the integrand.

Using Numerical Integration ;

$II_{\Delta PQR}=2\Delta_{pqr} \sum_{i=1}^{N}\sum_{j=1}^{N}\left(\frac{W_i^{(N)}W_j^{(N)}(4+\xi_i^{(N)}+\eta_j^{(N)})}{96}\right) \sum_{e=1}^{3} \phi(x^{(e)}\left(u_{i,j}^{(N)}, v_{i,j}^{(N)}\right), y^{(e)}\left(u_{i,j}^{(N)}, v_{i,j}^{(N)}\right))$

----------------------- (17)

Where,

$u_{i,j}^{(N)} = u(\xi_i^{(N)},\ \eta_j^{(N)})$ and $v_{i,j}^{(N)} = v(\xi_i^{(N)},\ \eta_j^{(N)})$ ---------------------- (18)

and $(W_i^{(N)},\ \xi_i^{(N)})$, $(W_j^{(N)},\ \xi_j^{(N)})$ are the weight coefficients and sampling points of N^{th} order Gauss Legendre Quadrature rules.

The above composite rule is applied to numerical Integration over polygonal domains using convex quadrangulation and Gauss Legendre Quadrature Rules[27].

6. Global Derivative Integrals:

If $N_i^{(e)}$ denotes the basis function for node i of element e , then by chain rule of partial differentiation

$$\begin{pmatrix} \frac{\partial N_i^e}{\partial x} \\ \frac{\partial N_i^e}{\partial y} \end{pmatrix} = \begin{bmatrix} \frac{\partial u}{\partial x} & \frac{\partial v}{\partial x} \\ \frac{\partial u}{\partial y} & \frac{\partial u}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial N_i^e}{\partial u} \\ \frac{\partial N_i^e}{\partial v} \end{bmatrix} \quad \text{-------------------------- (19)}$$

We note that to transform $Q_e(e = 1,2,3)$ of ΔPQR in Cartesian space (x,y) into $\hat{Q}$, the quadrilateral spanned by vertices (1/3,1/3), (0,½), (0,0) and (1/2,0) in uv-plane we must use the earlier transformations.

$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_p \\ y_p \end{pmatrix} + \begin{pmatrix} x_q - x_p \\ y_q - y_p \end{pmatrix} u + \begin{pmatrix} x_r - x_p \\ y_r - y_p \end{pmatrix} v$ for Q_1 in ΔPQR --------------------------- (8)

$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_q \\ y_q \end{pmatrix} + \begin{pmatrix} x_r - x_q \\ y_r - y_q \end{pmatrix} u + \begin{pmatrix} x_p - x_q \\ y_p - y_q \end{pmatrix} v$ for Q_2 in ΔPQR -------------------------- (9)

$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_r \\ y_r \end{pmatrix} + \begin{pmatrix} x_p - x_r \\ y_p - y_r \end{pmatrix} u + \begin{pmatrix} x_q - x_r \\ y_q - y_r \end{pmatrix} v$ for Q_3 in ΔPQR ------------------------ (10)

and the above transformations viz Eqs.(8)-(10) are of the form

$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x_c \\ y_c \end{pmatrix} + \begin{pmatrix} x_a - x_c \\ y_a - y_c \end{pmatrix} u + \begin{pmatrix} x_b - x_c \\ y_b - y_c \end{pmatrix} v$ ------------------------ (20)

which can map an arbitrary triangle ΔABC , A(x_a ,y_a) ,B(x_b ,y_b) , C(x_c , y_c) in xy – plane into a right isosceles triangle in the uv – plane

Hence, we have

$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} (x_a - x_c) & (x_b - x_c) \\ (y_a - y_c) & (y_b - y_c) \end{pmatrix}^{-1} \begin{pmatrix} x - x_c \\ y - y_c \end{pmatrix}$ -------------------------- (21)

This gives

$u = (\alpha_a + \beta_a x + \gamma_a y)/(2\,\Delta_{abc})$

$v = (\alpha_b + \beta_b x + \gamma_b y)/(2\,\Delta_{abc})$ ------------------------ (22)

$\alpha_a = (x_b y_c - x_c y_b)$, $\alpha_b = (x_c y_a - x_a y_c)$,

$\beta_a = (y_b - y_c)$, $\beta_b = (y_c - y_a)$,

$\gamma_a = (x_c - x_b)$, $\gamma_b = (x_a - x_c)$,

$\frac{\partial(x,y)}{\partial(u,v)} = 2\Delta_{abc} = \begin{vmatrix} 1 & x_a & y_a \\ 1 & x_b & y_b \\ 1 & x_c & y_c \end{vmatrix} = 2 * \text{area of the triangle } \Delta ABC$

$= (\gamma_b \beta_a - \gamma_a \beta_b)$ ------------------- (23)

Hence from Eq.(19) and Eq.(22), we obtain

$$\begin{pmatrix} \frac{\partial N_i^e}{\partial x} \\ \frac{\partial N_i^e}{\partial y} \end{pmatrix} = \begin{pmatrix} \frac{\beta_a}{2\Delta_{abc}} & \frac{\beta_b}{2\Delta_{abc}} \\ \frac{\gamma_a}{2\Delta_{abc}} & \frac{\gamma_b}{2\Delta_{abc}} \end{pmatrix} \begin{pmatrix} \frac{\partial N_i^e}{\partial u} \\ \frac{\partial N_i^e}{\partial v} \end{pmatrix}$$

$$= \begin{pmatrix} \beta_a{}^* & \beta_b{}^* \\ \gamma_a{}^* & \gamma_b{}^* \end{pmatrix} \begin{pmatrix} \frac{\partial N_i^e}{\partial u} \\ \frac{\partial N_i^e}{\partial v} \end{pmatrix} \quad \text{------------------------------ (24)}$$

where $\beta_a{}^* = \frac{\beta_a}{(2\Delta_{abc})}$, $\beta_b{}^* = \frac{\beta_b}{(2\Delta_{abc})}$

$\gamma_a{}^* = \frac{\gamma_a}{(2\Delta_{abc})}$, $\gamma_b{}^* = \frac{\gamma_b}{(2\Delta_{abc})}$ --------------------------- (25)

Letting,

$$D_{x,y}^{i,e} = \begin{pmatrix} \frac{\partial N_i^e}{\partial x} \\ \frac{\partial N_i^e}{\partial y} \end{pmatrix} , \; P = \begin{pmatrix} \beta_a{}^* & \beta_b{}^* \\ \gamma_a{}^* & \gamma_b{}^* \end{pmatrix} , \; D_{u,v}^{i,e} = \begin{pmatrix} \frac{\partial N_i^e}{\partial u} \\ \frac{\partial N_i^e}{\partial v} \end{pmatrix} \quad \text{---------------------------- (26)}$$

We obtain from Eq.(24),

$D_{x,y}^{i,e} = P\, D_{u,v}^{i,e}$ ----------------------------- (27)

So that from Eq.(26) and Eq.(27) , we obtain

$$G_{x,y}^{i,j,e} = \begin{pmatrix} \frac{\partial N_i^e}{\partial x} \\ \frac{\partial N_i^e}{\partial y} \end{pmatrix} \left(\frac{\partial N_j^e}{\partial x} \;\; \frac{\partial N_j^e}{\partial y} \right) = (D_{x,y}^{i,e})\,(D_{x,y}^{j,e})^T$$

$$= \begin{pmatrix} \frac{\partial N_i^e}{\partial x}\frac{\partial N_j^e}{\partial x} & \frac{\partial N_i^e}{\partial x}\frac{\partial N_j^e}{\partial y} \\ \frac{\partial N_i^e}{\partial y}\frac{\partial N_j^e}{\partial x} & \frac{\partial N_i^e}{\partial y}\frac{\partial N_j^e}{\partial y} \end{pmatrix} \quad \text{----------------------------- (28 a)}$$

$$G_{u,v}^{i,j,e} = \begin{pmatrix} \frac{\partial N_i^e}{\partial u} \\ \frac{\partial N_i^e}{\partial v} \end{pmatrix} \left(\frac{\partial N_j^e}{\partial u} \;\; \frac{\partial N_j^e}{\partial v} \right) = (D_{u,v}^{i,e})\,(D_{u,v}^{j,e})^T$$

$$= \begin{pmatrix} \frac{\partial N_i^e}{\partial u}\frac{\partial N_j^e}{\partial u} & \frac{\partial N_i^e}{\partial u}\frac{\partial N_j^e}{\partial v} \\ \frac{\partial N_i^e}{\partial v}\frac{\partial N_j^e}{\partial u} & \frac{\partial N_i^e}{\partial v}\frac{\partial N_j^e}{\partial v} \end{pmatrix} \quad \text{----------------------------- (28 b)}$$

We have now from Eq.(27) and Eq.(28)

$G_{x,y}^{i,j,e} = (P\, D_{u,v}^{i,e})\,\left(\left(D_{u,v}^{j,e}\right)^T P^T\right)$

$= P\, G_{u,v}^{i,j,e}\, P^T$ --------------------------- (29)

We now define the submatrices of global derivative integrals in (x,y) and (u,v) space associated with the nodes i and j as ;

$S^{i,j,e} = \iint_{Q_e} G_{x,y}^{i,j,e}\; dx\,dy$, (e=1,2,3) --------------------------- (30) $K^{i,j,e} = \iint_{\hat{Q}} G_{u,v}^{i,j,e}\; du\,dv$ --------------------------- (31)

where, we have already defined the quadrilaterals Q_e (e=1,2,3) in (x,y) space and $\hat{Q}$ in (u,v) space in Fig 3a-b. From Eqs.(28)-(31) , we obtain the following relations connecting the submatrices $S^{i,j,e}$ and $K^{i,j,e}$

We now obtain the submatrices $S^{i,j,e}$ and $K^{i,j,e}$ in an explicit form from Eqs.(28a)- (28b) as

$$S^{i,j,e} = \iint_{Q_e} G_{x,y}^{i,j,e}\; dx\,dy = \begin{pmatrix} \iint_{Q_e} \frac{\partial N_i^e}{\partial x}\frac{\partial N_j^e}{\partial x}\, dxdy & \iint_{Q_e} \frac{\partial N_i^e}{\partial x}\frac{\partial N_j^e}{\partial y}\, dxdy \\ \iint_{Q_e} \frac{\partial N_i^e}{\partial y}\frac{\partial N_j^e}{\partial x}\, dxdy & \iint_{Q_e} \frac{\partial N_i^e}{\partial y}\frac{\partial N_j^e}{\partial y}\, dxdy \end{pmatrix}$$

$$= \begin{pmatrix} S_{2i-1,2j-1}^e & S_{2i-1,2j}^e \\ S_{2i,2j-1}^e & S_{2i,2j}^e \end{pmatrix} \text{ (say) } \text{------------------(32)}$$

and in similar manner

$$K^{i,j,e} = \iint_{\hat{Q}} G_{u,v}^{i,j,e}\; du\,dv = \begin{pmatrix} \iint_{\hat{Q}} \frac{\partial N_i^e}{\partial u}\frac{\partial N_j^e}{\partial u}\, dudv & \iint_{\hat{Q}} \frac{\partial N_i^e}{\partial u}\frac{\partial N_j^e}{\partial v}\, dudv \\ \iint_{\hat{Q}} \frac{\partial N_i^e}{\partial v}\frac{\partial N_j^e}{\partial u}\, dudv & \iint_{\hat{Q}} \frac{\partial N_i^e}{\partial v}\frac{\partial N_j^e}{\partial v}\, dudv \end{pmatrix}$$

$$= \begin{pmatrix} K_{2i-1,2j-1}^e & K_{2i-1,2j}^e \\ K_{2i,2j-1}^e & K_{2i,2j}^e \end{pmatrix} \text{ (say) } \text{------------------(33)}$$

We have now obtain from the above Eq.(19)-(33)

$$S^{i,j,e} = \iint_{Q_e} G^{i,j,e}_{x,y}\ dx\,dy = \iint_{\hat{Q}} (P\, G^{i,j,e}_{u,v} P^T) \frac{\partial(x,y)}{\partial(u,v)}\ du\,dv$$
$$= 2\Delta_{abc} \iint_{\hat{Q}} (P\, G^{i,j,e}_{u,v} P^T)\ du\,d$$
$$= 2\Delta_{abc}\, P \left(\iint_{\hat{Q}} G^{i,j,e}_{u,v}\, du\,dv\right) P^T$$
$$= 2\Delta_{abc}\ P\,(K^{i,j,e})\ P^T \qquad \text{-----------------------(34)}$$

We can thus obtain the global derivative integrals in the physical space or Cartesian space (x,y) by using the matrix triple product established in eqn.(34).

From Eq.(33) and noting the fact that $\hat{Q}$ is the quadrilateral in (u, v) space spanned by the vertices (1/3, 1/3), (0, 1/2), (0, 0)and (1/2, 0) we obtain

$$K^{i,j,e} = \iint_{\hat{Q}} G^{i,j,e}_{u,v}\ du\,dv$$
$$= \int_{-1}^{1}\int_{-1}^{1} G^{i,j,e}_{u,v} \frac{\partial(u,v)}{\partial(\xi,\eta)}\ d\xi\, d\eta \qquad \text{------------------------(35)}$$

We now refer to section 5 of this paper, in this section we have derived the necessary relations to integrate the integrals of Eq.(35), As in Eq.(14) and Eq.(15) , we use the transformation

$$u(\xi,\ \eta) = \frac{1}{3} N_1(\xi,\ \eta) + \frac{1}{2} N_4(\xi,\ \eta)$$
$$v(\xi,\ \eta) = \frac{1}{3} N_1(\xi,\ \eta) + \frac{1}{2} N_2(\xi,\ \eta) \qquad \text{------------------------(36)}$$

to map the quadrilateral $\hat{Q}$ to the 2-square -1≤ $\xi, \eta \leq 1$ Using Eq.(36) in Eq.(35), we obtain

$$K^{i,j,e} = \iint_{\hat{Q}} G^{i,j,e}_{u,v} \left(\frac{4+\xi+\eta}{96}\right) d\xi d\eta \qquad \text{----------------------- (37)}$$

The submatrices for the quadrilateral Q_e is expressed from Eq.(34) as

$$S^{i,j,e} = (2\Delta_{abc})\ P\,(K^{i,j,e})\ P^T \qquad \text{---------------------- (38)}$$

In eqn.(38) , $2\Delta_{abc}$= 2 x area of the triangle spanning vertices A(x_a ,y_a) ,B(x_b ,y_b) , C(x_c , y_c) which is scalar.

The matrices P, P^T depend purely on the nodel coordinates (x_a ,y_a) ,(x_b ,y_b) , (x_c , y_c) the matrix $K^{i,j,e}$ can be explicity computed by the relations obtained in section 2 and 3 . We find that $K^{i,j,e}$ is a (2X2) matrix of integrals whose integrands are rational functions with polynomial numerator and the linear denominator $(4 + \xi + \eta)$. Hence these integrals can be explicity computed. The explicit values of these integrals are expressible in terms of logarithmic constants. We have used symbolic mathematics software of MATLAB to compute the explicit values and their conversion to any number of digits can be obtained by using variable precision arithmetic (vpa) command. The matrix K^e as noted in Eq.(33) is of order ($2xn_{de}$) x ($2xn_{de}$) . We have computed K^e for the four noded isoparametric quadrilateral element. This is listed in Table 1A and Table 1B ,

7. Application Example :

In this section, we examine the application of the proposed explicit integration scheme to the Saint Venant Torsion problem [22]. Exact solutions for simple cross sections such as circle, ellipse, equilateral triangle and rectangle have been rigorously derived. These problems are described by the following boundary value problem ;

$$\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + 2G\theta = 0 \quad \text{in R} \qquad \text{-------------------- (39)}$$

$$\phi = 0 \quad \text{on} \quad \partial R \text{, the boundary of R} \qquad \text{--------------------- (40)}$$

where ϕ(x,y) is known as Prandtl stress function, G is the shear modulus, θ is the angle of twist per unit length, R is the cross sectional region and ∂R is the boundary of R. We choose Gθ = 1 for the sake of simplicity. Then the corresponding torisonal constant is given by the equation

$$t_c = 2 \iint_R \phi(x,y) dx\,dy \qquad \text{---------------------- (41)}$$

7.1 Torison of a rectangular Cross section :

We consider the region R as the rectangular cross section with the vertices (-a, b) , (a ,- b) and (a, b) ,(-a, b) as shown in Fig.4

From the theory of elasticity [22-24] , the Prandtl stress function ϕ and the torisonal constant t_c for the rectangular cross section of length 2b and breadth 2a are given by the following expression in series form

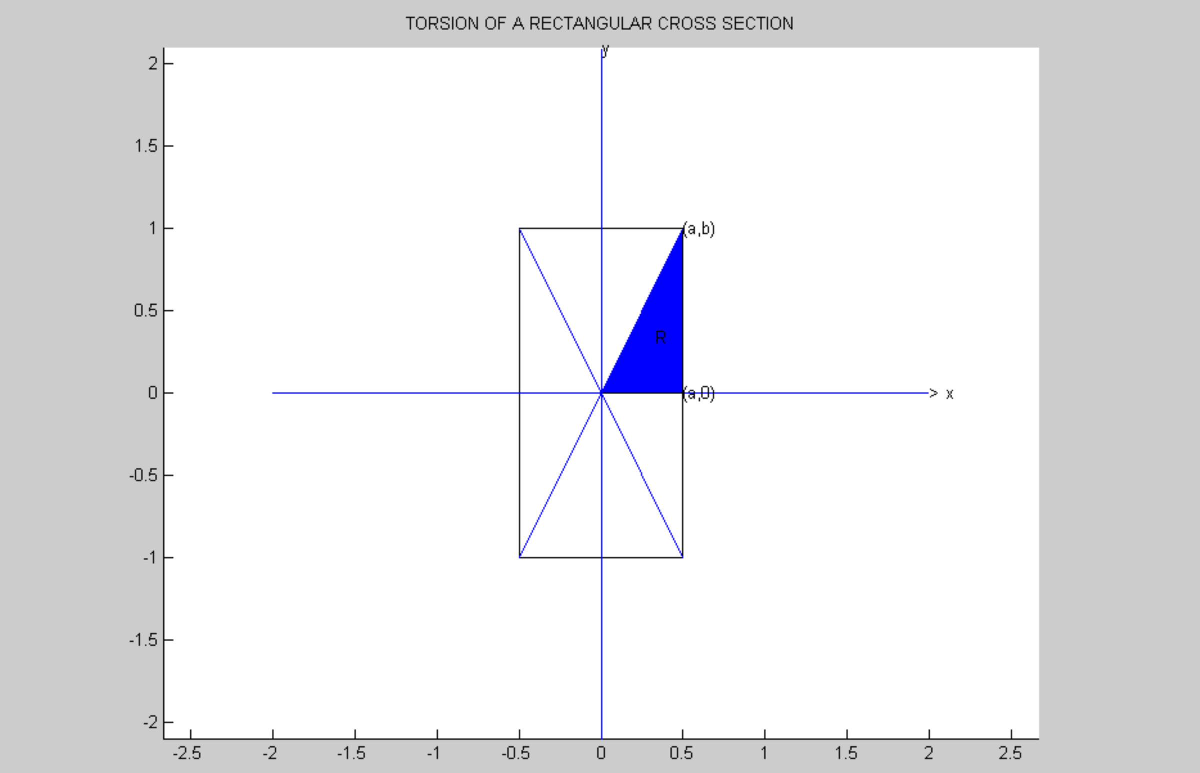


Fig4.rectangular shaft of dimension 2a*2b

$$\Phi = \frac{32a^2}{\pi^3}\sum_{n=1,3,5,......}\frac{(-1)^{\frac{(n-1)}{2}}}{n^3}\left[1-\frac{\cosh(\frac{n\pi y}{2a})}{\cosh(\frac{n\pi b}{2a})}\right]\cos(\frac{n\pi x}{2a}) \quad \text{------------------------- (42)}$$

$$t_c = \frac{(2a)^3(2b)}{3}\left[1-\frac{192}{\pi^5}\sum_{n=1,3,5,......}\frac{1}{n^5}\tan(\frac{n\pi b}{2a})\right] \quad \text{-------------------------- (43)}$$

These expressions converge rapidly for b > a. In this study we consider the square cross section of unit length for which a = b = ½ .

7.2 Finite Element procedure :

We consider the rectunglar region of Fig.4 with a = b = ½. This cross section has four axes of symmetry, therefore, only one eight of the cross section needs to be analysed. We thus have to model the region R, the right isosceles triangular cross section with vertices (0, 0), (1/2, 0) , (1/2 , ½) as shown in Fig 4 . We assume that the domain R is discretised by Quadrilateral elements Q_e , the Prandtl stress function $\phi^e(x,y)$ is expressed in terms of the natural coordinate variates (ξ,η) such that

$$\phi^e(x,y) = \sum_{i=1}^{n_{de}} N_i^e\,(\xi,\eta)\phi_i^e \quad \text{------------------------ (44)}$$

Where $N_i^e\,(\xi,\eta)$ denotes the basis function at node i , ϕ_i^e is the corresponding nodal value and n_{de} denots the number of nodes per element. Using the modified Galerkin weighted residual method, the numerical solution of the above boundary value problem for the domain R is expressed as

$$[\,K\,]\{\phi_N\} = \{F_N\} \;, \qquad (N = 1,2,\text{-----------}\; n_d) \quad \text{----------------------(45)}$$

Where n_d= the numbers of nodes used in discretisation of R .

Where $[\,K\,] = \sum_{e=1}^{n_e}[K_{ij}^e]$, $\{F_N\} = \sum_{e=1}^{n_e}\{F_i^e\}$ (i, j = 1,2,-------, n_{de}) -----------------(46)

$$K_{i,j}^e = \iint_{Q_e}\left(\frac{\partial N_i^e}{\partial x}\frac{\partial N_j^e}{\partial x}+\frac{\partial N_i^e}{\partial y}\frac{\partial N_j^e}{\partial y}\right)dxdy \quad \text{----------------------(47)}$$

$$F_i^e = 2\iint_{Q_e} N_i^e\,(\xi,\eta)dxdy \quad \text{---------------------(48)}$$

$$\{(\phi_N\,, F_N\,)\,,\;\; N = 1,2,\text{-------},\, n_e\,\} \quad \text{---------------------(49)}$$

n_e = the number of elements of the domain R

7.3 Discretisation and Finite Element Model :

In designing a finite element model, we shall discretise R , the one octant (a right isosceles triangle) with vertices (0, 0) , (1/2, 0) and (1/2, 1/2) into quadrilateral elements as described in section 2 – 4 . This discretisation generates the Jacobian $(4 + \xi + \eta) \times$ a constant for all such quadrilaterals of the domain R, We shall consider the following boundary value problem

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} + 2 = 0 \text{ , within R} \quad \text{----------------------(50a)}$$

$$\phi = 0 \text{ , on side AB} \quad \text{----------------------(50b)}$$

$$\frac{\partial \phi}{\partial n} = 0 \text{ , on sides OA and OB, the lines of symmetry} \quad \text{----------------------(50c)}$$

Mesh1: The region R, the right isosceles triangle, is discretised into three quadrilaterals which are obtained by joining the centroid to the midpoint of the sides.

Mesh 2: We discretise R into three triangles by joining the centroid of R to the three vertices of R . Then each of these triangles are divided three quadrilaterals . This discretises R into nine quadrilaterals as explained in section 4.

We then discretise the region R into 2^2, 3^2, 4^2, 5^2 ,6^2, 7^2 ,8^2 and 9^2 triangles of equal size, each of these triangles are further discretised into three quadrilaterals as explained in section 2-4. This will generate meshes 3-10 with 12,27,48,75,108,147,192 and 243 quadrilateral elements respectively We have depicted these meshes in Figs.5-14. Finite element solutions for these discretisations i.e. for meshes 1-10 is depicted in Tables. 2-12. We find that the numerical solutions converge as the meshes are refined.

In a recent paper[26] a new approach to automatic generation of all quadrilateral mesh for finite analysis is proposed. We have used this to discretised the 1/8-th of the square cross section into an all quadrilateral mesh. The following MATLAB PROGRAMS are written for this purpose:

(1) D2LaplaceEquationQ4Ex3automeshgen.m
(2)coordinate_rtisoscelestriangle00_h0_hh.m
(3)nodaladdresses4special_convex_quadrilaterals.m
(4)quadrilateralmesh_square_cross_section_q4.m

These are appended for reference

Conclusions:

This paper proposes the explicit integration scheme for a unique linear convex quadrilateral which can be obtained from an arbitrary linear triangle by joining the centroid to the midpoints of sides of the triangle. The explicit integration scheme proposed for these unique linear convex quadrilaterals is derived by using the standard transformations in two steps. We first map an arbitrary triangle into a standard right isosceles triangle by using a affine linear transformation from global (x, y) space into a local space (u, v). We discritise this standard right isosceles triangle in (u, v) into three unique linear convex quadrilaterals. We have shown that any unique linear convex quadrilateral in (x, y) space can be mapped into one of the unique quadrilaterals in (u, v) space. We can always map these linear convex quadrilaterals into a 2-aquare in (ξ, η) space by an approximate transformation. Using these two mapping, we have established an integral derivative product relation between the linear convex quadrilaterals in the (x, y) space interior to the arbitrary triangle and the linear convex quadrilaterals of the local space (u, v) interior to the standard right isosceles triangle. Further , we have shown that the product of global derivative integrals $S^{i,j,e}$ in (x, y) space can be expressed as a matrix triple product P $(K^{i,j,e})$ P^T X (2 * area of the arbitrary triangle in (x, y) space) in which P is a geometric properties matrix and $K^{i,j,e}$ is the product of global derivative integrals in (u, v) space.

We have shown that the explicit integration of the product of local derivative integrals in (u, v) space over the unique quadrilateral spanning vertices (1/3 ,1/3), (0, ½), (0, 0) , (1/2, 0) is now possible by use of symbolic processing capabilities in MATLAB which are based on Maple – V software package. The proposed explicit integration scheme is a useful technique for boundary value problems governed by either a single equation or a system of second order partial differential equations. The physical applications of such problems are numerous in science, and engineering, the examples of single equations are the well known Laplace and Poisson equations with suitable boundary conditions and the examples of the system of equations are plane stress, plane stress and axisymmetric stress analysis etc in linear elasticity. We have demonstrated the proposed explicit integration scheme to solve the St. Venant Torsion problem for a square cross section. Monotonic convergence from below is observed with known analytical solutions for the Prandtl stress function and the torisonal constant which are expessed in terms of infinite series as noted in Eqs.(42-43) of this paper. We conclude that an efficient scheme is developed in this paper which will be useful for the solution of many physical problems governed by second order partial differential equations.

REFERENCES:

[1] Zienkiewicz O.C, Taylor R.L and J.Z Zhu, Finite Element Method, its basis and fundamentals, Elservier, (2005)
[2] Bathe K.J, Finite Element Procedures, Prentice Hall, Englewood Cliffs, N J (1996)
[3] Reddy J.N, Finite Element Method, Third Edition, Tata Mc Graw-Hill (2005)
[4] Burden R.L and J.D Faires, Numerical Analysis, 9th Edition, Brooks/Cole, Cengage Learning (2011)
[5] Stroud A.H and D.Secrest, Gaussian quadrature formulas, Prentice Hall,Englewood Cliffs, N J, (1966)
[6] Stoer J and R. Bulirsch, Introduction to Numerical Analysis, Springer-Verlag, New York (1980)
[7] Chung T.J, Finite Element Analysis in Fluid Dynamics, pp.191-199, Mc Graw Hill, Scarborough, C A , (1978)
[8] Rathod H.T, Some analytical integration formulae for four node isoparametric element, Computer and structures 30(5), pp.1101-1109, (1988)
[9] Babu D.K and G.F Pinder, Analytical integration formulae for linear isoparametric finite elements, Int. J. Numer. Methods Eng 20, pp.1153-1166
[10] Mizukami A, Some integration formulas for four node isoparametric element, Computer Methods in Applied Mechanics and Engineering. 59 pp. 111-
121(1986)
[11] Okabe M, Analytical integration formulas related to convex quadrilateral finiteelements, Computer methods in Applied mechanics and Engineering. 29, pp.201-218 (1981)
[12] Griffiths D.V, Stiffness matrix of the four node quadrilateral element in closed form, International Journal for Numerical Methods in Engineering. 28, pp.687- 703(1996)
[13] Rathod H.T and Md. Shafiqul Islam, Integration of rational functions of bivariate polynomial numerators with linear denominators over a (-1,1) square in a local parametric two dimensional space, Computer Methods in Applied Mechanics and Engineering. 161 pp.195-213 (1998)
[14] Rathod H.T and Md. Sajedul Karim, An explicit integration scheme based onrecursion and matrix multiplication for the linear convex quadrilateral elements, International Journal of Computational Engineering Science. 2(1) pp. 95-135(2001)
[15] Yagawa G, Ye G.W and S. Yoshimura, A numerical integration scheme for finite element method based on symbolic manipulation, International Journal for Numerical Methods in Engineering. 29, pp.1539-1549 (1990)
[16] Rathod H.T and Md. Shafiqul Islam , Some pre-computed numeric arrays for linear convex quadrilateral finite elements, Finite Elements in Analysis and Design 38, pp. 113-136 (2001)
[17] Hanselman D and B. Littlefield, Mastering MATLAB 7 , Prentice Hall, Happer Saddle River, N J . (2005)
[18] Hunt B.H, Lipsman R.L and J.M Rosenberg , A Guide to MATLAB for beginners and experienced users, Cambridge University Press (2005)
[19] Char B, Geddes K, Gonnet G, Leong B, Monagan M and S.Watt , First Leaves; A tutorial Introduction to Maple V , New York : Springer–Verlag (1992)
[20] Eugene D, Mathematica , Schaums Outlines Theory and Problems, Tata Mc Graw Hill (2001)
[21] Ruskeepaa H, Mathematica Navigator, Academic Press (2009)
[22] Timoshenko S.P and J.N Goodier , Theory of Elasticity, 3rd Edition, Tata Mc Graw Hill Edition (2010)
[23] Budynas R.G, Applied Strength and Applied Stress Analysis, Second Edition,Tata Mc Graw Hill Edition (2011)
[24] Roark R.J, Formulas for stress and strain, Mc Graw Hill, New York (1965)
[25] Nguyen S.H, An accurate finite element formulation for linear elastic torsion calculations, Computers and Structures. 42, pp.707-711 (1992)
[26]Rathod H.T,Rathod .Bharath,Shivaram.K.T,Sugantha Devi.K, A new approach to automatic generation of all quadrilateral mesh for finite analysis, International Journal of Engineering and Computer Science, Vol. 2,issue 12,pp3488-3530(2013)
[27] Rathod H.T, Venkatesh.B, Shivaram. K.T,Mamatha.T.M, Numerical Integration over polygonal domains using convex quadrangulation and Gauss Legendre Quadrature Rules, International Journal of Engineering and Computer Science, Vol. 2,issue 8,pp2576-2610(2013)

TABLES

Table-1A
Values of integrals of the product of global derivatives over the quadrilateral {(xk,yk),k=1,2,3,4}={(1/3,1/3),(0,1/2),(0,0),(1/2,0)}, in the interior of the standard triangle (see eqn (33))
IntJdnidnjuvrs=[Ke (2*i-1,2*j-1) Ke (2*i-1,2*j)
Ke (2*i,2*j-1) Ke (2*i,2*j)]
where, (i,j=1,2,3,4,5,6,7,8)
ANALYTICAL VALUES

—————————————————————

```
IntJdn1dn1uvrs = [  -11/2-34*log (2)+27*log (3), -1/2-20*log (2)+27/2*log (3);...
            -1/2-20*log (2)+27/2*log (3),  -11/2-34*log (2)+27*log (3) ]
```

```
IntJdn1dn2uvrs =  [11/3+68/3*log (2)-18*log (3),   5/6+40/3*log (2)-9*log (3);...
             1/3+40/3*log (2)-9*log (3), 25/6+68/3*log (2)-18*log (3) ]
```

```
IntJdn1dn3uvrs = [   -7/3-34/3*log (2)+9*log (3), -2/3-20/3*log (2)+9/2*log (3);...
            -2/3-20/3*log (2)+9/2*log (3),   -7/3-34/3*log (2)+9*log (3)]
```

```
IntJdn1dn4uvrs = [ 25/6+68/3*log (2)-18*log (3),   1/3+40/3*log (2)-9*log (3);...
             5/6+40/3*log (2)-9*log (3), 11/3+68/3*log (2)-18*log (3)]
```

```
IntJdn2dn1uvrs = [ 11/3+68/3*log (2)-18*log (3),   1/3+40/3*log (2)-9*log (3);...
             5/6+40/3*log (2)-9*log (3), 25/6+68/3*log (2)-18*log (3)]
```

```
IntJdn2dn2uvrs = [ -22/9-136/9*log (2)+12*log (3),    -5/9-80/9*log (2)+6*log (3);...
              -5/9-80/9*log (2)+6*log (3), -22/9-136/9*log (2)+12*log (3) ]
```

```
IntJdn2dn3uvrs =[   14/9+68/9*log (2)-6*log (3),   4/9+40/9*log (2)-3*log (3);...
            -1/18+40/9*log (2)-3*log (3), 19/18+68/9*log (2)-6*log (3)]
```

```
IntJdn2dn4uvrs = [ -25/9-136/9*log (2)+12*log (3),    -2/9-80/9*log (2)+6*log (3);...
              -2/9-80/9*log (2)+6*log (3), -25/9-136/9*log (2)+12*log (3)]
```

```
IntJdn3dn1uvrs = [   -7/3-34/3*log (2)+9*log (3), -2/3-20/3*log (2)+9/2*log (3);...
            -2/3-20/3*log (2)+9/2*log (3),   -7/3-34/3*log (2)+9*log (3) ]
```

```
IntJdn3dn2uvrs = [ 14/9+68/9*log (2)-6*log (3), -1/18+40/9*log (2)-3*log (3);...
             4/9+40/9*log (2)-3*log (3), 19/18+68/9*log (2)-6*log (3)]
```

```
IntJdn3dn3uvrs =[   -5/18-34/9*log (2)+3*log (3), 5/18-20/9*log (2)+3/2*log (3);...
            5/18-20/9*log (2)+3/2*log (3),  -5/18-34/9*log (2)+3*log (3) ]
```

```
IntJdn3dn4uvrs = [ 19/18+68/9*log (2)-6*log (3),   4/9+40/9*log (2)-3*log (3);...
            -1/18+40/9*log (2)-3*log (3),  14/9+68/9*log (2)-6*log (3)]
```

```
IntJdn4dn1uvrs =  [25/6+68/3*log (2)-18*log (3),   5/6+40/3*log (2)-9*log (3);...
             1/3+40/3*log (2)-9*log (3), 11/3+68/3*log (2)-18*log (3)]
```

```
IntJdn4dn2uvrs =  -25/9-136/9*log (2)+12*log (3),    -2/9-80/9*log (2)+6*log (3);...
              -2/9-80/9*log (2)+6*log (3), -25/9-136/9*log (2)+12*log (3)
```

```
 IntJdn4dn3uvrs = [ 19/18+68/9*log (2)-6*log (3), -1/18+40/9*log (2)-3*log (3);...
              4/9+40/9*log (2)-3*log (3),  14/9+68/9*log (2)-6*log (3)]
```

```
IntJdn4dn4uvrs = [ -22/9-136/9*log (2)+12*log (3),    -5/9-80/9*log (2)+6*log (3);...
              -5/9-80/9*log (2)+6*log (3), -22/9-136/9*log (2)+12*log (3)]
```

Table-1B

Values of integrals of the product of global derivatives over the quadrilateral {(xk,yk),k=1,2,3,4}={(1/3,1/3),(0,1/2),(0,0),(1/2,0)}, in the interior of the standard triangle (see eqn (33)).

```
 IntJdnidnjuvrs= K_e (2*i-1,2*j-1)  K_e (2*i-1,2*j)
             K_e (2*i,2*j-1)       K_e (2*i,2*j)
 where,  (i,j=1,2,3,4,5,6,7,8)
```

NUMERICAL VALUES IN VARIABLE PRECISION ARITHMETIC

```
intJdn1dn1uvrs = vpa (sym (' .59552765500082114748572926733O')), vpa (sym ('.468322285820574645491168269290'));...
            vpa (sym (' .46832228582057464549116826929')), vpa (sym ('.59552765500082114748572926733O'))  ;
intJdn1dn2uvrs = vpa (sym ('  -.397018436667214098323819511552')),  vpa (sym ('.187785142786283569672554487139 5'));...
            vpa (sym (' -.312214857213716430327445512860 4')),   vpa (sym ('.102981563332785901676180488448'))  ;
intJdn1dn3uvrs = vpa (sym (' -.301490781666392950838090244223 5')),vpa (sym (' -.343892571393141784836277243570 0'));...
           vpa (sym (' -.343892571393141784836277243570 0')), vpa (sym ('-.301490781666392950838090244223 5'))  ;
intJdn1dn4uvrs = vpa (sym ('   .102981563332785901676180488448')), vpa (sym ('-.312214857213716430327445512860 4'));...
           vpa (sym ('  .187785142786283569672554487139 5')),  vpa (sym ('-.397018436667214098323819511552'))  ;
```

```
intJdn2dn1uvrs = vpa (sym ('  -.397018436667214098323819511552')), vpa (sym ('-.312214857213716430327445512860 4'));...
            vpa (sym ('  .187785142786283569672554487139 5')),  vpa (sym (' .102981563332785901676180488448'))  ;
intJdn2dn2uvrs = vpa (sym ('   .264678957778142732215879674369')), vpa (sym ('-.125190095190855713115036324760 0'));...
            vpa (sym (' -.125190095190855713115036324760 0')),  vpa (sym (' .264678957778142732215879674369'))  ;
intJdn2dn3uvrs = vpa (sym ('  .200993854444261967225393496149 1')),  vpa (sym ('.229261714262094523224184829046 6'));...
            vpa (sym (' -.270738285737905476775815170953 4')), vpa (sym ('-.299006145555738032774606503850 9'))  ;
intJdn2dn4uvrs = vpa (sym (' -.686543755551906011174536589 65e-1')),   vpa (sym ('.208143238142477620218297008573 4'));...
            vpa (sym ('   .208143238142477620218297008573 4')), vpa (sym ('-.686543755551906011174536589 65e-1'))  ;
```

```
intJdn3dn1uvrs = vpa (sym (' -.301490781666392950838090244223 5')), vpa (sym ('-.343892571393141784836277243570 0'));...
```

```
            vpa (sym (' -.3438925713931417848362772435700')),vpa (sym (' -.3014907816663929508380902442235'))  ;
intJdn3dn2uvrs = vpa (sym ('  .2009938544442619672253934961491')), vpa (sym ('-.2707382857379054767758151709534'));...
            vpa (sym ('  .2292617142620945232241848290466')), vpa (sym ('-.2990061455557380327746065038509'))  ;
intJdn3dn3uvrs = vpa (sym (' .3995030727778690163873032519254')), vpa (sym ('.3853691428689527383879075854768'));...
            vpa (sym (' .3853691428689527383879075854768')), vpa (sym ('.3995030727778690163873032519254'))  ;
intJdn3dn4uvrs = vpa (sym (' -.2990061455557380327746065038509')), vpa (sym (' .2292617142620945232241848290466'));...
           vpa (sym (' -.2707382857379054767758151709534')),  vpa (sym ('.2009938544442619672253934961491'))  ;
___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
intJdn4dn1uvrs = vpa (sym ('   .102981563332785901676180488448')),  vpa (sym ('.1877851427862835696725544871395'));...
            vpa (sym (' -.3122148572137164303274455128604')), vpa (sym (' -.397018436667214098323819511552'))  ;
intJdn4dn2uvrs = vpa (sym (' -.68654375555190601117453658965e-1')),  vpa (sym (' .2081432381424776202182970085734'));...
            vpa (sym ('   .2081432381424776202182970085734')), vpa (sym ('-.68654375555190601117453658965e-1'))  ;
intJdn4dn3uvrs = vpa (sym (' -.2990061455557380327746065038509')), vpa (sym ('-.2707382857379054767758151709534'));...
            vpa (sym ('  .2292617142620945232241848290466')),  vpa (sym ('.2009938544442619672253934961491'))  ;
intJdn4dn4uvrs = vpa (sym ('   .264678957778142732215879674369')), vpa (sym ('-.1251900951908557131150363247600'));...
            vpa (sym (' -.1251900951908557131150363247600')),  vpa (sym (' .264678957778142732215879674369'))  ;
___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___ ___
```

Table-2

TORSION OF SQUARE CROSS-SECTION:MESH-1: THREE QUADRILATERAL ELEMENTS
(ONLY NONZERO Prandtl Stress Values at the centroid of triangles SHOWN)

node number	Prandtl Stress Values	
	fem-computed values	analytical (theoretical)-values
5,	.77947339554968456166610494884639e-1,	.79829131790170537262176536669693e-1

Table-3

TORSION OF SQUARE CROSS-SECTION:MESH-2: 9- QUADRILATERAL ELEMENTS
(ONLY NONZERO Prandtl Stress Values SHOWN)

node number	Prandtl Stress Function Values	
	fem-computed values	analytical (theoretical)-values
8,	.99523800240084704809635838469436e-1,	.10556520659621227098813928595278
9,	.28816342706667324479818830743933e-1,	.29706675401915148676262816502474e-1
10,	.83610741082209335687697724611636e-1,	.89368831708826024134386380235600e-1

Table-4

TORSION OF SQUARE CROSS-SECTION:MESH-3: 12- QUADRILATERAL ELEMENTS
(ONLY NONZERO Prandtl Stress Values at the centroid of triangles SHOWN)

node number	Prandtl Stress Function Values	
	fem-computed values	analytical (theoretical)-values
16,	.13029059137868921922272702133360,	.13010464886468359125466215623221
17,	.79223767223566222816511863813028e-1,	.79829131790170537262176536669693e-1
18,	.48150403466555868168471181547365e-1,	.48693699564520332429831250748512e-1
19,	.31242397535836620124969388790690e-1,	.32225496081814471416762780958210e-1

Table-5

TORSION OF SQUARE CROSS-SECTION:MESH-4: 27- QUADRILATERAL ELEMENTS
(ONLY NONZERO Prandtl Stress Values at the centroid of triangles SHOWN)

node number	Prandtl Stress Function Values	
	fem-computed values	analytical (theoretical)-values
29,	.13989501131742645868829286759857,	.13965110124150523273465789171824
30,	.11679997023158445074338650908185,	.11686601505688190360612072456382
31,	.10550979541349207421985362063955,	.10556520659621227098813928595280
32,	.60938560229034914148038557897210e-1,	.61189615839758513842734917399555e-1
33,	.34005220622060315145263656353142e-1,	.34241043563578881418698404142963e-1
34,	.89154073277178416294433103529623e-1,	.89368831708826024134386380235602e-1
35,	.47836096130094456475157600946899e-1,	.48243665496142199368678617585149e-1
36,	.29355696601266451973852274280867e-1,	.29706675401915148676262816502474e-1
37,	.16986979174838342966632790370480e-1,	.17509583144943248667659734963092e-1

Table-6

TORSION OF SQUARE CROSS-SECTION:MESH-5: 48- QUADRILATERAL ELEMENTS

(ONLY NONZERO Prandtl Stress Values at the centroid of triangles SHOWN)

node number	Prandtl Stress Values fem-computed values	anlytical (theoretical)-values
46,	.14321101947553282495576418553650,	.14301005788295951725595268076918
47,	.13013794692350628069853269842020,	.13010464886468359125466215623221
48,	.12427441282759811061477609009812,	.12425010763562328210221787165997
49,	.99893766015774738256159770495766e-1,	.99966331891680868840902694826967e-1
50,	.86792214881356328312243769271991e-1,	.86848158236067333455652287667199e-1
51,	.48560477985656668095881354604925e-1,	.48693699564520332429831250748513e-1
52,	.26188172918558176248811071081980e-1,	.26320778834398639235310402689663e-1
53,	.11337784297370207933554753204885,	.11342449829597557980841251335624
54,	.87739776367379138970744541662629e-1,	.87868441821279855606778663607124e-1
55,	.79718154463353022219783879304410e-1,	.79829131790170537262176536669691e-1
56,	.43214110377026819154603624871216e-1,	.43405635618515166865021126035825e-1
57,	.24277731550516451214337020414154e-1,	.24453761494389594391945108114715e-1
58,	.62771016318011897507939300324700e-1,	.62957785243503804052677732105945e-1
59,	.31954605447154489855113085978633e-1,	.32225496081814471416762780958211e-1
60,	.19670330186818827327089734384029e-1,	.19897928847669679779254406105713e-1
61,	.10796926414864615571974859275303e-1,	.11120097424799775346324124458546e-1

Table-7

TORSION OF SQUARE CROSS-SECTION:MESH-6:75-QUADRILATERAL ELEMENTS

(ONLY NONZERO Prandtl Stress Values at the centroid of triangles SHOWN)

node number	Prandtl Stress Values fem-computed values	anlytical (theoretical)-values
67,	.14472816091258142826536472656803,	.14456814738793862691472853858124
68,	.13633685483814324931012363169042,	.13628207221289742574377559202182
69,	.13272853820062009678921009615629,	.13268399708946871663449981516470
70,	.11732519029077747241631713467691,	.11733862929285050890042926192301
71,	.10954983129822753484268302414135,	.10956157070281676867272762921215
72,	.86079726960103685971729556272346e-1,	.86135610414287094635657604327851e-1
73,	.73285827310651491804709086957975e-1,	.73329545131624707159651924771251e-1
74,	.40166306466813122647680495284043e-1,	.40249054055218616682320834651519e-1
75,	.21271954747874086184277691176839e-1,	.21359044257827833510365163584636e-1
76,	.12525364046018136467953459767034,	.12524934142062286363549511609375
77,	.10813688326828442799026762651477,	.10818013363726278917382204456608
78,	.10365939057722727095085919037027,	.10369876666427781117751751894623
79,	.79747020420204826359705429509995e-1,	.79829131790170537262176536669693e-1
80,	.69624400818389479438265126667743e-1,	.69693083395112164735148953082738e-1
81,	.37471652227034369110805070009322e-1,	.37581999150598016943305148301890e-1
82,	.20304300275860357961144324338020e-1,	.20413616727298556319246302126125e-1
83,	.90062284317490524920926776601800e-1,	.90138938192527541739117743928278e-1
84,	.67148113716386073852653925039631e-1,	.67262701546232552806716724178216e-1
85,	.61107932593720463260535304963036e-1,	.61210310291009902905540140634793e-1
86,	.32070697771012559192141692163563e-1,	.32214292180078706690963711081372e-1
87,	.18066410912048367267860827651059e-1,	.18195828228972691993131269996738e-1
88,	.46456934801244082536312624289246e-1,	.46601374740419173778862258507862e-1
89,	.22958622501612040683735283938797e-1,	.23150481555609002594280554851447e-1
90,	.14155851883316966379902332665019e-1,	.14316112721877375026405489141045e-1
91,	.75272046209275391130589346848070e-2,	.77496393243762219590605393381412e-2

Table-8

TORSION OF SQUARE CROSS-SECTION:MESH-7:108-QUADRILATERAL ELEMENTS

(ONLY NONZERO Prandtl Stress Values at the centroid of triangles SHOWN)

node number	Prandtl Stress Values fem-computed values	anlytical (theoretical)-values
92,	.14554423990634697033182606799007,	.14541519087750028202634154821047
93,	.13970765941860376304106806861032,	.13965110124150523273465789171824
94,	.13725638608260297839172849967174,	.13720834127291368719539871135471
95,	.12663468703175169917430463885978,	.12662549883532569572743325352434
96,	.12146658145118814306765795277883,	.12145954122022938968452993091895
97,	.10554370074306498685875582831562,	.10556520659621227098813928595280
98,	.97242110118208303458153466192466e-1,	.97260097236328765263200055165380e-1
99,	.75245592683941808240976665683763e-1,	.75287259490189508378116804846224e-1
100,	.63273294158415731542153780894391e-1,	.63307206738420259249897012707332e-1
101,	.34185085985871815063389076840582e-1,	.34241043563578881418698404142963e-1
102,	.17902909001617725318101903041581e-1,	.17964036283480590393250895198593e-1
103,	.13187653058204768134042117386174,	.13185461021582553108259007040873
104,	.11972286908603705852902075076569,	.11973265527647618199196028783687
105,	.11685677521603804868027195394189,	.11686601505688190360612072456381

106,	.10002216428157233658096296087973,	.10005824037205152490864362009527
107,	.93729107782348875192044221543913e-1,	.93760976630456786971846916919905e-1
108,	.71547790514793486363578759957512e-1,	.71603651444943789079865602629054e-1
109,	.61142202122119947126963920609466e-1,	.61189615839758513842734917399556e-1
110,	.32638732680036736738637088987260e-1,	.32710184522811314173416214041160e-1
111,	.17345651614452988928330606092157e-1,	.17420152375733780864684142564698e-1
112,	.10639651408197707345502481795394,	.10642822520964413670159914706797
113,	.89314397840449013339676976434407e-1,	.89368831708826024134386380235601e-1
114,	.85722748999966170056266475822171e-1,	.85773645847253225146116320708813e-1
115,	.64335576254949389159747146679782e-1,	.64409343238997166504491410470652e-1
116,	.56269892537082185417177796166702e-1,	.56334516054221143714776791003322e-1
117,	.29618753760178281097198630803242e-1,	.29706675401915148676262816502475e-1
118,	.16085202159697177590774667478171e-1,	.16169970406443486043453603567273e-1
119,	.72521807876201537230321175547094e-1,	.72594364557231069204583796183556e-1
120,	.52879580216693540057095031302893e-1,	.52973542743876851661390213749539e-1
121,	.48157968296782009552340289040314e-1,	.48243665496142199368678617585149e-1
122,	.24772335776702475885835557004824e-1,	.24882209531338024315770834330267e-1
123,	.13976167205258931378892818884035e-1,	.14073725212626256202038391471982e-1
124,	.35830828826802392205261136139839e-1,	.35943355110016520747997583332094e-1
125,	.17367050947457062465349155911403e-1,	.17509583144943248667659734963090e-1
126,	.10719534580431434317437792545774e-1,	.10837904576413393274612847141552e-1
127,	.55776106389832203814928787138901e-2,	.57400930978983226134825232072044e-2

Table-9

TORSION OF SQUARE CROSS-SECTION:MESH-8:147-QUADRILATERAL ELEMENTS
(ONLY NONZERO Prandtl Stress Values at the centroid of triangles SHOWN)

node number	Prandtl Stress Values fem-computed values	anlytical (theoretical)-values
121,	.14603217556430304869457280055914,	.14592638948386181559691578243499
122,	.14173995727024080076716433762745,	.14168688745104490995867442564200
123,	.13996317940054296405007106541551,	.13991716447668890639130238850990
124,	.13219266115967322896216479830675,	.13217458156484612345766327865011
125,	.12849852352620175980324933619439,	.12848372538079796455836855412330
126,	.11696647453410832275409166541707,	.11697137629490281003259246387914
127,	.11112852686991736260465091545341,	.11113298180522807143982099779576
128,	.95414986948635208348857181790627e-1,	.95436333251404062976863898036597e-1
129,	.87136852258296073050014792718505e-1,	.87154554834900945414015914032371e-1
130,	.66682402599740370336418495916205e-1,	.66714217000946955175145645160642e-1
131,	.55608401453778319099338416637129e-1,	.55634724495423579109788216807996e-1
132,	.29730046880298155641792037088237e-1,	.29770967611006227722548997385638e-1
133,	.15452351410945717350307875774215e-1,	.15495979131821820652917467346131e-1
134,	.13592692502329209759579539424412,	.13589933212232736920891195421906
135,	.12687943664292446739529166293533,	.12687466156831386414456593573012
136,	.12488082136189257886628765001776,	.12487673398953419195468477100815
137,	.11240764085062867275928767958589,	.11242190887433523992327093847497
138,	.10810855259263575836033470046996,	.10812157631592580659521931207127
139,	.91857365147623001945517104409319e-1,	.91887060685317781630916876522118e-1
140,	.84883623405071132521517794644115e-1,	.84909384205373072467998139654374e-1
141,	.64341552061700572518626100101222e-1,	.64381916712723653417740436072347e-1
142,	.54261734074256062342417301070875e-1,	.54296266299973233575820265672747e-1
143,	.28761018150358929238962755027113e-1,	.28811671860809641997791304787447e-1
144,	.15102192088425719325426936920319e-1,	.15154692668211919065652810452581e-1
145,	.11674802461639802292335221083984,	.11675915578218528436942311333328
146,	.10368558641094007009735766476601,	.10371195428844917910963777527596
147,	.10130180480480852701011155245165,	.10132688180698420530753759793272
148,	.85026599018615807873122202610533e-1,	.85067089302638490329846380468040e-1
149,	.79792591648841458106257064902882e-1,	.79829131790170537262176536669704e-1
150,	.59833730561568157299297266026250e-1,	.59884593812913286735236931089497e-1
151,	.51215752131504220119942240259808e-1,	.51260199025061282181006652334001e-1
152,	.26897477458846348423948852445084e-1,	.26957490392351579082517320991947e-1
153,	.14320932102626183965019788403055e-1,	.14379507325368081889695650413719e-1
154,	.90312367476068310067721221811856e-1,	.90350960376336886947737012872892e-1
155,	.74480938857028692850163336489639e-1,	.74533230804212425379977321489265e-1
156,	.71525741919314972707029732259091e-1,	.71574957038422225517786445007445e-1
157,	.52826438053917456026420673437133e-1,	.52889584933149044047788577064648e-1
158,	.46245334142672692320161501970308e-1,	.46301359932017083243007958977734e-1
159,	.23987745553597611232959999604914e-1,	.24059233255905805797611268093001e-1
160,	.13044259388914004389647731433876e-1,	.13109656970059284758151911469294e-1
161,	.59491975844052992125370706627155e-1,	.59555191348521799921448767681899e-1
162,	.42734907267576041561689907029666e-1,	.42811746107054280672305209012921e-1
163,	.38935511662927522735775107871093e-1,	.39005937190063388467382392661731e-1
164,	.19753005584013999090345049824638e-1,	.19840166960456361216923631430329e-1
165,	.11155448052546333130849602844822e-1,	.11229876883291515068112211651503e-1
166,	.28542836461903862046807795831728e-1,	.28631975579291262778774886676735e-1
167,	.13644002051068503391666807248703e-1,	.13754823603488171613218188336765e-1
168,	.84279172634096237249771714736477e-2,	.85174190478610588075567305844306e-2
169,	.43155561141156335430422693442858e-2,	.44381678929476186244561113181490e-2

Table-10

TORSION OF SQUARE CROSS-SECTION:MESH-9:192-QUADRILATERAL ELEMENTS
(ONLY NONZERO Prandtl Stress Values at the centroid of triangles SHOWN)

node number	Prandtl Stress Values: fem-computed values	anlytical (theoretical)-values
154	.14634653554777987125274240559658	.14625800572427238855872195713380
155	.14305825810033093099901496875320	.14301005788295951725595268076918
156	.14170992085273292060267835568968	.14166760569746282343907761055730
157	.13577708908440297077607669500441	.13575571989496849937124947179516
158	.13299994058350132831165028002231	.13298180360243664900611326219531
159	.12425338378927740215601345131298	.12425010763562328210221787165998
160	.11991419706085567827702568309963	.11991173709550494952699154651953
161	.10810642679517570590071565755951	.10811600524559497933702899774486
162	.10202948972029851397485994669607	.10203783794553199515609861763521
163	.86829381257108487802071333231075e-1	.86848158236067333455652287667199e-1
164	.78800504219526948965346993500049e-1	.78816293923525654496074837356758e-1
165	.59798502176018389466712293453672e-1	.59823601770331444094709978470898e-1
166	.49571700101741499606583680990219e-1	.49592488018298537801346850337020e-1
167	.26291127030041432129754798062572e-1	.26320778834398639235310402689664e-1
168	.13590593076434694148606977762650e-1	.13623237373811829954185964042501e-1
169	.13857806429165344322604744805759	.13854948115069686535967414175032
170	.13159231290514186805028400236985	.13158085130727430175288314789359
171	.13011518878340920136870120552276	.13010464886468359125466215623221
172	.12051250476506369242538983892288	.12051581611974483038552993377550
173	.11738180157774891341903512221519	.11738509111782797499708781463440
174	.10495751313228598566168419038306	.10497258055491199207030832896719
175	.99952776621892208193161915402779e-1	.99966331891680868840902694826965e-1
176	.84410256535585874265788457209208e-1	.84434317166913011257250880437524e-1
177	.77272818258249100866024904942641e-1	.77293743737041241819601348339065e-1
178	.58225088738349474425463351200635e-1	.58255762277003269074116307610788e-1
179	.48667496853500062101633220293229e-1	.48693699564520332429831250748513e-1
180	.25643920938979853690340761954860e-1	.25680179489915586692002345256768e-1
181	.13356195832879411479871838564330e-1	.13395128375523027744433964564116e-1
182	.12366672808011697048653695314896	.12366726423053812015025811119718
183	.11341248615078979218899211714894	.11342449829597557980841251335624
184	.11171154611767250697469556964085	.11172317545195040191637015989482
185	.98967542457360587335892071000641e-1	.98989868678714803840324827829327e-1
186	.95295154793228579989250018880542e-1	.95315877932282247352296976656648e-1
187	.79798303928824888033520833357839e-1	.79829131790170537262176536669693e-1
188	.73841853592584253368768304865525e-1	.73869426449453346260010388802398e-1
189	.55220956710071323023475935483611e-1	.55258226824607823649970786466965e-1
190	.46636818318381298435260536322009e-1	.46669277341083998072339791917486e-1
191	.24411646964380935101815259258242e-1	.24453761494389594391945108114715e-1
192	.12838074210144669540542733531844e-1	.12880944328265806177680600376462e-1
193	.10266679924127930716222146178505	.10268781507636937406046465859372
194	.89865025409659917603614497095178e-1	.89895330597281725112954778154425e-1
195	.87839342816649907147119435741927e-1	.87868441821279855606778663607123e-1
196	.72759053003358681639929106843376e-1	.72797605570266589097450766540492e-1
197	.68330227886473766810272857528888e-1	.68365607770162579958277184890891e-1
198	.50617859381749633725270711157157e-1	.50662778860007909719826628761236e-1
199	.43366193462570470069621126296126e-1	.43405635618515166865021126035824e-1
200	.22519689792552289309141586766232e-1	.22568560188375041087224792019057e-1
201	.12003661725132241264289702191484e-1	.12050685583447566463119472402700e-1
202	.77224507202129625522450720670336e-1	.77262510944337357034808439985678e-1
203	.62911128061269556092708027681875e-1	.62957785243503804052677732105948e-1
204	.60433651954872659511788083933460e-1	.60477896465333378649437415184368e-1
205	.44131509843718274728090978658638e-1	.44185301398866359221673132894819e-1
206	.38654734085537548644350664727333e-1	.38702490399152458669802740855107e-1
207	.19840523486365493228149122478496e-1	.19897928847669679779254406105712e-1
208	.10798557962446369846744731551803e-1	.10850583300801482524726020279668e-1
209	.49671404922619140535101479766773e-1	.49725435093505123305505220987829e-1
210	.35296246342948841350143479983845e-1	.35359944476345665783263082120396e-1
211	.32167240491060174308371186392817e-1	.32225496081814471416762780958210e-1
212	.16153173989818981185225241458735e-1	.16222362952480223955677393158016e-1
213	.91290317911218391561644473232356e-2	.91878150598037827090979740293448e-2
214	.23324630582332958718358743083866e-1	.23396745960078524057608442063126e-1
215	.11032932748901124596240453431315e-1	.11120097424799775346324124458548e-1
216	.68189830891300267338440628383479e-2	.68892199109707352179745513652752e-2
217	.34485207373789427857946139568814e-2	.35446193557322503151795220028075e-2

Table-11

TORSION OF SQUARE CROSS-SECTION:MESH-10:243-QUADRILATERAL ELEMENTS
(ONLY NONZERO Prandtl Stress Values at the centroid of triangles SHOWN)

node number Prandtl Stress Values

fem-computed values **anlytical (theoretical)-values**

191, .14662035046283917363880537183069, .14648559012941637969898953360208
192, .14402110916078072191436650304695, .14391819240732150678302666792598
193, .14296218306959277748452178422334, .14286421806373270433042605183193
194, .13828332317405230509548681203758, .13820181362973855956283284710315
195, .13611643711223459318761663093685, .13603817042923104314037124070908
196, .12924980897729028620205496012455, .12918338008863611549726099066959
197, .12589133368789122487829992018296, .12582693203695697659863680676945
198, .11668120086664601432006310013615, .11662730400057442335896220382281
199, .11202038963255984237527293424377, .11196806593371102597368642758199
200, .10025817481091249016677791306698, .10021691297758777459141409021574
201, .94158896233189273375374986921318e-1, .94119398985631108021510652055242e-1
202, .79584783545704922694249501957635e-1, .79558072503314662181291190387044e-1
203, .71888460338964780071453708605609e-1, .71864028951832878624302828595342e-1
204, .54196397699467517934035906083761e-1, .54186949193389887164601577353965e-1
205, .44726645987535638535363540768034e-1, .44719960035949751247527951099896e-1
206, .23572442974377355977263013275234e-1, .23583062071391794231843530778811e-1
207, .12134866988423414476567640695728e-1, .12154619893372349855863546228363e-1
208, .14046536864637013059699116095492, .14037778049798911526658825244289
209, .13491431571927091460792101552694, .13483972293420226090881948971454
210, .13377556212300852411148153057224, .13370221639375423152955592924911
211, .12615974071346343444653614350264, .12609656317446959734132151520232
212, .12377202587114177713675474754405, .12371026087391560075421415238827
213, .11396374909810178542943235727218, .11391088421539154881060021167023
214, .11018866240086390071441535204733, .11013755077683459888135918315248
215, .98003396527842847907419750038935e-1, .97961452234535170700961244316719e-1
216, .92676966316611176658381724136395e-1, .92637601640289036241541221157546e-1
217, .77873665288709180807037836554191e-1, .77845134055672972050417708795692e-1
218, .70810472749044957194484016666407e-1, .70785688936690856929164149053373e-1
219, .53094079220717594973183329396722e-1, .53083246495439693935254460691517e-1
220, .44093770776802464942148656016140e-1, .44087694461971841139983444524069e-1
221, .23121519149336036314227925468409e-1, .23134134796563022736522873407692e-1
222, .11971198259447265346293600221653e-1, .11994664052533665244482619610670e-1
223, .12856198214556680833552818705695, .12849563592166507086875049616863
224, .12032465020921853684580460661004, .12026531016655805588265169877242
225, .11904646460171436292848124934918, .11898780252523643548758634555538
226, .10882459435215396610548869945994, .10877193340990049351413022373709
227, .10609981315736258022610908242603, .10604865238873079293725864828579
228, .93732901250117394965393873553801e-1, .93687599081478826502778903677624e-1
229, .89365784058630941220422710806144e-1, .89323384384246148312021452185357e-1
230, .74629420795999669092057316165339e-1, .74594044764646108464819144717208e-1
231, .68401008362101298602744291444347e-1, .68370926490201496242360880100447e-1
232, .51004511572315829364549998828989e-1, .50985619730813326750327862786304e-1
233, .42681130133758646790823337252963e-1, .42670517329736106642493906566544e-1
234, .22270859495604892142548613740448e-1, .22280221574871542867335535075477e-1
235, .11612572955503855275827889553199e-1, .11635965836723091377182428096732e-1
236, .11156504973969380222380048157086, .11150927125726302660960793474863
237, .10108958701345548952648013591076, .10103624975503209370966995334434
238, .99611450915180379765552579597686e-1, .99558897002778281909771533740434e-1
239, .87286788512302167313156891654769e-1, .87235667518047025137865911879029e-1
240, .84099260327532702622755533905062e-1, .84050202312798189254982221884419e-1
241, .69720828574294645096161267460750e-1, .69671754454481326339211262342176e-1
242, .64563159241416422516663348376312e-1, .64519752925413891121272243930435e-1
243, .47842622124312698841485669845808e-1, .47801761352370108350943413895438e-1
244, .40432667166609529733752648865621e-1, .40406142204420841641273238222758e-1
245, .20987526766612787082413202716151e-1, .20982090058334556315860425523899e-1
246, .11043564060156075255236908776218e-1, .11062293135413218906239829015902e-1
247, .90492770846342478821009109742326e-1, .90438308289771914934565513147863e-1
248, .78412969220755010157068004345092e-1, .78356466017917993995698681843352e-1
249, .76666590808897651758624390708741e-1, .76609486511481642948707483753198e-1
250, .62931340060808473430264154122799e-1, .62865212232063371628763559229749e-1
251, .59130944418028726059067072008818e-1, .59063811664675131764475234856125e-1
252, .43473361978291197410770151430942e-1, .43380321218649711718180912873599e-1
253, .37260278168216063232215362370799e-1, .37187818988711704580348976885938e-1
254, .19238514979012290858225196882771e-1, .19174404271969881044986586173264e-1
255, .10247622455683928157647211423654e-1, .10245564273289834554200416271795e-1
256, .66749314172847188737374596381573e-1, .66691353669211374423985407255566e-1
257, .53909382675575735664372563255638e-1, .53854627961965242172538842817159e-1
258, .51823251836925344936168984343154e-1, .51743750974787724968850979544551e-1
259, .37658881670729148969053176821236e-1, .37484556572385142196148703243636e-1
260, .33061665150543982600775591140253e-1, .32845773252763213589333645179093e-1
261, .17119736797963405132229499453704e-1, .16751966253885096236622662903170e-1
262, .92410147860274578382318121148048e-2, .91403233302755250695232618019829e-2
263, .42149989610474383153843166287693e-1, .42167893805004264641054080135656e-1
264, .29510399820106733300800548920283e-1, .29738899225946110653414904377857e-1
265, .27201724850057265371508700307275e-1, .27108510548354051346499846649297e-1
266, .13477499388549894445555638942295e-1, .16552101513257453110840999392127e-1
267, .81693388207229677812074387903611e-2, .76713270993821192467251305444270e-2

268, .18780917400477502364324189566515e-1, .19515278577344551258933654927860e-1
269, .86097821272342685368839801513849e-2, .91979862039801810018217452272695e-2
270, .51651252790266562236492297172247e-2, .57006599699877794167836617272014e-2
271, .27031725106197243438372065120880e-2, .29037068245629229892790949481972e-2

TABLE-12

TORISONAL CONSTANT (tc)

tc=.14057701495515551037840396020329(ANALYTICAL VALUE)

FEM MODEL	NO.OF QUADRILATERALS (NO.OF T6-TRIANGLES)* NO OF NODES	fem-computed values of tc
MESH-1	3(1)*5	.13083989888840093881392607627265
MESH-2	12(4)*19	.13795314846010888055935252663739
MESH-3	27(9)*37	.13941867439787643949762680667990
MESH-4	48(16)*61	.13993063385734349514300036322959
MESH-5	75(25)*91	.14016582079079076182350837373090
MESH-6	108(36)*127	.14029273101444520386658952876026
MESH-7	147(49)*169	.14036885017128652307484475599502
MESH-8	192(64)*217	.14041804850446187732724063014883
MESH-9	243(81)*271	.14053626365883553894393627363271
MESH-10	300(100) *331	0.140475648374825
MESH-11	363 (121)*397	0.14049335294041
MESH-12	432 (144)*469	0.140506793836505
MESH-13	507 (169)* 547	0.14051723770694
MESH-14	588 (196)* 631	0.140525513575901
MESH-15	675 (225)* 721	0.140532182472916
MESH-16	768 (256)* 817	0.140537635039182
MESH-17	867 (289)* 919	0.140542150048631
MESH-18	972 (324)* 1027	0.140545930756678
MESH-19	1083 (361)*1141	0.140549128181663
MESH-20	1200 (400)*1261	0.140551856423775
MESH-25	1875 (625)*1951	0.140560935627972
MESH-30	2700 (900)* 2791	0.140565858672553
MESH-35	3675 (1225)*3781	0.140568823558516
MESH-40	4800 (1600)*4921	0.14057074624724
MESH-45	6075 (2025)*6211	0.140572063602523
MESH-50	7500 (2500)*7651	0.140573005439404

COMPUTER PROGRAMMES

(1)PROGRAM-1

```
function[]=D2LaplaceEquationQ4Ex3automeshgen(ndiv)
syms coord
nnel=4;
ndof=1;
nc=(ndiv/2)^2;
nnode=(ndiv+1)*(ndiv+2)/2+nc;
nel=3*nc;
sdof=nnode*ndof;
ff=(zeros(sdof,1));ss=(zeros(sdof,sdof));
format long g
for i=1:nel
N(i,1)=i;
end
for i=1:nel
NN(i,1)=i;
end
disp('rr=nodal arrangement  over the  1/8th square cross section')
   [coord,gcoord]=coordinate_rtisoscelestriangle00_h0_hh(ndiv);
   [nodetel,nodes]=nodaladdresses4special_convex_quadrilaterals(ndiv);
   %
   %bcdof=[2;5;3]
   nnn=0;
   for nn=1:nnode
      if coord(nn,1)==(1/2)
         nnn=nnn+1;
         bcdof(nnn,1)=nn;
      end
   end
```

```
   bcdof;
   mm=length(bcdof);

format long g
k1 =double(0.140577014955155510378403960203 29);
 xi=(zeros(nnode,1));
a0=8/pi^3;
for m=1:nnode
   x=(gcoord(m,1));y=(gcoord(m,2));rr=(0);
for n=1:2:99
rr=rr+(-1)^((n-1)/2)*(1-(cosh(n*pi*y)/cosh(n*pi/2)))*cos(n*pi*x)/n^3;
end
xi(m,1)=(a0*rr);
end


 for L=1:nel
   for M=1:3
     LM=nodetel(L,M);
     xx(L,M)=gcoord(LM,1);
     yy(L,M)=gcoord(LM,2);
   end
  end
%_______________________________________________________________
table1=[N nodes];
table2=[N xx yy];
%disp([xx yy])
intJdn1dn1uvrs =[vpa(sym(' .5955276550008211474857292673 30')), vpa(sym('.46832228582057464549116826929 0'));vpa(sym('
.46832228582057464549116826929')), vpa(sym('.595527655000821147485729267330'))];
intJdn1dn2uvrs =[vpa(sym('  -.397018436667214098323819511552')),  vpa(sym('.187785142786283569672554487139 5'));vpa(sym(' -
.312214857213716430327445512860 4')),   vpa(sym('.102981563332785901676180488448'))];
intJdn1dn3uvrs =[vpa(sym(' -.301490781666392950838090244223 5')),vpa(sym(' -.343892571393141784836277243570 0'));vpa(sym(' -
.343892571393141784836277243570 0')), vpa(sym('-.301490781666392950838090244223 5'))];
intJdn1dn4uvrs =[vpa(sym('   .102981563332785901676180488448')), vpa(sym('-.312214857213716430327445512860 4'));vpa(sym('
.187785142786283569672554487139 5')),  vpa(sym('-.397018436667214098323819511552'))];
%
intJdn2dn1uvrs =[vpa(sym('  -.397018436667214098323819511552')), vpa(sym('-.312214857213716430327445512860 4'));vpa(sym('
.187785142786283569672554487139 5')),  vpa(sym(' .102981563332785901676180488448'))];
intJdn2dn2uvrs =[vpa(sym('   .264678957778142732215879674369')), vpa(sym('-.125190095190855713115036324760 0'));vpa(sym(' -
.125190095190855713115036324760 0')),  vpa(sym(' .264678957778142732215879674369'))];
intJdn2dn3uvrs =[vpa(sym('  .200993854444261967225393496149 1')),  vpa(sym('.229261714262094523224184829046 6'));vpa(sym(' -
.270738285737905476775815170953 4')), vpa(sym('-.299006145555738032774606503850 9'))];
intJdn2dn4uvrs =[vpa(sym(' -.686543755551906011174536589 65e-1')),   vpa(sym('.208143238142477620218297008573 4'));vpa(sym('
.208143238142477620218297008573 4')), vpa(sym('-.686543755551906011174536589 65e-1'))];
%
intJdn3dn1uvrs =[vpa(sym(' -.301490781666392950838090244223 5')), vpa(sym('-.343892571393141784836277243570 0'));vpa(sym(' -
.343892571393141784836277243570 0')),vpa(sym(' -.301490781666392950838090244223 5'))];
intJdn3dn2uvrs =[vpa(sym('  .200993854444261967225393496149 1')), vpa(sym('-.270738285737905476775815170953 4'));vpa(sym('
.229261714262094523224184829046 6')), vpa(sym('-.299006145555738032774606503850 9'))];
intJdn3dn3uvrs =[vpa(sym(' .399503072777869016387303251925 4')), vpa(sym('.385369142868952738387907585476 8'));vpa(sym('
.385369142868952738387907585476 8')), vpa(sym('.399503072777869016387303251925 4'))];
intJdn3dn4uvrs =[vpa(sym(' -.299006145555738032774606503850 9')), vpa(sym(' .229261714262094523224184829046 6'));vpa(sym(' -
.270738285737905476775815170953 4')),  vpa(sym('.200993854444261967225393496149 1'))];
%
intJdn4dn1uvrs =[vpa(sym('   .102981563332785901676180488448')),  vpa(sym('.187785142786283569672554487139 5'));vpa(sym(' -
.312214857213716430327445512860 4')), vpa(sym(' -.397018436667214098323819511552'))];
intJdn4dn2uvrs =[vpa(sym(' -.686543755551906011174536589 65e-1')),  vpa(sym(' .208143238142477620218297008573 4'));vpa(sym('
.208143238142477620218297008573 4')), vpa(sym('-.686543755551906011174536589 65e-1'))];
intJdn4dn3uvrs =[vpa(sym(' -.299006145555738032774606503850 9')), vpa(sym('-.270738285737905476775815170953 4'));vpa(sym('
.229261714262094523224184829046 6')),  vpa(sym('.200993854444261967225393496149 1'))];
intJdn4dn4uvrs =[vpa(sym('   .264678957778142732215879674369')), vpa(sym('-.125190095190855713115036324760 0'));vpa(sym(' -
.125190095190855713115036324760 0')),  vpa(sym(' .264678957778142732215879674369'))];
%
%
intJdndn=double([intJdn1dn1uvrs intJdn1dn2uvrs intJdn1dn3uvrs intJdn1dn4uvrs;...
        intJdn2dn1uvrs intJdn2dn2uvrs intJdn2dn3uvrs intJdn2dn4uvrs;...
        intJdn3dn1uvrs intJdn3dn2uvrs intJdn3dn3uvrs intJdn3dn4uvrs;...
        intJdn4dn1uvrs intJdn4dn2uvrs intJdn4dn3uvrs intJdn4dn4uvrs]);



%
for iel=1:nel
index=zeros(nnel*ndof,1);

X=xx(iel,1:3);
Y=yy(iel,1:3);
%disp([X Y])
xa=X(1,1);
xb=X(1,2);
```

```
xc=X(1,3);
ya=Y(1,1);
yb=Y(1,2);
yc=Y(1,3);
bta=yb-yc;btb=yc-ya;
gma=xc-xb;gmb=xa-xc;
delabc=gmb*bta-gma*btb;
G=[bta btb;gma gmb]/delabc;
GT=[bta gma;btb gmb]/delabc;
Q=GT*G;
sk(1:4,1:4)=(zeros(4,4));SK(1:8,1:8)=(zeros(8,8));
for i=1:4
 for j=i:4
 sk(i,j)=(delabc*sum(sum(Q.*(intJdndn(2*i-1:2*i,2*j-1:2*j)))));
 sk(j,i)=sk(i,j);
 end
end
f =[5/144;1/24;7/144;1/24]*(2*delabc);


%_______________________________________
 edof=nnel*ndof;
 k=0;
 for i=1:nnel
     nd(i,1)=nodes(iel,i);
     start=(nd(i,1)-1)*ndof;
     for j=1:ndof
         k=k+1;
         index(k,1)=start+j;
     end
 end
 %-------------------------------------------------------------------------
for i=1:edof
    ii=index(i,1);
    ff(ii,1)=ff(ii,1)+f(i,1);
 for j=1:edof
     jj=index(j,1);
     ss(ii,jj)=ss(ii,jj)+sk(i,j);
 end
end
end%for iel
 %-------------------------------------------------------------------------
%bcdof=[13;37;35;33;31;29;27;25;23;21;19;17;15];
for ii=1:mm
    kk=bcdof(ii,1);
    ss(kk,1:nnode)=zeros(1,nnode);
    ss(1:nnode,kk)=zeros(nnode,1);
    ff(kk,1)=0;
end
for ii=1:mm
    kk=bcdof(ii,1);
    ss(kk,kk)=1;
end
phi=ss\ff;
for I=1:nnode
NN(I,1)=I;
phi_xi(I,1)=phi(I,1)-xi(I,1);
end
MAXPHI_XI=max(abs(phi_xi));
%disp('_________________________________________________________________')
%disp('number of nodes,elements & nodes per element')
%[nnode nel nnel ndof]
%disp('element number    nodal connectivity for quadrilateral element')
%table1
%disp('_________________________________________________________________________________')
%disp('element number   coordinates of the triangle spanning the quadrilateral element')
%table2
disp('_____________________________________________________________________________________________________')
disp('                              Prandtl Stress Values')
disp('              node number       computed values          anlytical(theoretical)       error          ')
disp('_____________________________________________________________________________________________________')
disp([NN phi xi phi_xi])
disp('_____________________________________________________________________________________________________')

t=0;
 for iii=1:nnode
    t=t+phi(iii,1)*ff(iii,1);
end
T=8*t;
```

```
disp('                              torisonal constants')
disp('                fem=phi              exact=xi        error=(max(abs(phi_xi))')
disp('-----------------------------------------------------------------------------------------------------------')
disp([T k1 MAXPHI_XI ])
disp('-----------------------------------------------------------------------------------------------------------')
disp('number of nodes,elements & nodes per element')
disp([nnode nel nnel ])
disp('-----------------------------------------------------------------------------------------------------------')
```

(2)PROGRAM-2

```
function[coord,gcoord]=coordinate_rtisoscelestriangle00_h0_hh(n)
%cartesian coordinates ((xi,yi),i=1,2,3) for the right isosceles triangle
%with vertices (x1,y1)=(0,0),(x2,y2)=(1/2,0) and (x3,y3)=(1/2,1/2)
syms ui vi wi xi yi
[ui,vi,wi]=coordinates_stdtriangle(n);
[eln,spqd]=nodaladdresses_special_convex_quadrilaterals(n);
qq=(n+1)*(n+2)/2;
   nc=(n/2)^2;
for pp=1:nc
 qq=qq+1;
 q1=eln(pp,1);
 q2=eln(pp,2);
 q3=eln(pp,3);
 ui(qq,1)=(ui(q1,1)+ui(q2,1)+ui(q3,1))/3;
 vi(qq,1)=(vi(q1,1)+vi(q2,1)+vi(q3,1))/3;
 wi(qq,1)=1-ui(qq,1)-vi(qq,1);
end

%disp([ui vi wi])
N=length(ui);
if N==(n+1)*(n+2)/2+nc
   NN=(1:N)';
for i=1:N
   xi(i,1)=(ui(i,1)+vi(i,1))/2;
   yi(i,1)=vi(i,1)/2;
end
%disp('_________________________________________________')
%disp('NN   xi    yi')
%disp([NN xi yi])
%disp('_________________________________________________')
else
   disp('ERROR')
end
coord(:,1)=(xi(:,1));
coord(:,2)=(yi(:,1));
gcoord(:,1)=double(xi(:,1));
gcoord(:,2)=double(yi(:,1));
%disp(gcoord)
```

(3)PROGRAM-3

```
function[nodetel,nodes]=nodaladdresses4special_convex_quadrilaterals(n)
%eln=6-node triangles with centroid
%spqd=4-node special convex quadrilateral
%n must be even,i.e.n=2,4,6,.......
syms mst_tri x
%disp('vertex nodes of triangle')
elm(1,1)=1;
elm(n+1,1)=2;
elm((n+1)*(n+2)/2,1)=3;
%disp('vertex nodes of triangle')
kk=3;
for k=2:n
   kk=kk+1;
   elm(k,1)=kk;
end
%disp('left edge nodes')
nni=1;
for i=0:(n-2)
   nni=nni+(n-i)+1;
   elm(nni,1)=3*n-i;
end
%disp('right edge nodes')
nni=n+1;
for i=0:(n-2)
   nni=nni+(n-i);
   elm(nni,1)=(n+3)+i;
end

%disp('interior nodes')
nni=1;jj=0;
```

```
for i=0:(n-3)
   nni=nni+(n-i)+1;
   for j=1:(n-2-i)
      jj=jj+1;
      nnj=nni+j;
      elm(nnj,1)=3*n+jj;
   end
end
%disp(elm)
%disp(length(elm))

jj=0;kk=0;
for j=0:n-1
   jj=j+1;
for k=1:(n+1)-j
   kk=kk+1;
   row_nodes(jj,k)=elm(kk,1);
end
end
row_nodes(n+1,1)=3;
%for jj=(n+1):-1:1
%    (row_nodes(jj,:));
%end
%[row_nodes]
rr=row_nodes;
%rr
%disp('element computations')
if rem(n,2)==0
ne=0;N=n+1;

for k=1:2:n-1
N=N-2;
i=k;
for j=1:2:N
   ne=ne+1;
eln(ne,1)=rr(i,j);
eln(ne,2)=rr(i,j+2);
eln(ne,3)=rr(i+2,j);
eln(ne,4)=rr(i,j+1);
eln(ne,5)=rr(i+1,j+1);
eln(ne,6)=rr(i+1,j);
end%i
%me=ne
%N-2
if (N-2)>0
for jj=1:2:N-2
ne=ne+1;
eln(ne,1)=rr(i+2,jj+2);
eln(ne,2)=rr(i+2,jj);
eln(ne,3)=rr(i,jj+2);
eln(ne,4)=rr(i+2,jj+1);;
eln(ne,5)=rr(i+1,jj+1);
eln(ne,6)=rr(i+1,jj+2);
end%jj
end
end%k
end
%ne
%for kk=1:ne
%[eln(kk,1:6)]
%end
%add node numbers for element centroids

nnd=(n+1)*(n+2)/2;
for kkk=1:ne
   nnd=nnd+1;
   eln(kkk,7)=nnd;
end
%for kk=1:ne
%[eln(kk,1:7)]
%end
%to generate special quadrilaterals
% and the spanning triangle
mm=0;
for iel=1:ne
   for jel=1:3
   mm=mm+1;
      switch jel
       case 1
        nodes(mm,1:4)=[eln(iel,7) eln(iel,6) eln(iel,1) eln(iel,4)];
```

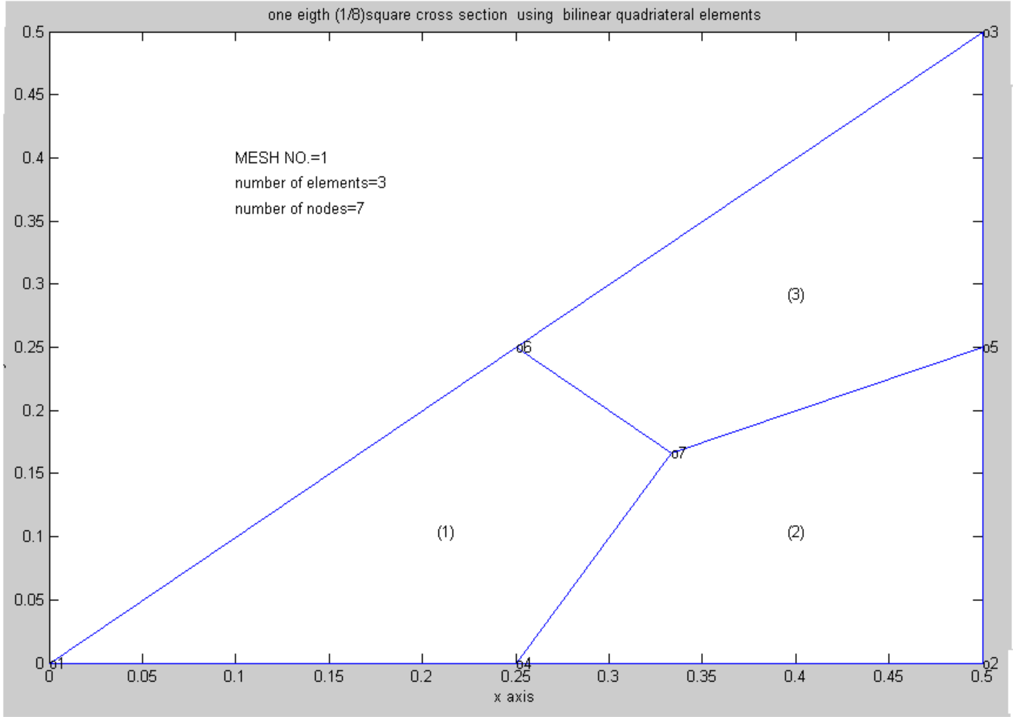
```
        nodetel(mm,1:3)=[eln(iel,2) eln(iel,3) eln(iel,1)];
        case 2
        nodes(mm,1:4)=[eln(iel,7) eln(iel,4) eln(iel,2) eln(iel,5)];
        nodetel(mm,1:3)=[eln(iel,3) eln(iel,1) eln(iel,2)];
        case 3
        nodes(mm,1:4)=[eln(iel,7) eln(iel,5) eln(iel,3) eln(iel,6)];
        nodetel(mm,1:3)=[eln(iel,1) eln(iel,2) eln(iel,3)];
       end
    end
end
%for mmm=1:mm
   %spqd(:,1:4)
 %end
%
%ss1='number of 6-node triangles with centroid=';
%[p1,q1]=size(eln);
%disp([ss1 num2str(p1)])
%
%eln
%
%ss2='number of 4-node special convex quadrilaterals =';
%[p2,q2]=size(spqd);
%disp([ss2 num2str(p2)])
%
%spqd
```

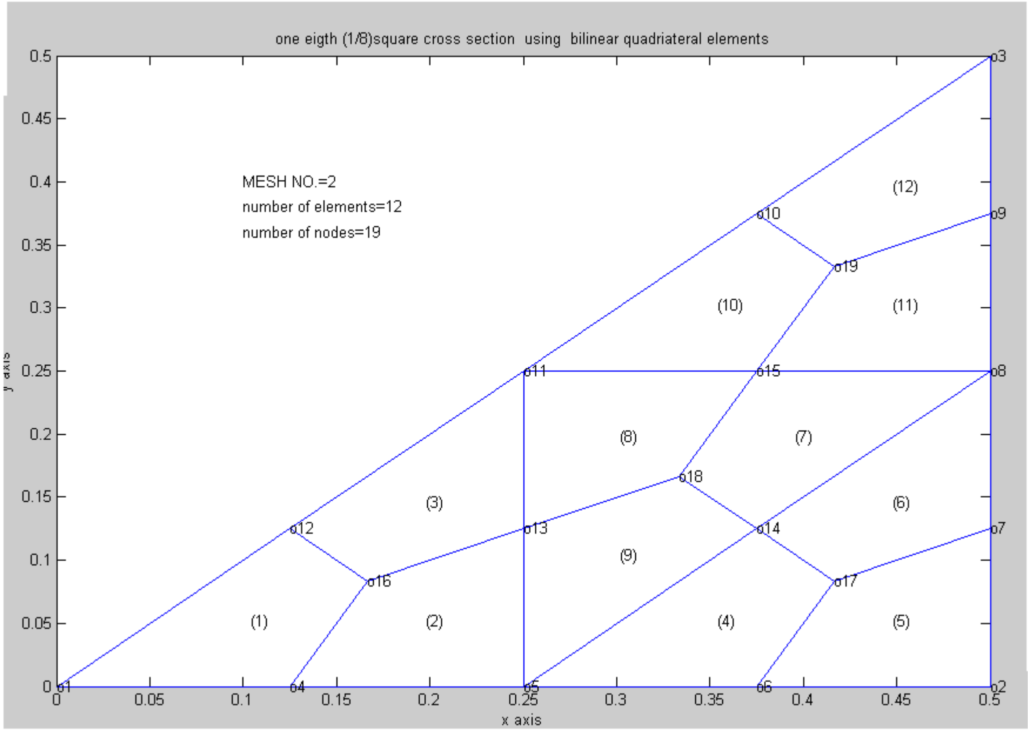
(4)PROGRAM-4

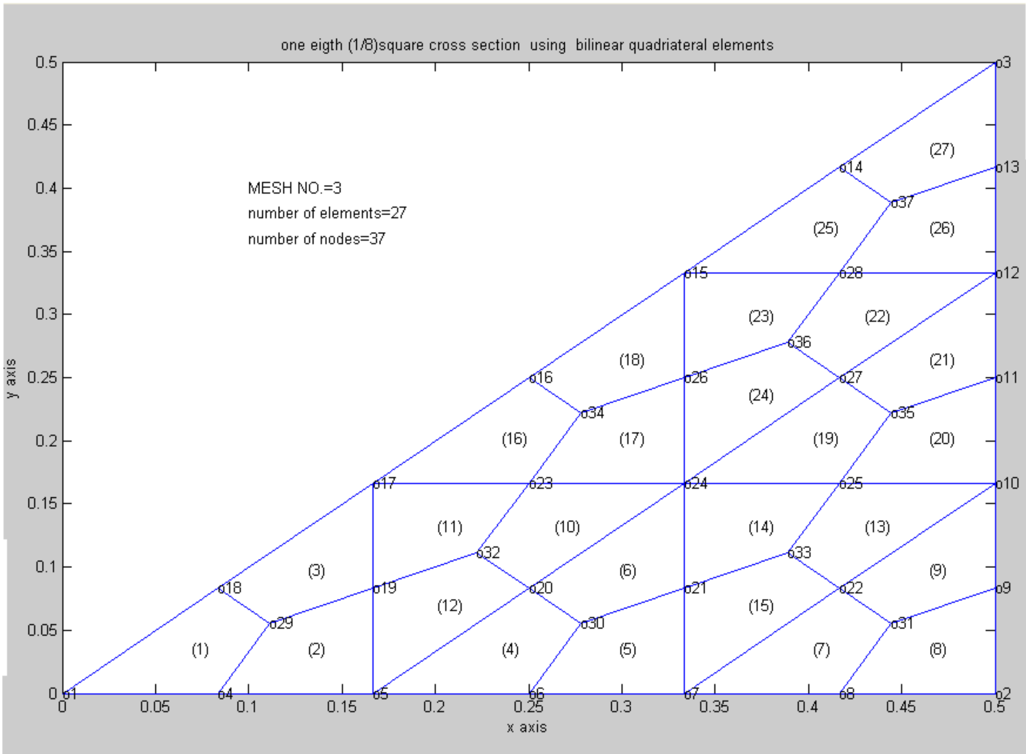
```
function[]=quadrilateralmesh_square_cross_section_q4(nmesh)
%skip=0 or 1
%skip=0,generates  meshes for the nodal and coordinate data given for ten cases
%skip=1,generates  meshes automatically by dividing sides of triangle into equal sizes of 2,4,6,8,etc......
clf
skip=1;
for mesh=1:nmesh

      figure(mesh)
   ndiv=mesh*2;
 [coord,gcoord]=coordinate_rtisoscelestriangle00_h0_hh(ndiv);
   [nodetel,nodes]=nodaladdresses4special_convex_quadrilaterals(ndiv);
end

[nel,nnel]=size(nodes)
[nnode,dimension]=size(gcoord)
%plot the mesh for the generated data
%x and y coordinates
xcoord(1:nnode,1)=gcoord(1:nnode,1);
ycoord(1:nnode,1)=gcoord(1:nnode,2);
%extract coordinates for each element

for i=1:nel
for j=1:nnel
x(1,j)=xcoord(nodes(i,j),1);
y(1,j)=ycoord(nodes(i,j),1);
end;%j loop
xvec(1,1:5)=[x(1,1),x(1,2),x(1,3),x(1,4),x(1,1)];
yvec(1,1:5)=[y(1,1),y(1,2),y(1,3),y(1,4),y(1,1)];
%axis equal
axis tight
xmin=0;xmax=1/2;ymin=0;ymax=1/2;
axis([xmin,xmax,ymin,ymax]);
plot(xvec,yvec);%plot element
hold on;
%place element number
if mesh<6
midx=mean(xvec(1,1:4));
midy=mean(yvec(1,1:4));
text(midx,midy,['(',num2str(i),')']);
end
end;%i loop
xlabel('x axis')
ylabel('y axis')
st1='one eigth (1/8)square cross section ';
st2=' using  ';
st3='bilinear ';
st4='quadriateral';
st5=' elements'
title([st1,st2,st3,st4,st5])
text(0.1,0.4,['MESH NO.=',num2str(mesh)])
text(0.1,0.38,['number of elements=',num2str(nel)])
text(0.1,0.36,['number of nodes=',num2str(nnode)])
```

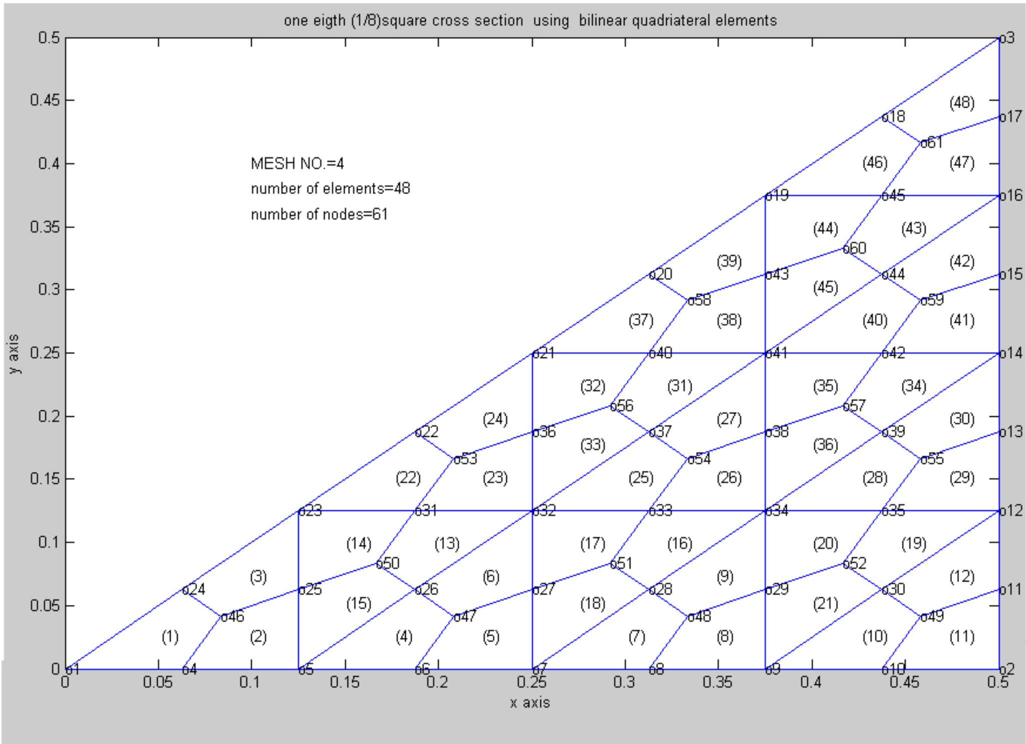
```
%put node numbers
if mesh<6
for jj=1:nnode
text(gcoord(jj,1),gcoord(jj,2),['o',num2str(jj)]);
end
end
hold on
%axis off
end%for nmesh-the number of meshes
```

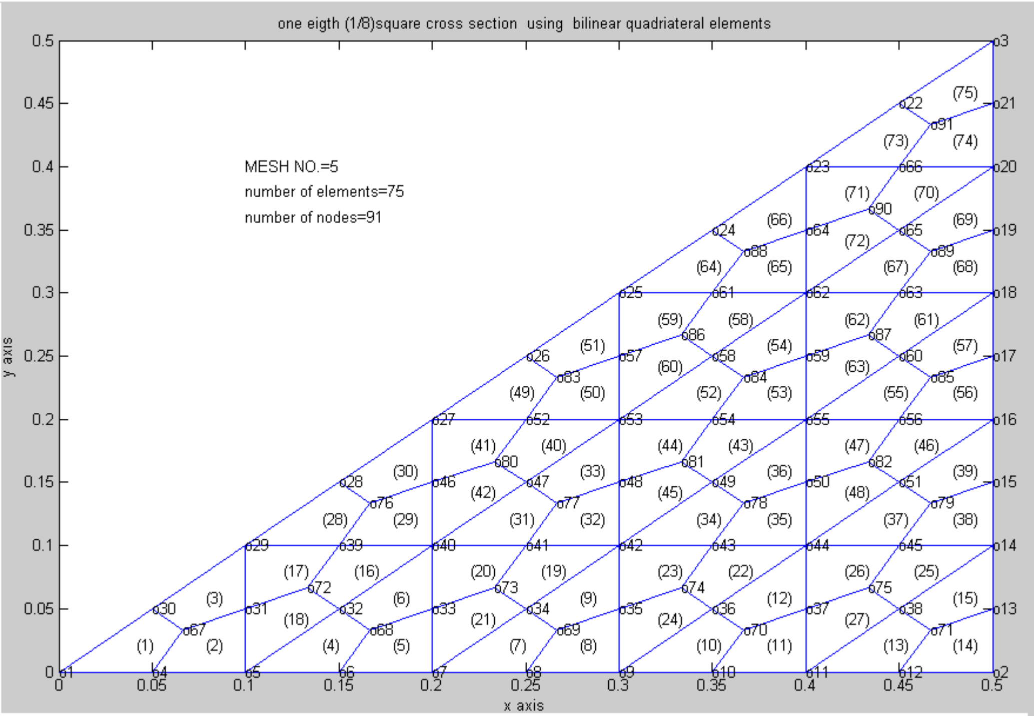
FIGURES

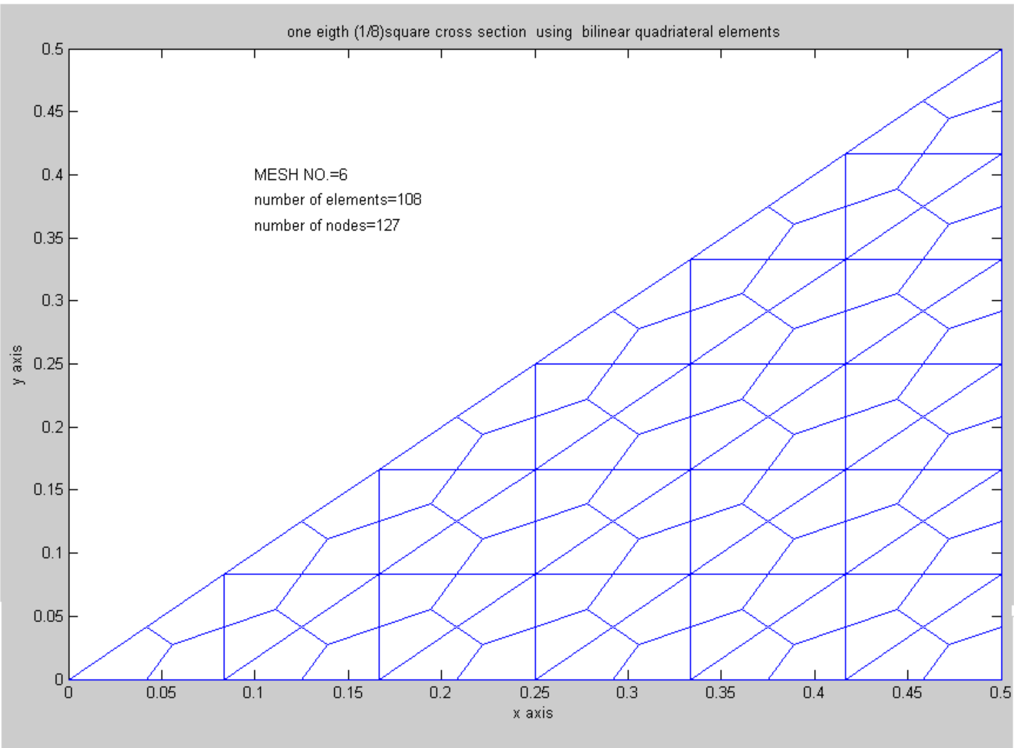

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=1
number of elements=3
number of nodes=7
(1)
(2)
(3)
x axis

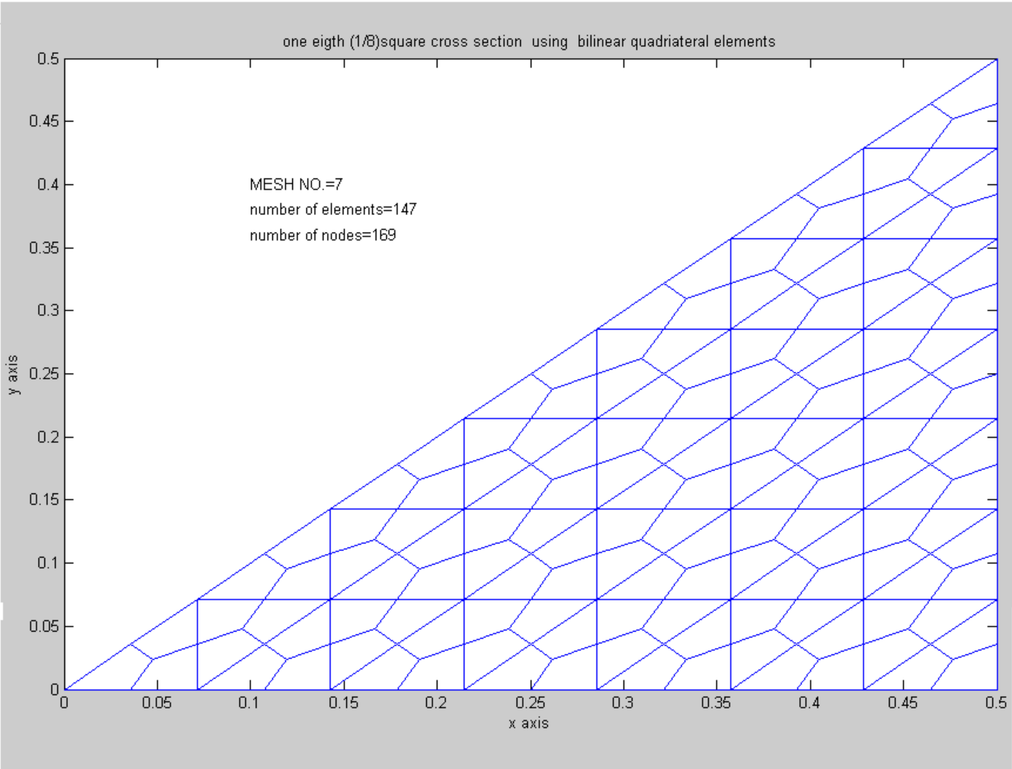

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=2
number of elements=12
number of nodes=19
x axis


one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=3
number of elements=27
number of nodes=37
y axis
x axis

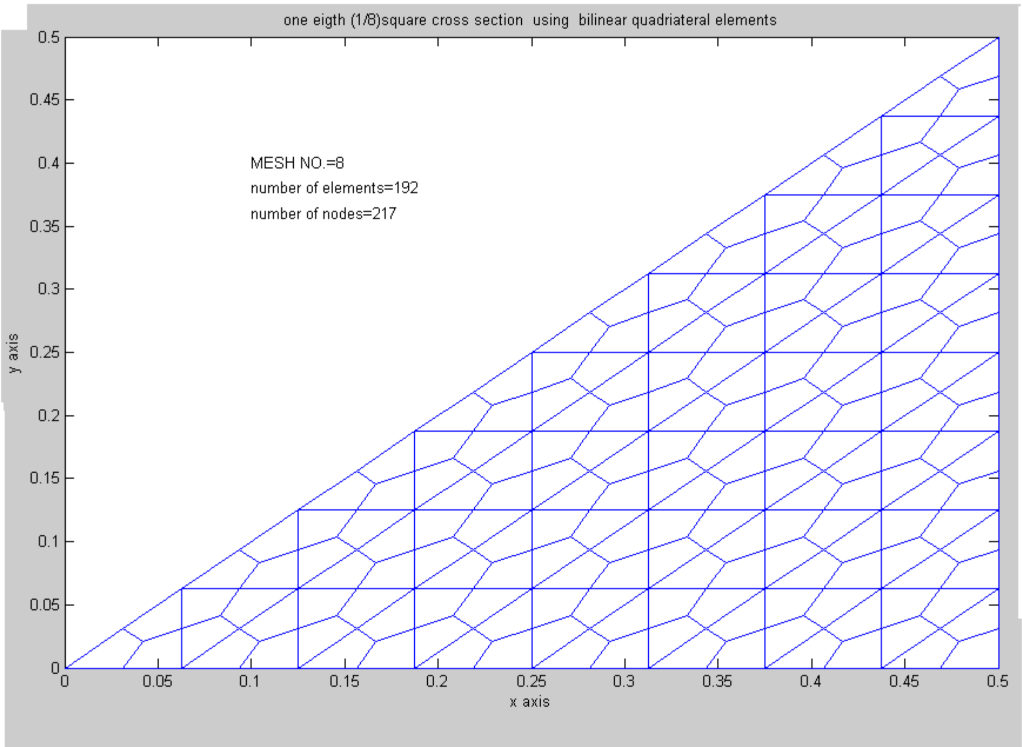

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=4
number of elements=48
number of nodes=61
y axis
x axis


one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=5
number of elements=75
number of nodes=91
y axis
x axis
0 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45 0.5

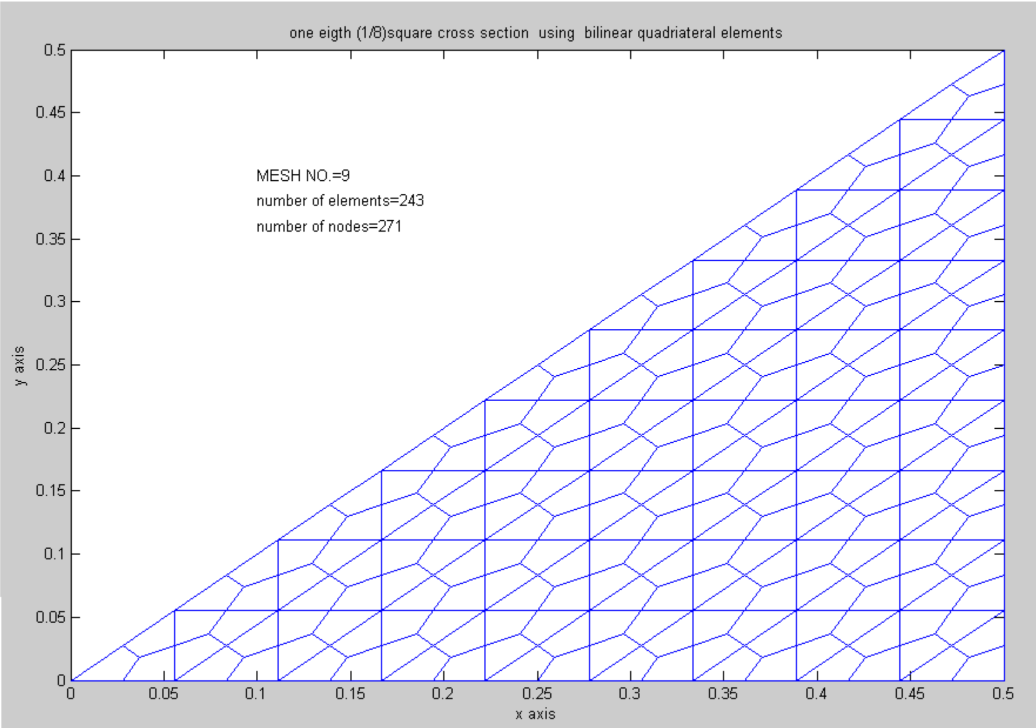

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=6
number of elements=108
number of nodes=127
y axis
x axis
0 0.05 0.1 0.15 0.2 0.25 0.3 0.35 0.4 0.45 0.5

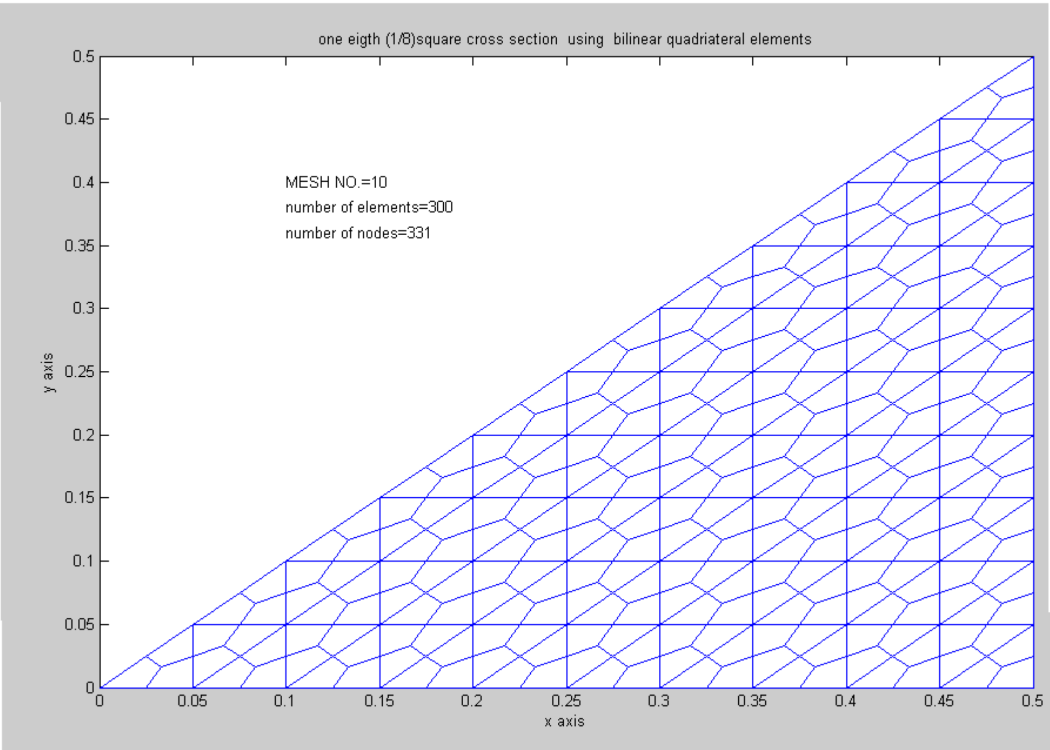


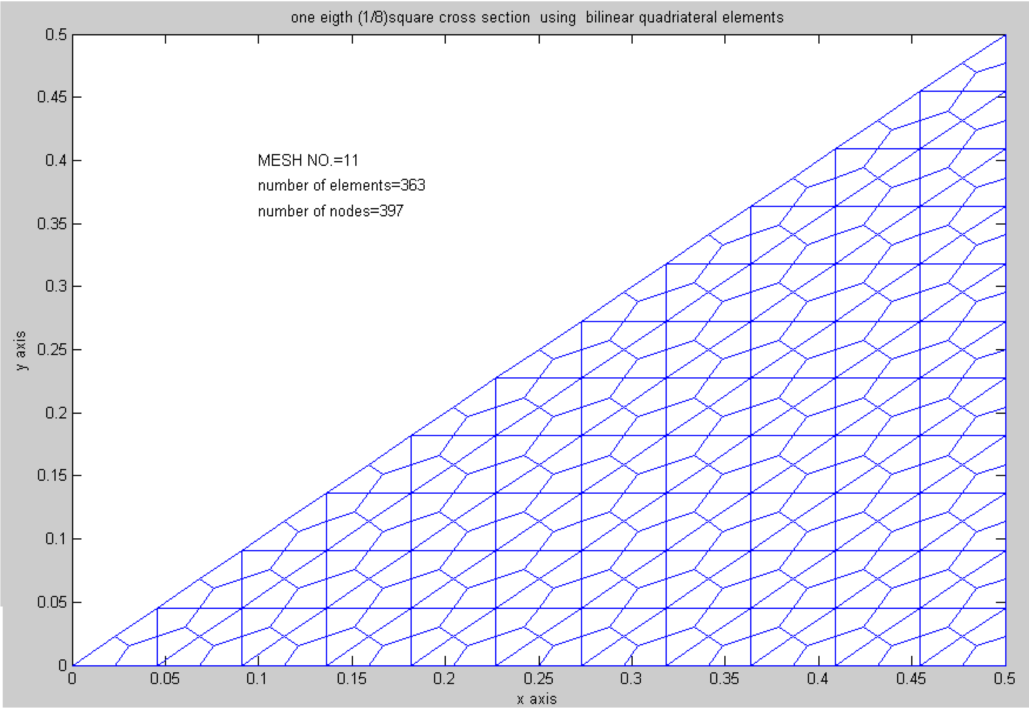
one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=7
number of elements=147
number of nodes=169
y axis
x axis

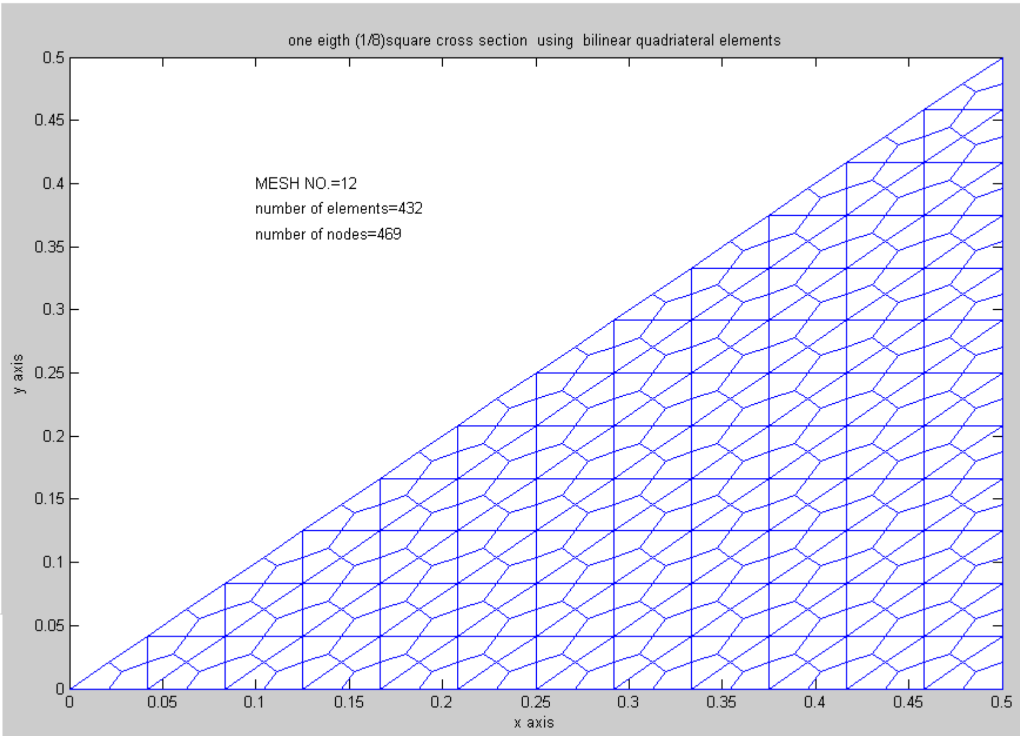


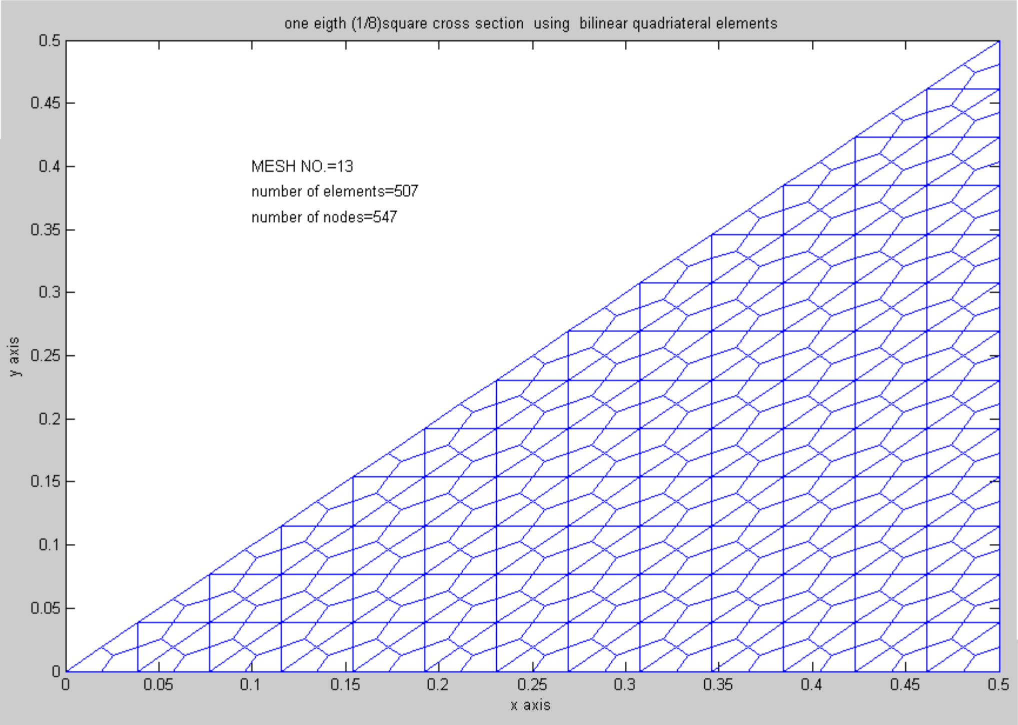
one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=8
number of elements=192
number of nodes=217
y axis
x axis

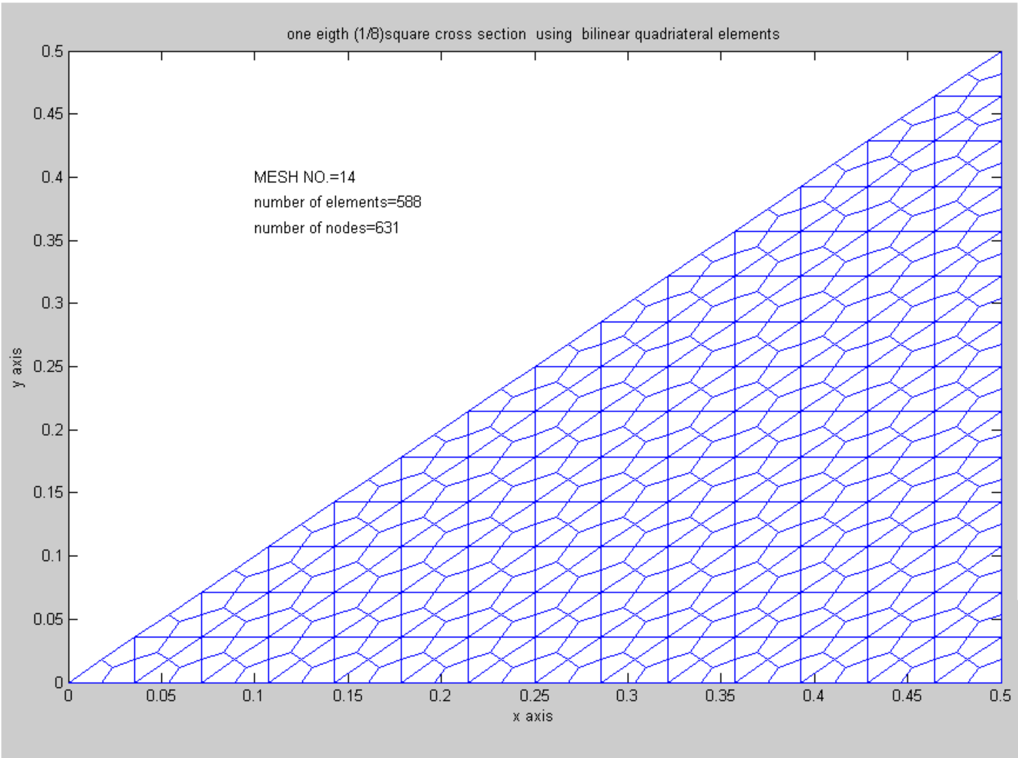

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=9
number of elements=243
number of nodes=271
y axis
x axis

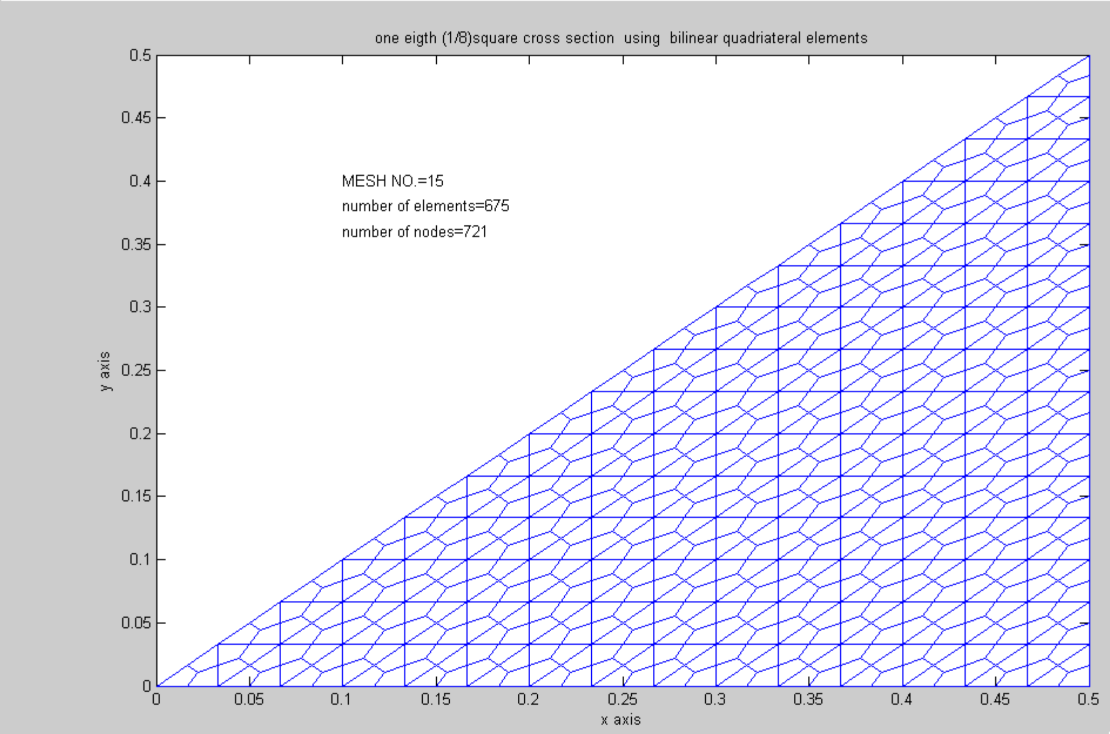

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=10
number of elements=300
number of nodes=331
y axis
x axis

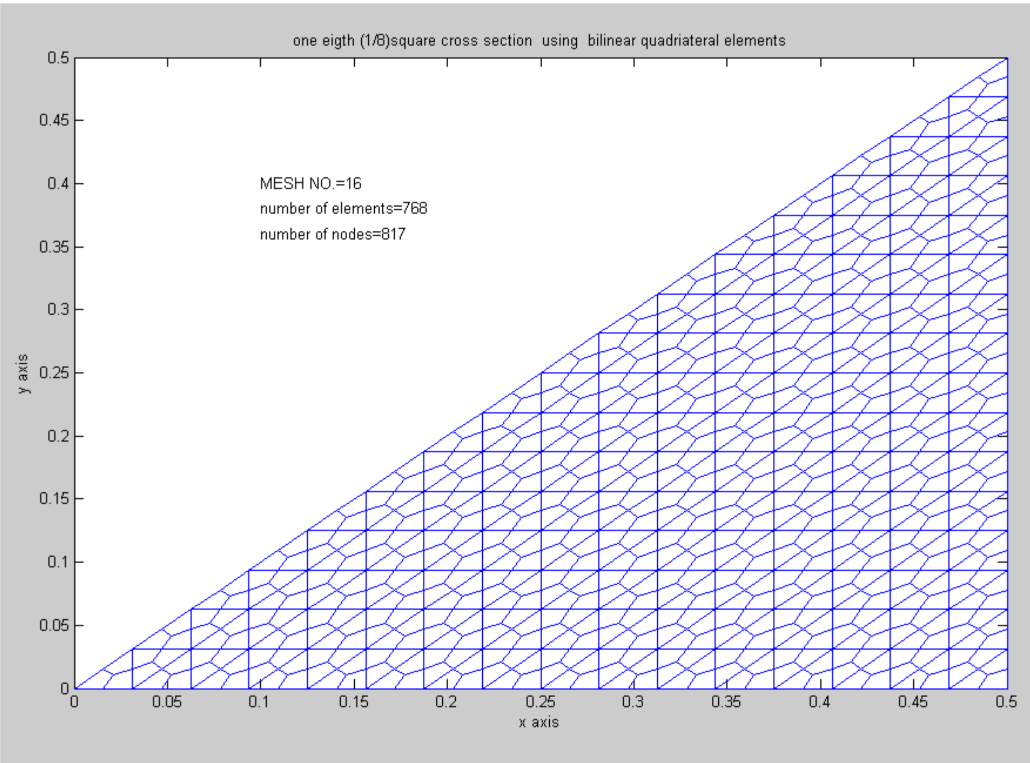

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=11
number of elements=363
number of nodes=397
y axis
x axis
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5

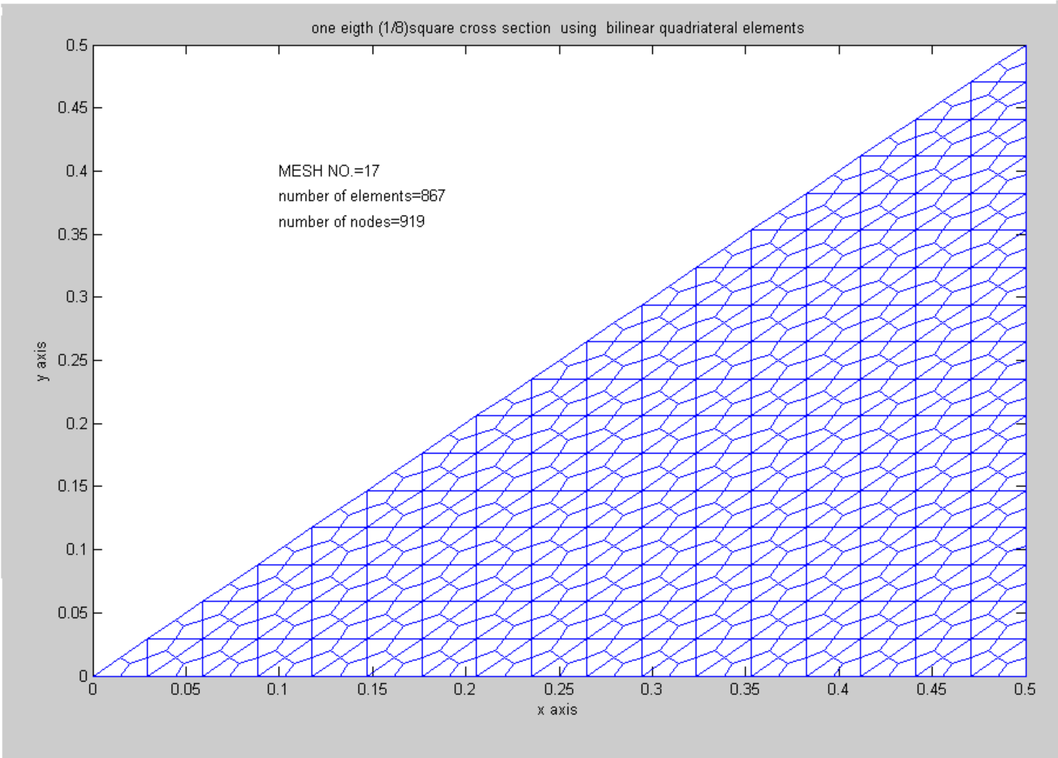

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=12
number of elements=432
number of nodes=469
y axis
x axis
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5

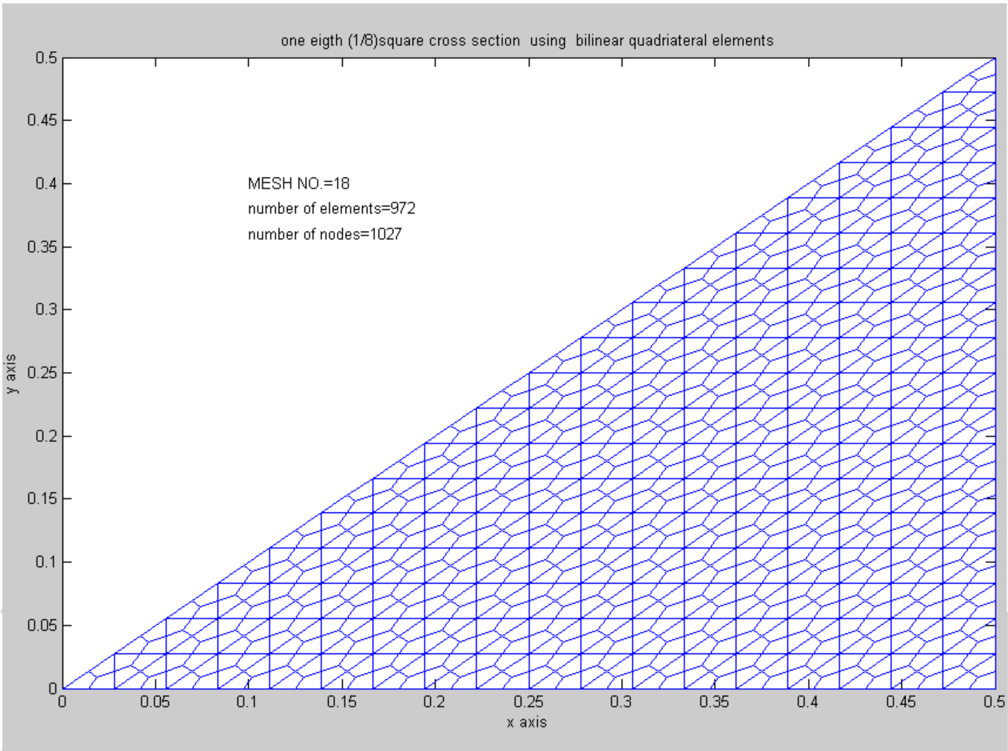

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=13
number of elements=507
number of nodes=547
y axis
x axis

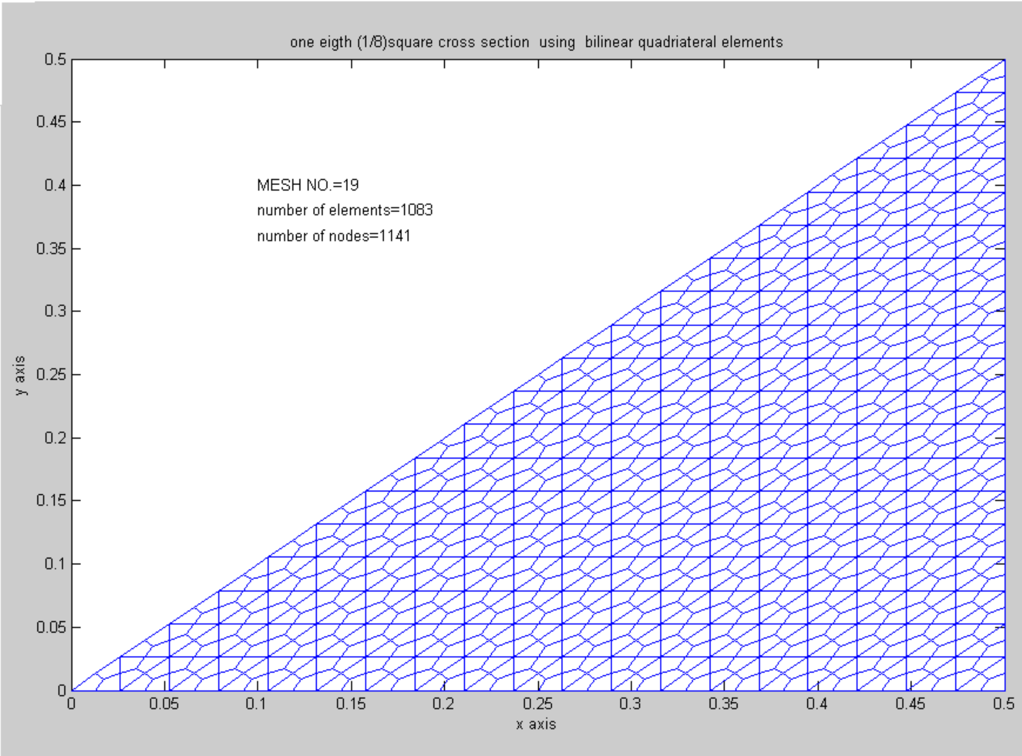

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=14
number of elements=588
number of nodes=631
y axis
x axis

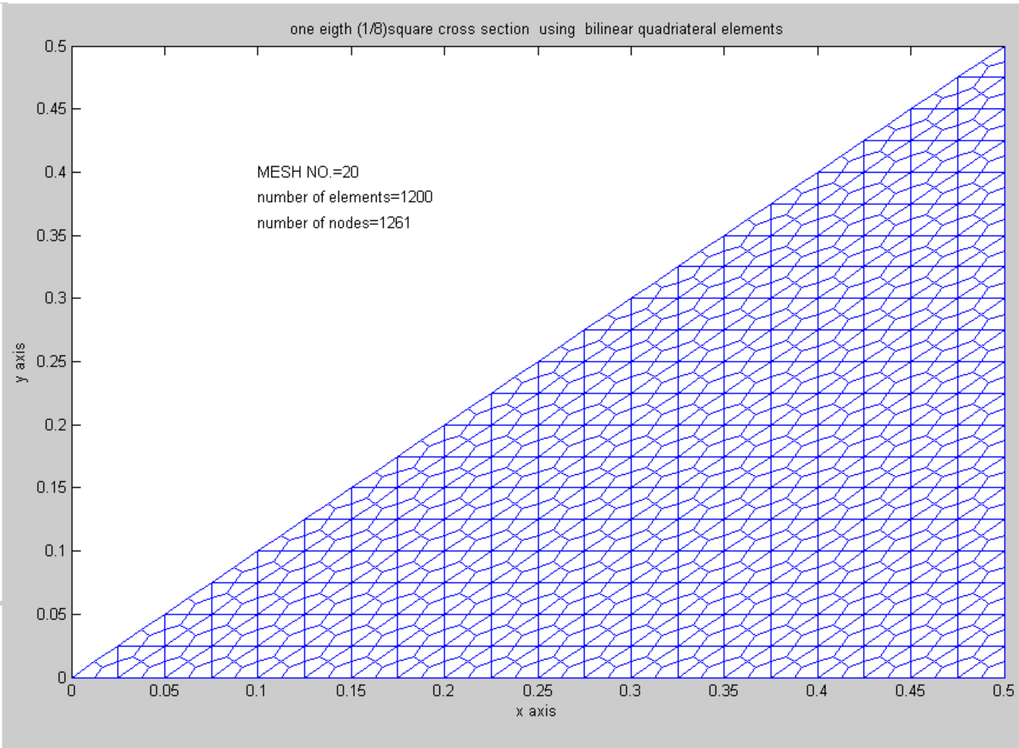

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=15
number of elements=675
number of nodes=721
y axis
x axis

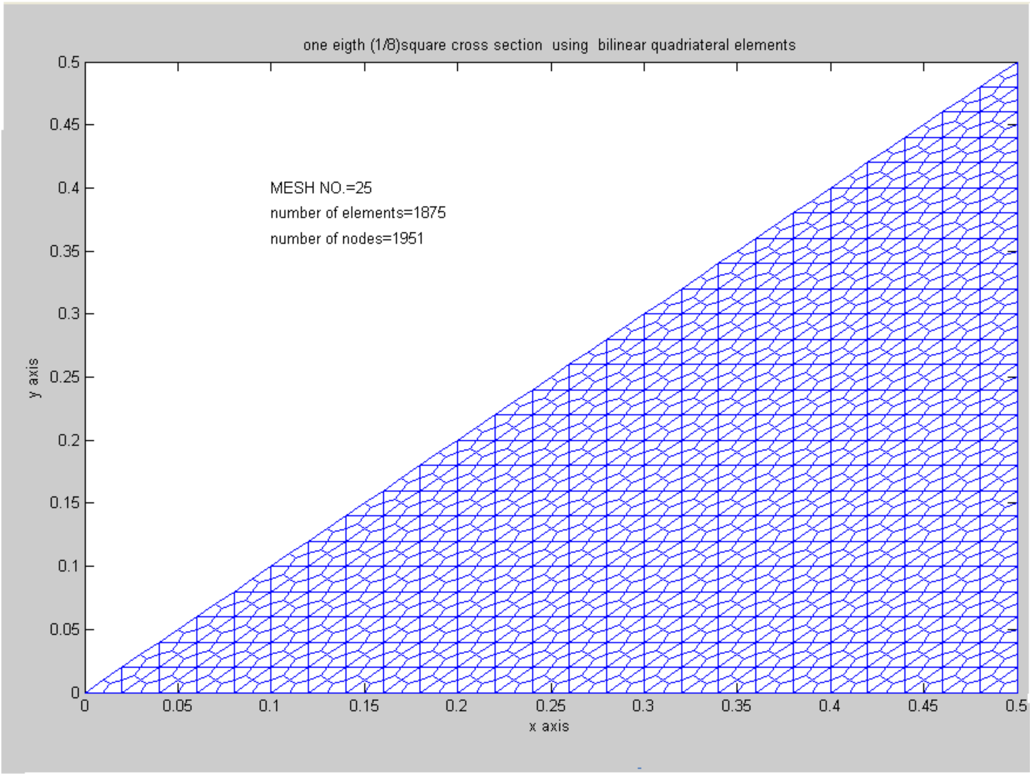

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=16
number of elements=768
number of nodes=817
y axis
x axis

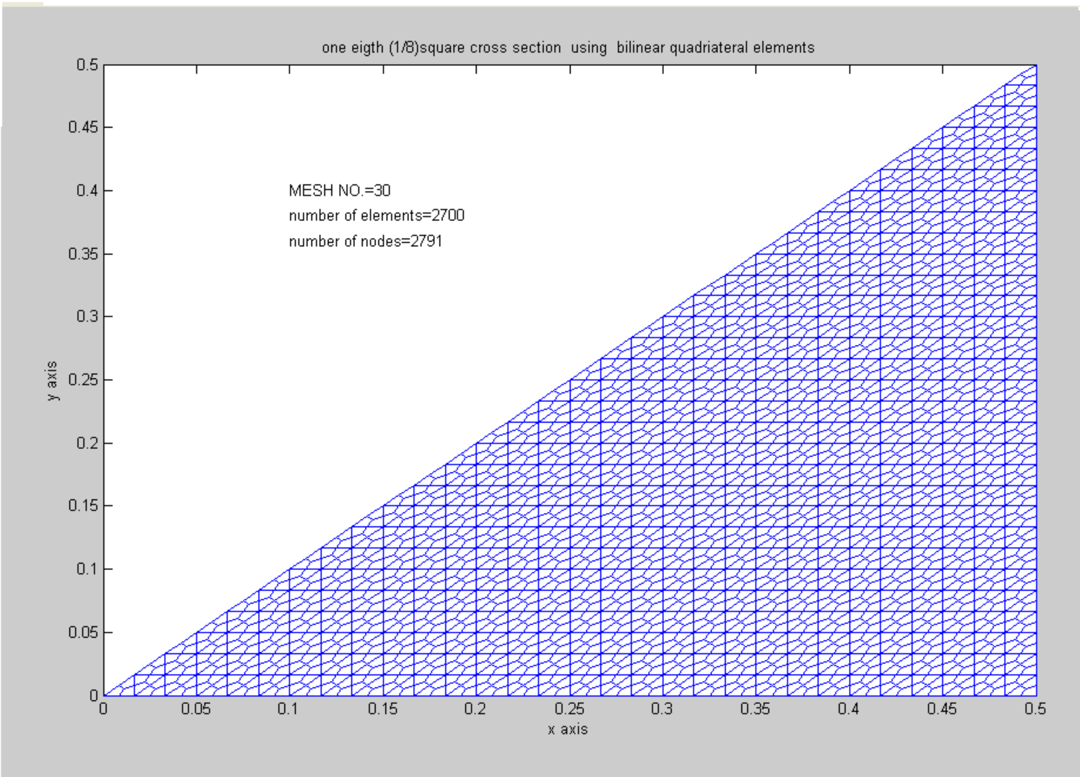

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=17
number of elements=867
number of nodes=919
y axis
x axis

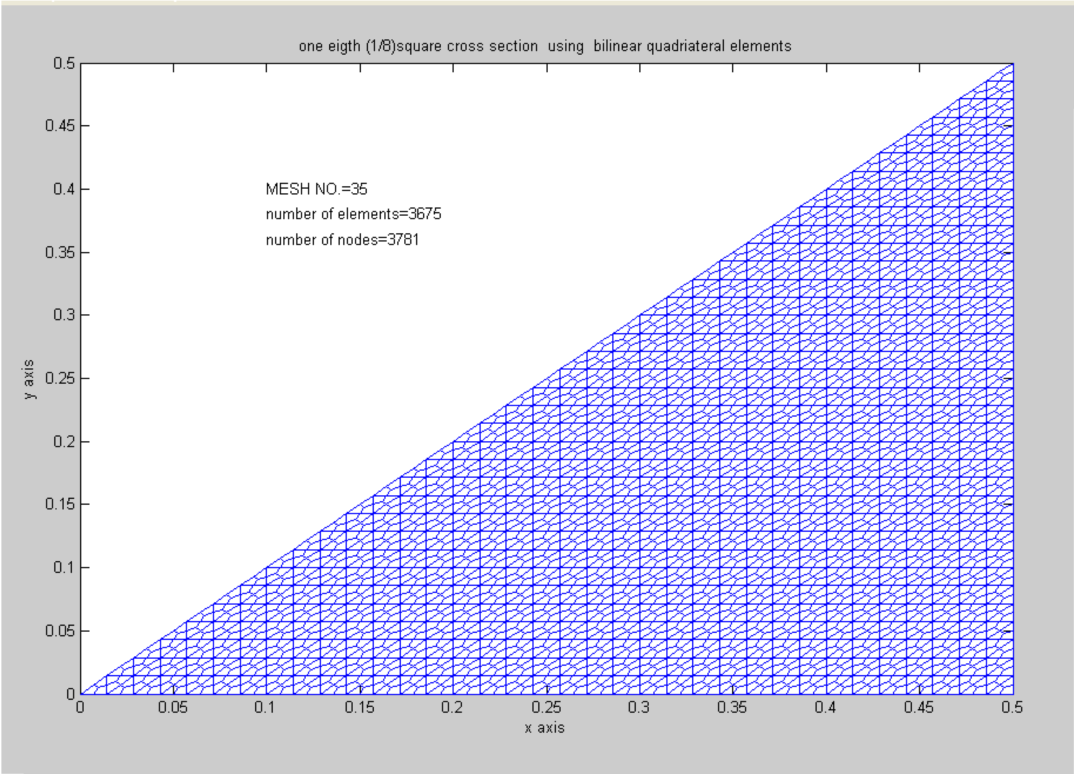

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=18
number of elements=972
number of nodes=1027
y axis
x axis

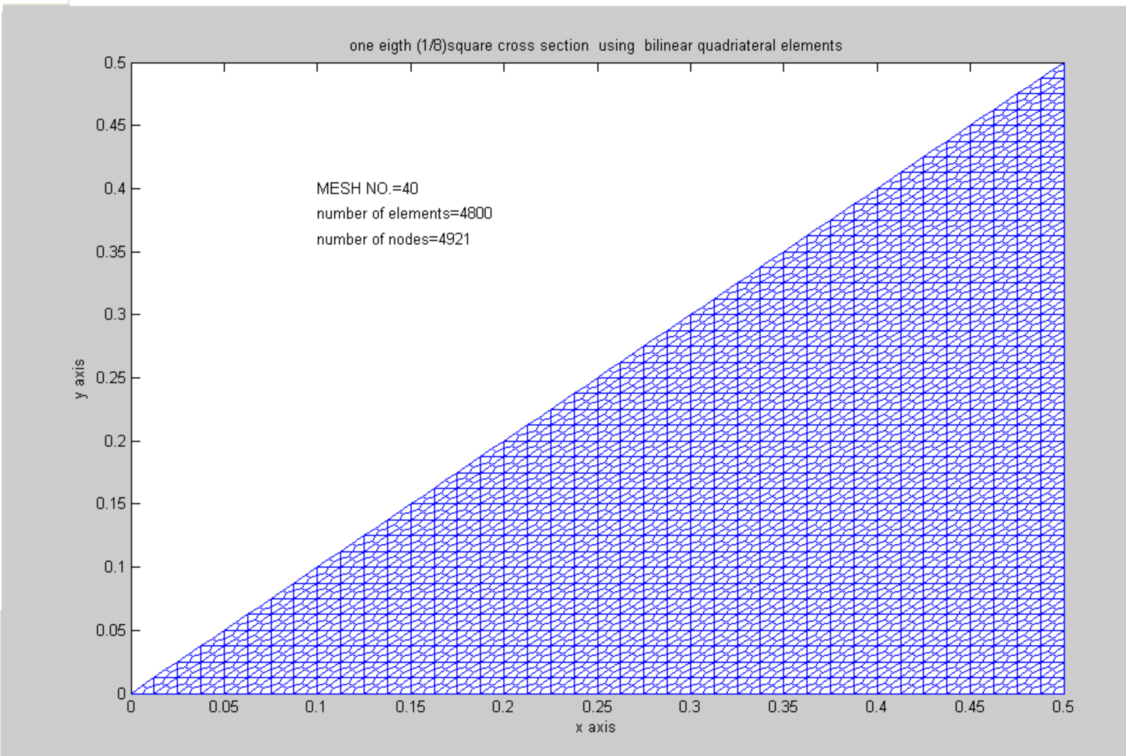

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=19
number of elements=1083
number of nodes=1141
y axis
x axis

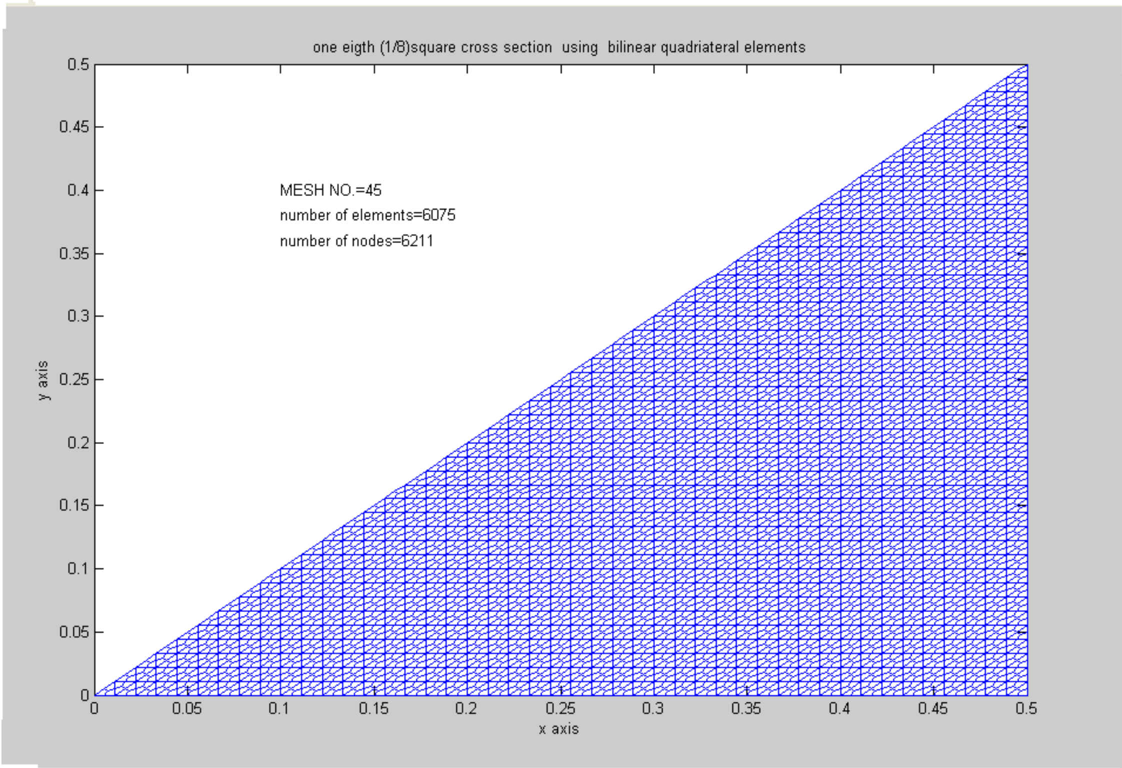

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=20
number of elements=1200
number of nodes=1261
y axis
x axis

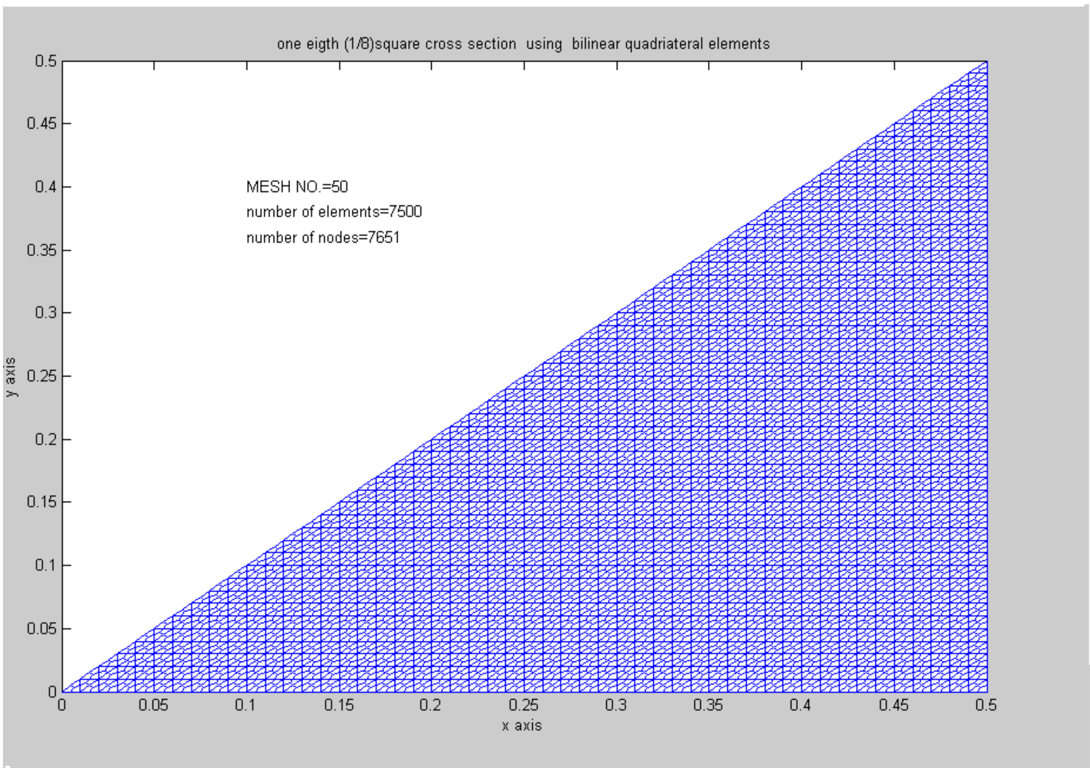

one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=25
number of elements=1875
number of nodes=1951
y axis
x axis
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5


one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=30
number of elements=2700
number of nodes=2791
y axis
x axis
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5


one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=35
number of elements=3675
number of nodes=3781
y axis
x axis
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5


one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=40
number of elements=4800
number of nodes=4921
y axis
x axis
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5


one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=45
number of elements=6075
number of nodes=6211
y axis
x axis
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5


one eigth (1/8)square cross section using bilinear quadriateral elements
MESH NO.=50
number of elements=7500
number of nodes=7651
y axis
x axis
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0.5