www.ijecs.in
International Journal Of Engineering And Computer Science
Volume 9 Issue 10 October 2020, Page No. 25208-25239
ISSN: 2319-7242 DOI: 10.18535/ijecs/v9i10.4536

A New Approach to Automatic Mesh Generation over Polygonal Domains with Linear, Quadratic, Cubic and Quartic Order Quadrilaterals of Serendipity, Lagrange and Complete Lagrange Finite Elements

[1]**H.T.Rathod**[*], [2]**Bharath Rathod,** [3]**K.Suguntha Devi**

[1,3]Department of Mathematics Jnana Bharathi Campus, Bangalore University, Bangalore -560056, Karnataka state, India
[2]PGDFM student IIFM Bhopal,MPstate INDIA

Abstract
This paper presents a novel mesh generation scheme of all quadrilateral elements over a linear polygonal domain. We first decompose the linear polygon into simple sub regions in the shape of quadrilaterals. These simple regions are then quadrangulated to generate first into a fine mesh of four node quadrilateral elements using bilinear transformations.We propose then an automatic quadrilateral conversion scheme. Each four node quadrilateral is converted to an 8-node,9-node,12-node ,16-node,17-node and 25-noded quadrilaterals by inserting the midside nodes appropriately. Examples are presented to illustrate the simplicity and efficiency of the new mesh generation method for standard and arbitrary shaped domains. We have appended two important MATLAB programs which incorporate the mesh generation scheme for the 17-noded complete Lagrange elements developed in this paper.Other MATLAB programs can be coded on similar lines. These programs provide valuable output on the nodal coordinates,element connectivity and graphic display of the all quadrilateral meshes for application to finite element analysis.

Keywords: Finite elements of Serendipity and Lagrange families,quadrilateral mesh generation, convex and nonconvex polygonal domains, uniform refinement, quadrangulation and triangulation.

1. Introduction

The finite element method (FEM**)** is nowadays the most powerful computational tool in science and engineering applications which has originated from the pioneering works of Courant[1],Argyris[2] and Clough[3]. Finite Element Analysis (FEA) is widely used in many fields including structures and optimization. The FEA in engineering applications comprises three phases: domain discretization, equation solving and error analysis. The domain discretization or mesh generation is the preprocessing phase which plays an important role in the achievement of accurate solutions.

FEM requires dividing the analysis region into many sub regions. These small regions are the elements which are connected with adjacent elements at their nodes. Mesh generation is a procedure of generating the geometric data of the elements and their nodes, and involves computing the coordinates of nodes, defining their connectivity and thus constructing the elements. Hence mesh designates aggregates of elements nodes and lines representing their connectivity. Though the FEM is a powerful and versatile tool, its usefulness is often hampered by the need to generate a mesh. Creating a mesh is the first step in a wide range of applications, including scientific and engineering computing and computer graphics. But generating a mesh can be very time consuming and prone to error if done manually. In recognition of this problem a large number of methods have been devised to automate the mesh generation task. An attempt to create a fully automatic mesh generator that is capable of generating valid finite element meshes over arbitrary complex domains and needs only the information of the specified geometric boundary of the domain and the element size, started from the pioneering work [4] in the early 1970's. Since then many methodologies have been proposed and different algorithms have been devised in the development of automatic mesh generators [5-7]. In order to perform a reliable finite element simulation a number of researchers [8-10] have made efforts to develop adaptive FEA method which integrates with error estimation and automatic mesh modification. Traditionally adaptive mesh generation process is started from coarse mesh which gives large discretization error levels and takes a lot of iterations to get a desired final mesh. The research literature on the subject is vast and different techniques have been proposed [11], as several engineering applications to real world problems cannot be defined on a rectangular domain or solved on a structured square mesh. The description and discretization of the design domain geometry, specification of the boundary conditions for the governing state equation, and accurate computation of the design response may require the use of unstructured meshes.

An unstructured simplex mesh requires a choice of mesh points (vertex nodes) and triangulation or quadrangulation. Many mesh generators produce a mesh of triangles by first creating all the nodes and then connecting nodes to form triangles. The question arises as to what is the 'best' triangulation on a given set of points. One particular scheme, namely Delaunay triangulation [11], is considered by many researchers to be most suitable for finite element analysis. If the problem domain is a subset of the Cartesian plane, triangular or quadrilateral meshes are typically employed.

In this paper, we present a novel mesh generation scheme of all quadrilateral elements for linear polygonal domains. This scheme converts the elements in background quadrilateral into quadrilaterals using bilinear transformation We first decompose the linear polygon into simple subregions in the shape of quadrilaterals. These simple subregions are then quadrangulated by using the one to one mapping concept between the original quadrilateral and the square. We propose then an automatic quadrilateral conversion scheme in which each background quadrilateral with n by m divisions into nm quadrilaterals according to the bilinear mapping scheme. Further, to preserve the mesh conformity a similar procedure is also applied to every quadrilateral of the domain and this fully discretizes the given linear polygonal domain into all quadrilaterals, thus propogating uniform refinement and quadrangulation. In section-2 of this paper,we explain the particular transformations required in generating the degenerate forms of the quadrilateral: rectangles,parallelograms,trapeziums etc shapes, In section-3 of this paper,we present a scheme to discretize the arbitrary quadrilateral into a fine mesh of quadrilateral elements.In section- 4,we explain the procedure to create higher order quadrilateral by inserting midside nodes in each four node element. In section-5,we have presented a method of piecing together of all quadrilateral subregions and eventually creating an all quadrilateral mesh for the given linear polygonal domain. In section-6,we present several examples to illustrate the simplicity and efficiency of the proposed mesh generation method for triangles,rectangles , arbitrary quadrilaterals and convex and non convex polygonal domains.

2. Linear Convex Quadrilateral and Isoparametric Coordinate Transformation

Let us consider an arbitrary four noded linear convex quadrilateral element in the Cartesian space (x, y) which is mapped into a 2-square in the local parametric space (ξ, η).

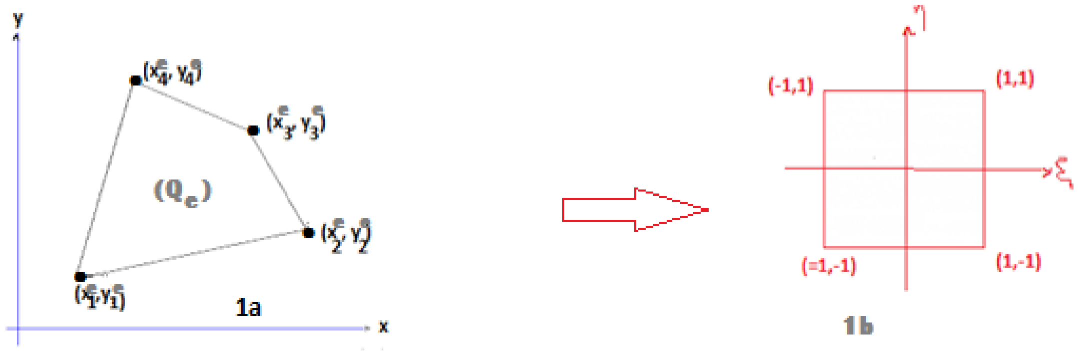


Fig.1a: Linear: convex quadrilateral Q^e in (x,y) space,

Fig.1b: Standard 2-square in in (ξ, η) space

The Isoparametric coordinate transformation from (x, y) space to the (ξ, η) space is given by $\begin{pmatrix} x^e \\ y^e \end{pmatrix} = \sum_{k=1}^{4} \frac{1}{4}(1 + \xi\xi_k)(1 + \eta\eta_k) \begin{pmatrix} x_k^e \\ y_k^e \end{pmatrix}$……………………………………..(1)

Where $((x_k^e, y_k^e), k = 1,2,3,4)$ are the vertices of the linear convex quadrilateral element 'e' in the Cartesian space (x, y) with $((\xi_k, \eta_k), k = 1,2,3,4) = \{(-1,-1).(1,-1),(1,1),(-1,1)\}$ are the vertices of the 2-square in (ξ, η) space.

From Eqn(1), we obtain

$$\begin{pmatrix} x^e \\ y^e \end{pmatrix} = \begin{pmatrix} a_0^e + a_1^e\xi + a_2^e\eta + a_3^e\xi\eta \\ b_0^e + b_1^e\xi + b_2^e\eta + b_3^e\xi\eta \end{pmatrix} \quad \text{------------------------------(2)}$$

where

$$\begin{pmatrix} a_0^e \\ b_0^e \end{pmatrix} = \begin{pmatrix} \frac{1}{4}(x_1^e + x_2^e + x_3^e + x_4^e) \\ \frac{1}{4}(y_1^e + y_2^e + y_3^e + y_4^e) \end{pmatrix} \quad \text{------------------------------(3a)}$$

$$\begin{pmatrix} a_1^e \\ b_1^e \end{pmatrix} = \begin{pmatrix} \frac{1}{4}(-x_1^e + x_2^e + x_3^e - x_4^e) \\ \frac{1}{4}(-y_1^e + y_2^e + y_3^e - y_4^e) \end{pmatrix} \quad \text{------------------------------(3b)}$$

$$\begin{pmatrix} a_2^e \\ b_2^e \end{pmatrix} = \begin{pmatrix} \frac{1}{4}(-x_1^e - x_2^e + x_3^e + x_4^e) \\ \frac{1}{4}(-y_1^e - y_2^e + y_3^e + y_4^e) \end{pmatrix} \quad \text{------------------------------(3c)}$$

$$\begin{pmatrix} a_3^e \\ b_3^e \end{pmatrix} = \begin{pmatrix} \frac{1}{4}(x_1^e - x_2^e + x_3^e - x_4^e) \\ \frac{1}{4}(y_1^e - y_2^e + y_3^e - y_4^e) \end{pmatrix} \quad \text{------------------------------(3d)}$$

The nature of the constants $((a_i^e, b_i^e), i = 0,1,2,3)$ will determine the element geometry. We have briefly listed some of these element geometries.

Rectangular elements

When $a_1^e = a^e,\ \ a_2^e = 0,\ \ a_3^e = 0; b_1^e = 0,\ \ b_2^e = b^e,\ \ b_3^e = 0$

and $(a_0^e,\ b_0^e)$ as coordinate of the element centroids, we can generate rectangular elements whose sides are parallel to coordinate axes with half side length = a^e and half side width = b^e. This gives

$$x^e = a_0^e + a^e\xi, \quad y^e = b_0^e + b^e\eta \quad \text{--------------------(4a)}$$

Parallelogram elements

(i)When $a_3^e = 0, \quad b_1^e = 0, \quad b_3^e = 0$, gives

$$x^e = a_0^e + a_1^e\boldsymbol{\xi} + a_2^e\eta, \quad y^e = b_0^e + b_2^e\eta \quad \text{--------------------(4b)}$$

and this will generate a parallelogram whose two sides are parallel to y-axis.

(ii)When $a_2^e = 0, \quad a_3^e = 0, \quad b_3^e = 0$, this gives

$$x^e = a_0^e + a_1^e\boldsymbol{\xi}, \quad y^e = b_0^e + b_1^e\xi + b_2^e\eta \quad \text{--------------------(4c)}$$

and this will generate parallelogram element whose two sides are parallel to x- axis

(iii)Arbitrary oriented rectangles and parallelograms are generated

When $x^e = a_0^e + a_1^e\boldsymbol{\xi} + a_2^e\eta y^e = b_0^e + b_1^e\xi + b_2^e\eta$ ----------------(4d)

with $a_3^e = 0, \quad b_3^e = 0$

(iv)When all the parameters $((a_i^e,\ b_i^e), i = 0,1,2,3)$ are non-zero, arbitrary quadrilaterals are generated and this is the general case and we have

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

$$y^e = b_0^e + b_1^e\xi + b_2^e\eta + b_3^e\xi\eta \quad \text{--------------------(4e)}$$

This case also covers the trapezium elements when either x^e is linear or y^e is linear and one of these say when either x^e or y^e is nonlinear.That is:

$$x^e = a_0^e + a_1^e\xi + a_2^e\eta$$

$$y^e = b_0^e + b_1^e\xi + b_2^e\eta + b_3^e\xi\eta \quad \text{--------------------(4f)}$$

and

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

$$y^e = b_0^e + b_1^e\xi + b_2^e\eta \qquad \text{---(4g)}$$

This analysis as explained above covers all bilinear mappings for various shapes degenerated from the arbitrary linear convex quadrilateral.

3. Mesh Generation over Linear Convex Quadrilaterals

We can map an arbitrary quadrilateral Q_e with vertices $((x_i^e, y_i^e), i = 1,2,3\ 4)$, in Cartesian space (x, y) into a unit square in the local space (u, v). The mapping is shown in figs.1a and 1c The unit square in uv-space is a convenient choice for division into smaller squares or rectangles.

Let us consider the bilinear mapping of an arbitrary quadrilateral Q_e with vertices $((x_i^e, y_i^e), i = 1,2,3\ 4)$, into a standard unit square.

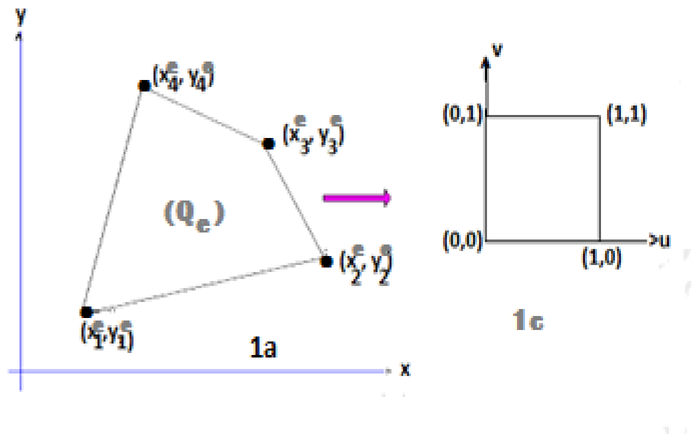


Fig.1a:convex quadrilateral in (x,y) space,Fig.1c:standard 1-square in (u,v) space,

The above mapping is defined as

$$x^e(u,v) = x_1^e + u(x_2^e - x_1^e) + v(x_4^e - x_1^e) + uv(x_1^e - x_2^e + x_3^e - x_4^e) \qquad \text{..............(5a)}$$

$$y^e(u,v) = y_1^e + u(y_2^e - y_1^e) + v(y_4^e - y_1^e) + uv(y_1^e - y_2^e + y_3^e - y_4^e) \qquad \text{...............(5b)}$$

We divide the unit square into $m \times n$ rectangles by making m divisions along u-axis and n-divisions along v-axis and this division (u,v) space has a one to one correspondence with a similar division of quadrilateral Q_e in (x,y) space.We now display the above mapping which shows divisions of a quadrilateral Q_e in (x, y) space and the corresponding divisions of a unit square in (u, v) space in Fig.2a and Fig.2b:

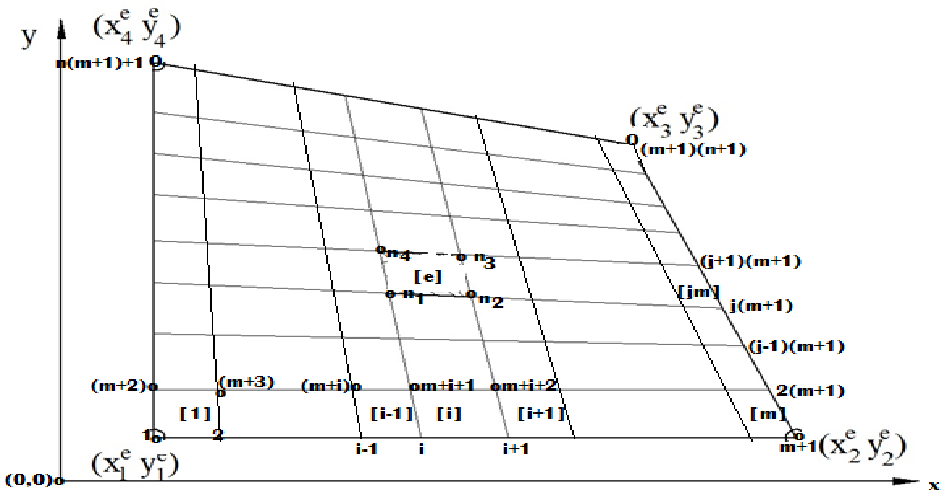


Fig.2a: mxn divisions of an arbitrary quadrilateral Q_e in (x, y) space and a typical quadrilateral element 'e' in the interior

[e]: quadrilateral with nodal vertices<n_1 ,n_2 ,n_3 ,n_4>,

n_1= (j-1)(m+1)+i, n_2= (j-1)(m+1)+i+1, n_3= j(m+1)+i+1 , n_4= j(m+1)+i.

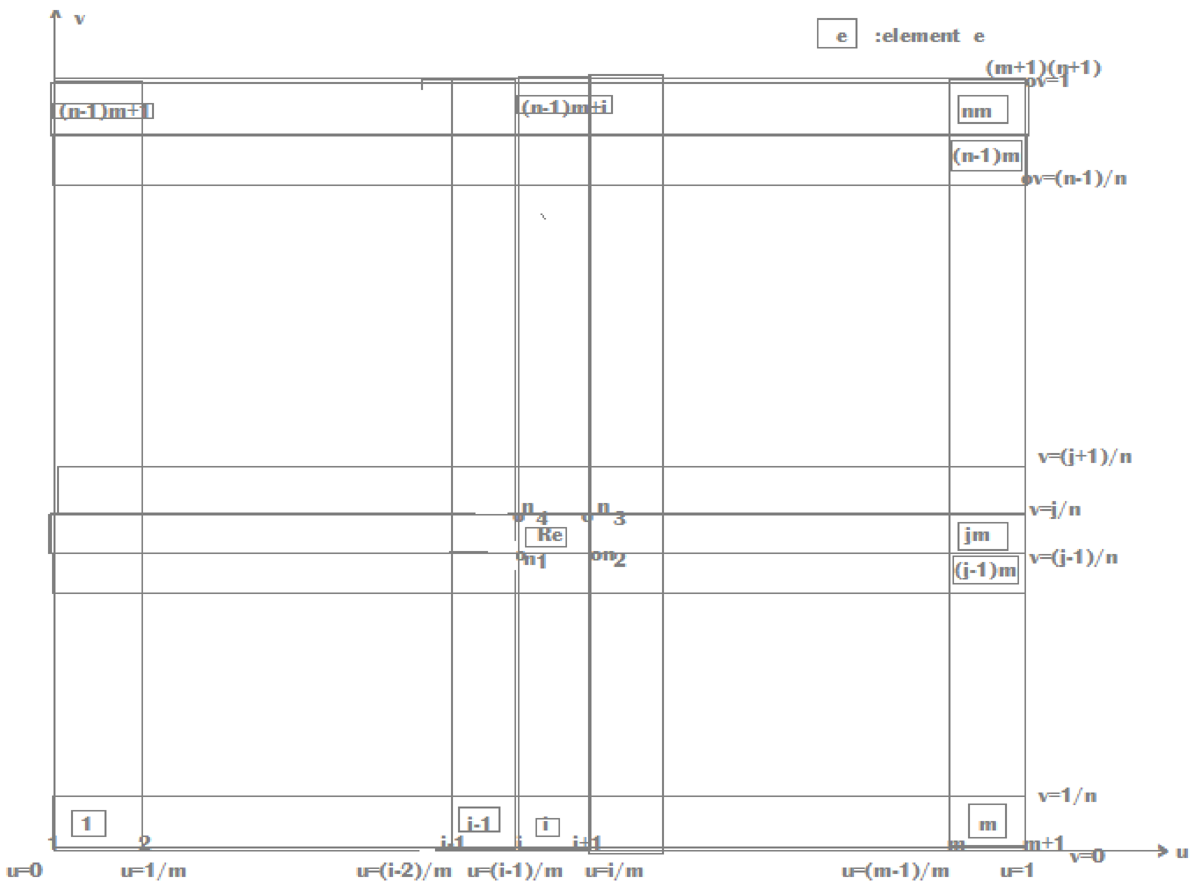


Fig.2b: Division of a unit square into 'mn' four node rectangles, in (u,v) space

[e]:: element'e'

[Re]: rectangle with nodal vertices<n_1 ,n_2 ,n_3 ,n_4>,

(n_i ,i=1,2,3,4)::node numbers of element vertices

((u_k,v_l) ,k=1,2,3,….m+1),$l = 1,2,3,\ldots n+1$)):local coordinates of unit square

n_1= (j-1)(m+1)+i, n_2= (j-1)(m+1)+i+1, n_3= j(m+1)+i+1 , n_4= j(m+1)+i.

The vertices of corner nodes for the element (e) has coordinates (in anticlockwise sense) are

$n_1(x^e(u_i,v_i),y^e(u_i,v_i))$, $\quad n_2\left(x^e(u_{i+1},v_j),y^e(u_{i+1},v_j)\right)$

$n_3(x^e(u_{i+1},v_{i+1}),y^e(u_{i+1},v_{i+1}))$; $\quad n_4\left(x^e(u_i,v_{j+1}),y^e(u_i,v_{j+1})\right)$ ------------(6)

Where

$n_1(u_i,v_j) = n_1((i-1)/m, \quad (j-1)/n)$,

$n_2(u_{i+1},v_j) = n_2(i/m, \quad (j-1)/n)$,

$n_3(u_{i+1},v_{j+1}) = n_3(i/m, \quad j/n)$,;

$n_4(u_i,v_{j+1}) = n_4((i-1)/m, \quad j/n)$, --(7)

and the node numbers in both the spaces are

$n_1 = j(m+1) + i$; $n_2 = j(m+1) + i + 1$;

$n_3 = (j+1)(m+1) + i + 1$; $n_4 = (j+1)(m+1) + i$ -----------------------------(8)

All the element nodes and coordinates can be obtained by varying i and j, $i = 1,2,\dots,(m+1)$; $j = 1,2,\dots,(n+1)$ and naturally over any typical element $(i+1) \le (m+1)$ and $(j+1) \le (n+1)$.

We have shown the division of an arbitrary quadrilateral Q_e and a unit square in Fig. 2a and Fig. 2b. respectively. We divide each side of the quadrilateral and unit square (in Cartesian space(x,y) and natural space(u,v)) into m equal division along x and u axes and n equal divisions along y and v axes. This creates (m+1)* $(n+1)$ nodes. These nodes are numbered from base line l_{12} (letting l_{ij} as the line joining the vertex $(x_i^e, y_i^e)$and(x_j^e, y_j^e)) *and move upwards upto the line* l_{34} *in quadrilateral* Q_e ; now with respect to the unit squqre in Fig.2b,we move along the line $v = 0$ and upwards up to the line $v = 1$. The nodes along v=0 are 1, 2,…(m+1);and then on $v_1 = 1/n$ are (m+2),(m+3),….2(m+1);etc and finally on v=1 are n(m+1)+1,n(m+1)+2,….,(n+1)*(m+1) and they are numbered layer by layer.This is shown in the following matrix of node numbers $\underline{rr}$:

$\underline{rr}$ =

u=0	u=1/m	u=2/m	u=(i-1)/m	u=i/m	u=(m-1)/m	u=1	
1	2	3	i	i+1	m	(m+1)	⇒ v=0
(m+2)	(m+3)	(m+4)	(m+i+1)	(m+i+2)	(2m+1)	2(m+1)	⇒ v=1/n
2(m+1)+1					(3m+2)	3(m+1)	⇒v=2/n
(j-1)(m+1)+1			(j-1)(m+1)+i+1	(j-1)(m+1)+i+2		j(m+1)	⇒v=(j-1)/n
j(m+1)+1			j(m+1)+i+1	j(m+1)+i+2		(j+1)(m+1)	⇒v=j/m
(n-1)(m+1)+1			(n-1)(m+1)+i+1	(n-1)(m+1)+i+2		n(m+1)	⇒v=(n-1)/n
n(m+1)+1			n(m+1)+i+1	n(m+1)+i+2		(n+1)(m+1)	⇒v=1

Fig 2c. Matrix rr of node numbers for the division of a unit square

…………………………………………(9)

4. Mesh Generation Using Higher order Quadrilaterals

In finite element applications, we may have to generate higher order quadrilateral elements. They contain midside nodes. We can obtain quadratic elements by inserting additional nodes at the midpoints of the linear four node element boundaries which gives us Serendipity quadratic elements. In addition to this when a node is also inserted at the centroid of the quadrilateral, we obtain Lagrange quadratic elements.

We next consider cubic elements, they can be obtained by inserting nodes at the trisectional points of the four node element boundaries which gives us Serendipity cubic elements. In addition to this, if we insert nodes in the interior of the elements at trisectional points, we obtain the Lagrange cubic elements.

Zienkiewicz[17] intended to define a Serendipity family so that polynomial completeness is realized with necessary minimum nodes and presented the few lower order elements viz.Linear,Quadratic and Cubic elements which have equal number of nodes along each side which are uniformly spaced.It is obvious that the Basis functions for Seredipity elements with nodes placed only along the edges cannot generate complete polynomials beyond cubic,for this reason,Zienkiewicz[17] has suggested a central node for the next Quartic member of this family, and remarks that progression to yet higher order members is difficult and requires some ingenuity. M.Okabe[29], H.T.Rathod and Sridevi. Kilari [30] determined the Basis functions of the Serendipity and Complete Lagrange family elements which allow uniform spacing of nodes over the element domain for orders 4-10.We intend to generate finite element meshes over polygonal domains at least upto Quartic order for Serendipity, Complete Lagrange and Lagrange family elements.

These rectangular elements in the local parametric space are depicted in the following figures.

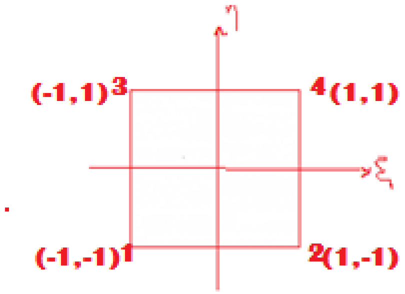


Fig.3a: Linear four node element

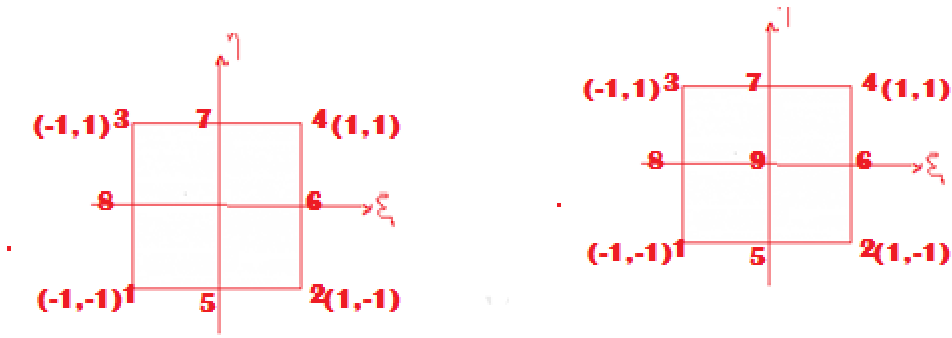


Fig.3b: Quadratic -Serendipity and Lagrange elements

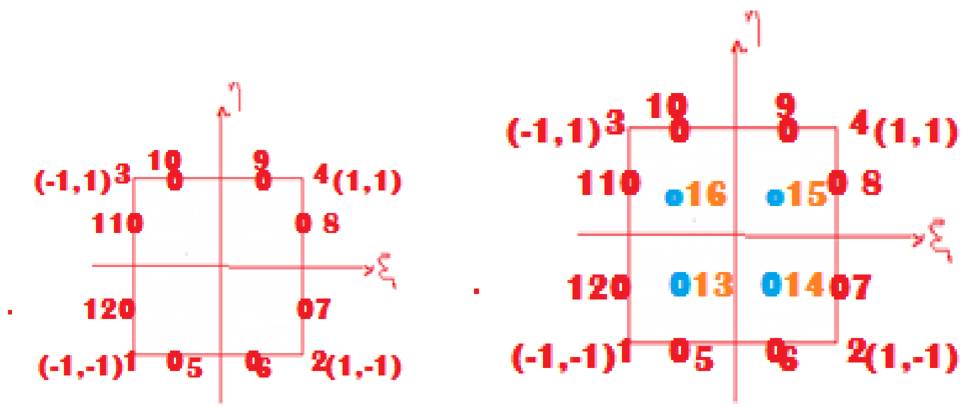


Fig.3c: Cubic -Serendipity and Lagrange elements

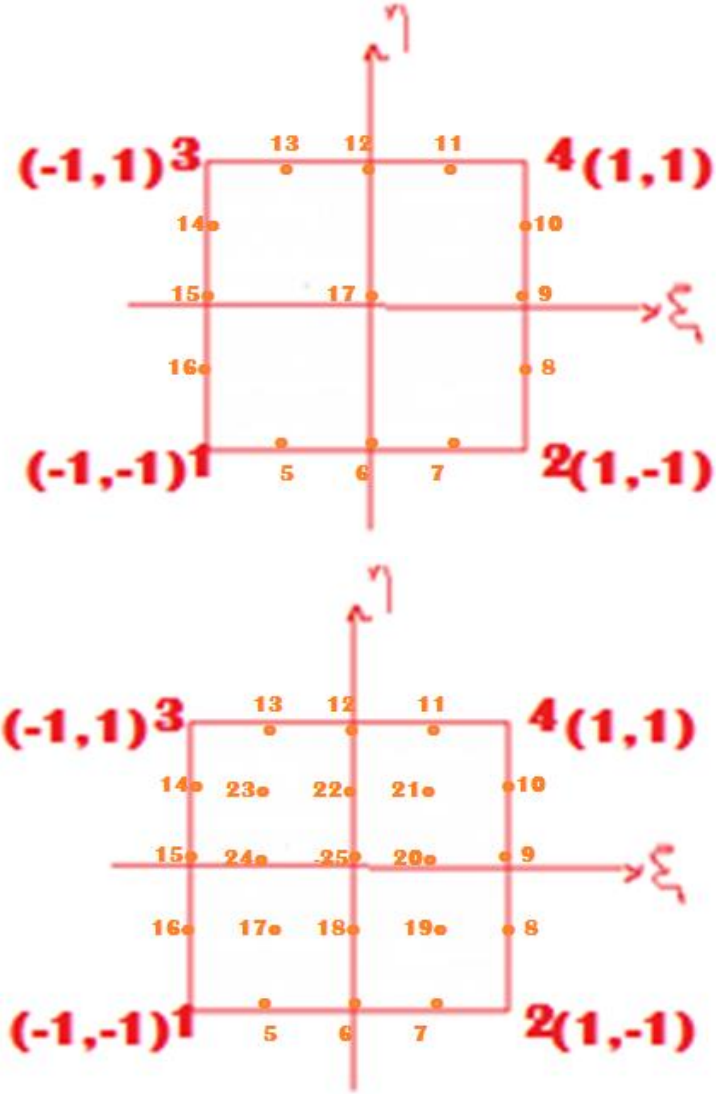


Fig.3d: Quartic- Complete Lagrange and Lagrange elements

5. Quad angulation of an Arbitrary Polygon

In finite element applications to physical problem require mesh generation over polygonal domains. We divide this domain into a coarse mesh of triangles or quadrilaterals or both. Our aim now is to generate a mesh of all quadrilaterals. This is first done by generating quadrilateral meshes over each coarse shape (triangles or quadrilateral) and then piecing together, we obtain an all quadrilateral mesh for the polygonal domains. These quadrilaterals must be conformal and one of the types 4, 8, 9 12, 16,17,25 noded elements.

In recent papers [24-28], a new mesh generation method for a convex polygonal domain was presented.This method decomposes the convex polygon into simple subregions in the shape of triangles. These simple regions are then triangulated to generate a fine mesh of triangular elements. We propose then an automatic triangular to quadrilateral conversion scheme. Each isolated triangle is split into three quadrilaterals according to the usual scheme, adding three vertices in the middle of the edges and a vertex at the barrycentre of the element. To preserve the mesh conformity a similar procedure is also applied to every triangle of the domain to fully discretize the given convex polygonal domain into all quadrilaterals, thus propagating uniform refinement. This simple method generates a high quality mesh whose elements confirm well to the requested shape by refining the problem domain.**In this paper we have proposed the decomposition of convex and nonconvex polygonal domains into simple subregions in the shape of arbitrary quadrilaterals**. These simple regions are then quadrangulated as explained in the previous section.This is further explained in the following Figs.4-5. We consider a convex polygon which is divided into four arbitrary quadrilaterals,we generate quadrangular mesh over each of them.The exploded view of the polygonal domain is shown in Figs.4-5.It is clear that by pieceing together all the four arbitrary quadrilateral which are already quadrangulated by the method of previous section,we can obtain the desired mesh

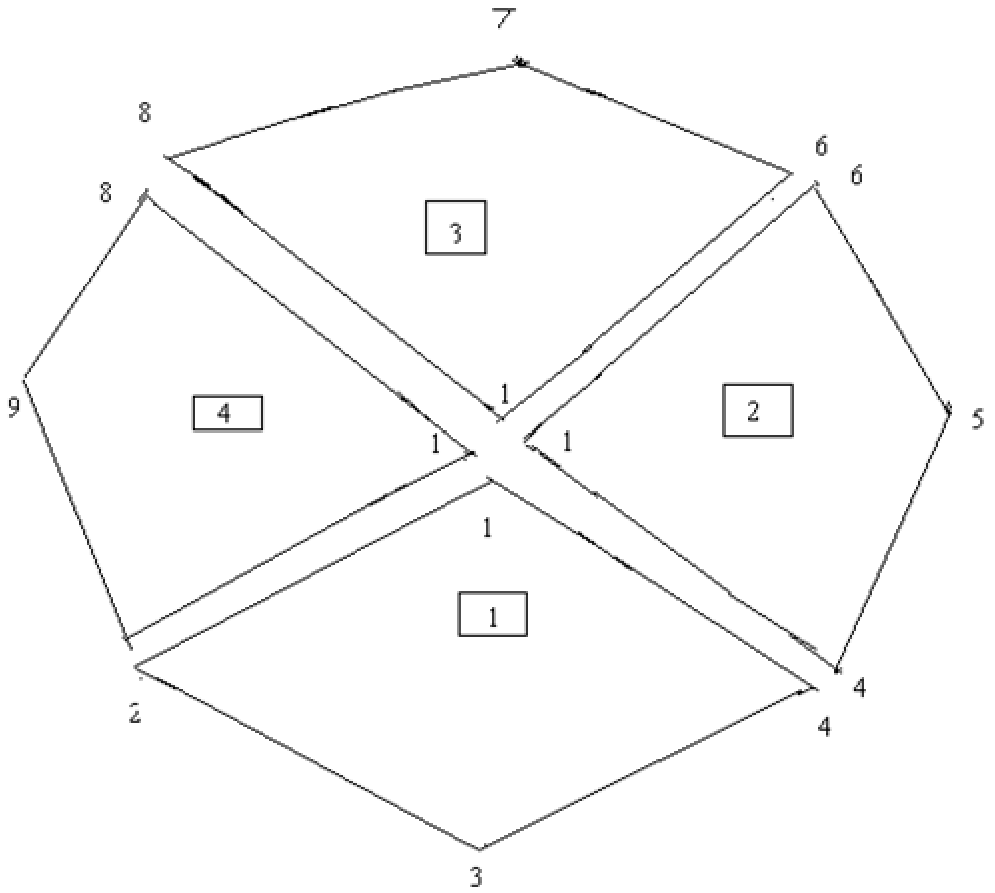


Fig.4

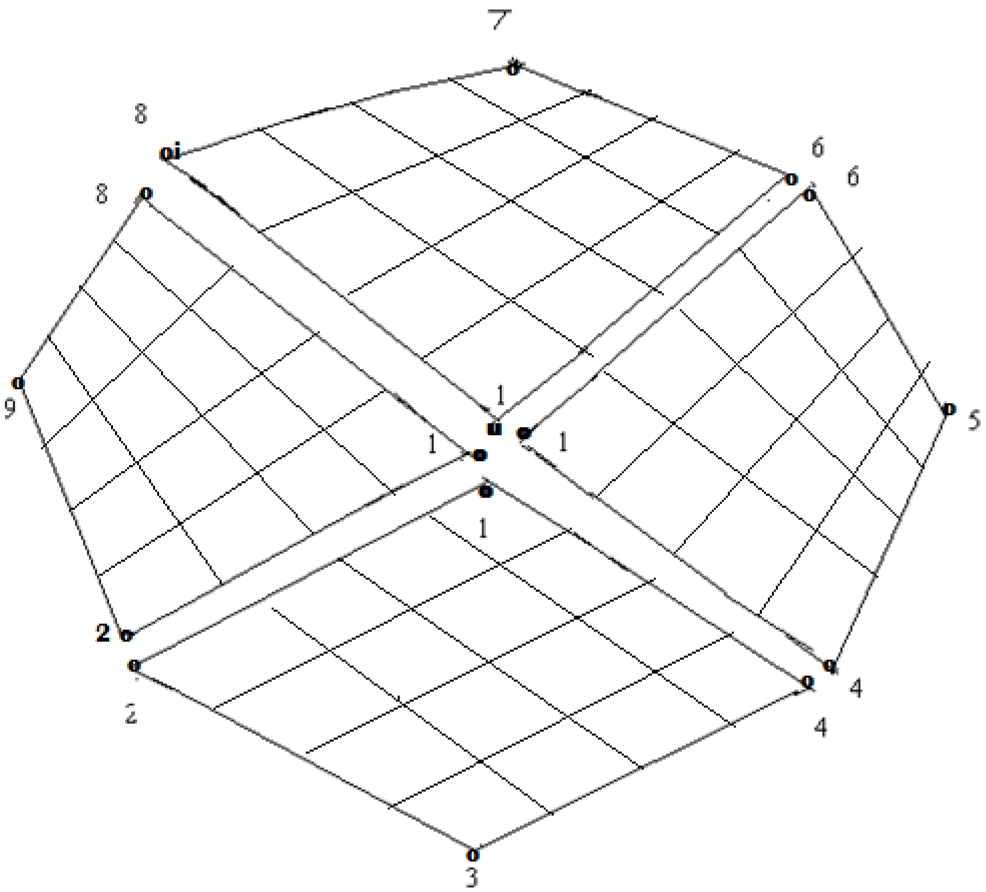


Fig.5

Figs.4-5 Piecing together of four quadrilaterals

6. Application Examples

In applications to boundary value problems, we may have to discretize an arbitrary polygonal domain using linear,quadratic,cubic and quartic finite elements. Our purpose is to have codes which automatically generates elements with linear convex quadrilaterals over the domain by assuming the input as coordinates of the vertices. We have choosen four typical examples:

(i)An Arbitrary Quadrilateral

(ii)An Equilateral Triangle

(iii)A Convex Polygon

(iv)A Nonconvex Polygon

We may note that the rectangles and parallelograms of any orientation will be discritsed into finite element meshes of rectangles(or squares) and parallelograms.Two codes written in MATLAB programming and based on the proposed scheme of this paper to generate meshes using Quartic Complete Lagrange elements with 17-nodes are appended.Codes for Linear,Quadratic and Cubic finite elements were developed on similar lines and the schemes explained in this paper but they are not included here. Several Figures on Finite Element mesh generation using 1-4th order elements(i.e,4,8,9,12,16,17,25-noded each) are presented immediately after **References.**

7. Conclusions

This paper presents a novel mesh generation scheme of all quadrilateral elements over a linear polygonal domain. We first decompose the linear polygon into simple sub regions in the shape of quadrilaterals. These simple regions are then quadrangulated to generate first into a fine mesh of four node quadrilateral elements using bilinear transformations.We propose then an automatic quadrilateral conversion scheme. Each four node quadrilateral is converted to an 8-node,9-node,12-node ,16-node,17-node and 25-noded quadrilaterals by inserting the midside nodes appropriately. Examples are presented to illustrate the simplicity and efficiency of the new mesh generation method for standard and arbitrary shaped domains. We have appended two important MATLAB programs which incorporate the mesh generation scheme for the 17-noded complete Lagrange elements developed in this paper.Other MATLAB programs can be coded on similar lines. These programs provide valuable output on the nodal coordinates ,element connectivity and graphic display of the all quadrilateral meshes for application in finite element analysis.

In finite element applications, we may have to generate higher order quadrilateral elements. They contain midside nodes. We can obtain quadratic elements by inserting additional nodes at the midpoints of the linear four node element boundaries which gives us Serendipity quadratic elements. In addition to this when a node is also inserted at the centroid of the quadrilateral, we obtain nine node Lagrange quadratic elements.

We next consider cubic elements, they can be obtained by inserting nodes at the trisectional points of the four node element boundaries which gives us Serendipity cubic elements. In addition to this, if we insert nodes in the interior of the elements at trisectional points, we obtain the Lagrange cubic elements. Zienkiewicz[17] intended to define a Serendipity family so that polynomial completeness is realized with necessary minimum nodes and presented the few lower order elements viz.Linear,Quadratic and Cubic elements which have equal number of nodes along each side which are uniformly spaced.It is obvious that the Basis functions for Seredipity elements with nodes placed only along the edges cannot generate complete polynomials beyond cubic,for this reason,Zienkiewicz[17] has suggested a central node for the next Quartic member of this family, and remarks that progression to yet higher order members is difficult and requires some ingenuity

This paper presents a novel mesh generation scheme of all quadrilateral elements over a linear polygonal domain. We first decompose the linear polygon into simple sub regions in the shape of quadrilaterals. These simple regions are then quadrangulated to generate first into a fine mesh of four node quadrilateral elements using **bilinear transformations**.We propose then an automatic quadrilateral conversion scheme for higher order elements. Each four node quadrilateral is converted to an 8-node,9-node,12-node ,16-node,17-node and 25-noded quadrilaterals by inserting the midside nodes appropriately. Examples are presented to illustrate the simplicity and efficiency of the new mesh generation scheme of all quadrilaterals for triangles,quadrilaterals and arbitrary shaped polygonal domains. We have appended two important MATLAB programs which incorporate the mesh generation scheme for the 17-noded complete Lagrange elements developed in [29,30].Other MATLAB programs can be coded on similar lines. These programs provide valuable output on the nodal coordinates ,element connectivity and graphic display of the all quadrilateral meshes for application to finite element analysis.

References

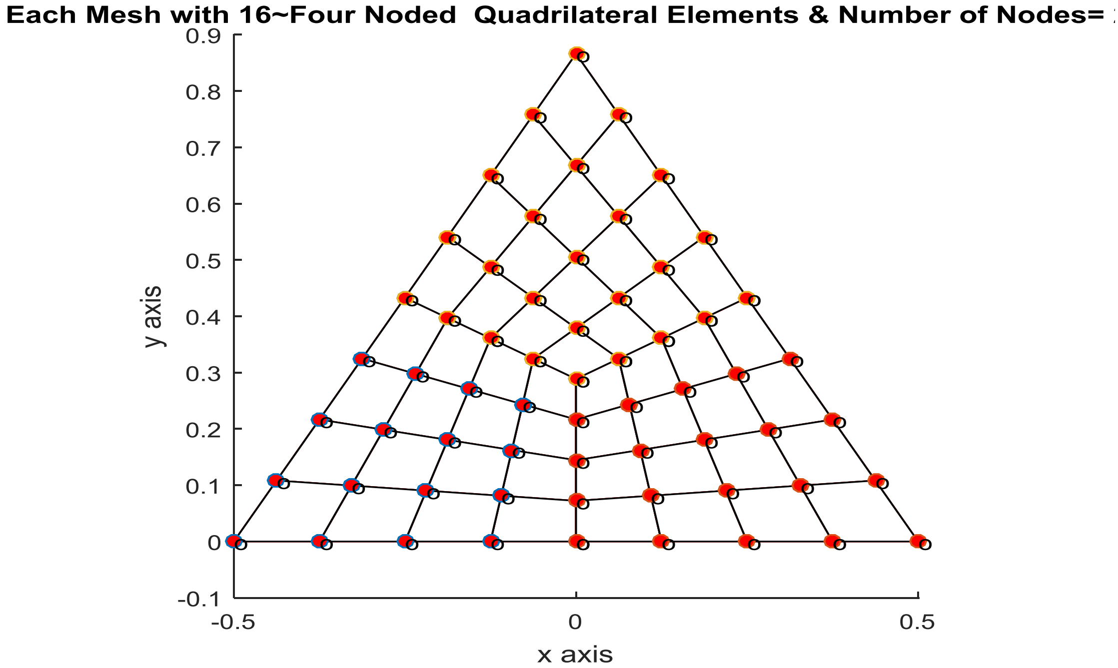
- R. Courant, Variational methods for the solution of problems of equilibrium and vibration, Bulletin of the American Mathematical Society, 49:1-23(1943).
- J.H. Argyris, Matrix displacement analysis of anisotropic shells by triangular elements, Journal of Royal Aeronautical Society 69:801 (1965).
- R.W. Clough, The finite element method in plane stress analysis, Proceedings of the 2nd ASCE Conference in Electronic Computation, Pittsburgh, PA 1960.
- Zienkiewicz. O. C, Philips. D. V, An automatic mesh generation scheme for plane and curved surface by isoparametric coordinates, Int. J.Numer. Meth.Eng, 3, 519-528 (1971) .
- Gordan. W. J and Hall. C. A, Construction of curvilinear coordinates systems and application to mesh generation, Int. J.Numer. Meth. Eng 3, 461-477 (1973).
- Cavendish. J. C, Automatic triangulation of arbitrary planar domains for the finite element method, Int. J. Numer. Meth. Eng 8, 679-696 (1974).
- Moscardini. A. O , Lewis. B. A and Gross. M A G T H M – automatic generation of triangular and higher order meshes, Int. J. Numer. Meth. Eng, Vol 19, 1331-1353(1983).
- Lewis. R. W, Zheng. Y, and Usmam. A. S, Aspects of adaptive mesh generation based on domain decomposition and Delaunay triangulation, Finite Elements in Analysis and Design 20, 47-70 (1995) .

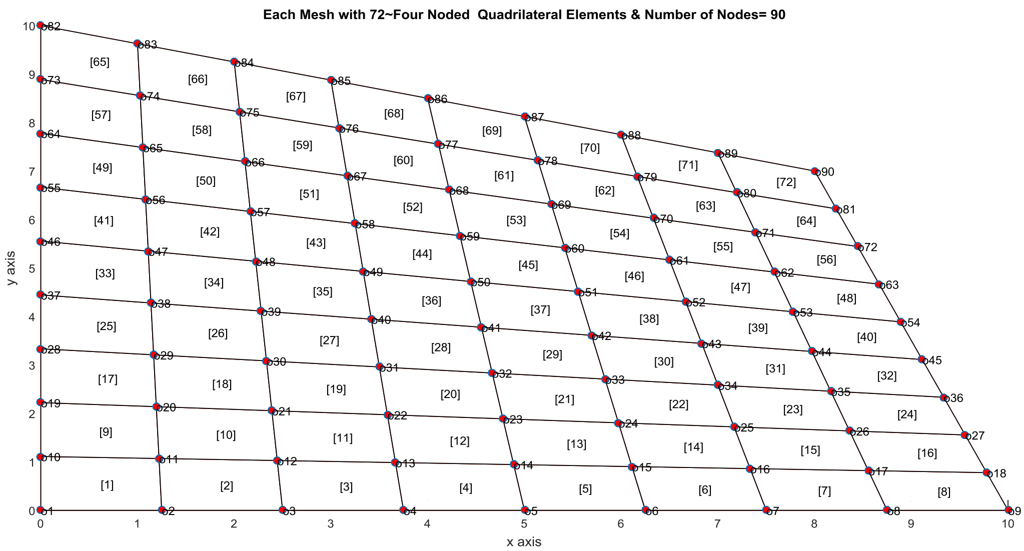
- W. R. Buell and B. A. Bush, Mesh generation a survey, J. Eng. Industry. ASME Ser B. 95 332-338(1973).
- Rank. E, Schweingruber and Sommer. M, Adaptive mesh generation and transformation of triangular to quadrilateral meshes, Common. Appl. Numer. Methods 9, 11 121-129(1993)
- Ho-Le. K, Finite element mesh generation methods, a review and classification, Computer Aided Design Vol.20, 21-38(1988)
- Lo. S. H, A new mesh generation scheme for arbitrary planar domains, Int. J. Numer. Meth. Eng. 21, 1403-1426(1985)
- Peraire. J. Vahdati. M, Morgan. K and Zienkiewicz. O. C, Adaptive remeshing for compressible flow computations, J. Comp. Phys. 72, 449-466(1987).
- George. P.L, Automatic mesh generation, Application to finite elements, New York , Wiley (1991).
- George. P. L, Seveno. E, The advancing-front mesh generation method revisited. Int. J. Numer. Meth. Eng 37, 3605-3619(1994)
- Pepper. D. W, Heinrich. J. C, The finite element method, Basic concepts and applications, London, Taylor and Francis (1992)
- Zienkiewicz. O. C, Taylor. R. L and Zhu. J. Z, The finite element method, its basis and fundamentals, 6th Edn, Elsevier (2007)
- Masud. A, Khurram. R. A, A multiscale/stabilized finite element method for the advection- diffusion equation, Comput. Methods. Appl. Mech. Eng 193, 1997-2018(2004)
- Johnston. B.P, Sullivan. J. M and Kwasnik. A, Automatic conversion of triangular finite element meshes to quadrilateral elements, Int. J. Numer. Meth. Eng. 31, 67-84(1991)
- Lo. S. H, Generating quadrilateral elements on plane and over curved surfaces, Comput. Stuct. 31(3) 421-426(1989)
- Zhu. J, Zienkiewicz. O. C, Hinton. E, Wu. J, A new approach to the development of automatic quadrilateral mesh generation, Int. J. Numer. Meth. Eng 32(4), 849-866(1991)
- Lau. T. S, Lo. S. H and Lee. C. S, Generation of quadrilateral mesh over analytical curved surfaces, Finite Elements in Analysis and Design, 27, 251-272(1997)
- Park. C, Noh. J. S, Jang. I. S and Kang. J. M, A new automated scheme of quadrilateral generation for randomly distributed line constraints, Computer Aided Design 39, 258- 267(2007).
- Thompson.E.G, Introduction to the finite element method,John Wiley & Sons Inc.(2005).
- H.T.Rathod , Bharath Rathod , K.T.Shivaram , K. Sugantha Devi , Tara Rathod, An explicit finite element integration scheme for linear eight node convex quadrilaterals using automatic mesh generation technique over plane regions, *International Journal of Engineering and Computer Science, vol.3,issue5(2014),pp5657-5713.*
- H.T. Rathod, K. Sugantha devi, An All Quadrilateral Automesh Generation Technique and Explicit Integration Scheme for Finite Elements to Solve Some Elliptic Boundary Value Problems in Two Space, International Journal of Engineering and Computer Science, ISSN 2319-7242, Vol.5 issue 12(December 2016), pp 19674-19778.
- H.T.Rathod, Bharath Rathod, Sugantha Devi.K,Hariprasad.A.S, Finite element solution of Poisson Equation over Polygonal Domains using a novel auto mesh generation technique and an explicit integration scheme for eight node linear convex quadrilaterals, International Journal of Engineering and Computer Science, ISSN:2319-7242, Volume (6) Issue 10 October 2017, pp 22632-22787.
- H. T. Rathod Md.Shafiqul Islam. H. Y. Shrivall , Bharath Rathod, K. Sugantha Devi, Finite element solution of Poisson Equation over Polygonal Domains using a novel auto mesh generation technique and an explicit integration scheme for nine node linear convex quadrilateral of Lagrange family.*International Journal Of Engineering And Computer Science ISSN:2319-7242Volume 6 Issue 11 November 2017, Page No. 22869-23058.*
- H. T. Rathod, Md.Shafiqul.Islam, Bharath Rathod , K. Sugantha Devi , Finite element solution of Poisson Equation over Polygonal Domains using a novel auto mesh generation technique and an explicit integration scheme for linear convex quadrilaterals of cubic order Serendipity and Lagrange families, *International Journal Of Engineering And Computer Science ISSN:2319-7242,Volume 7 Issue 1 January 2018, Page No. 23329-23482.*

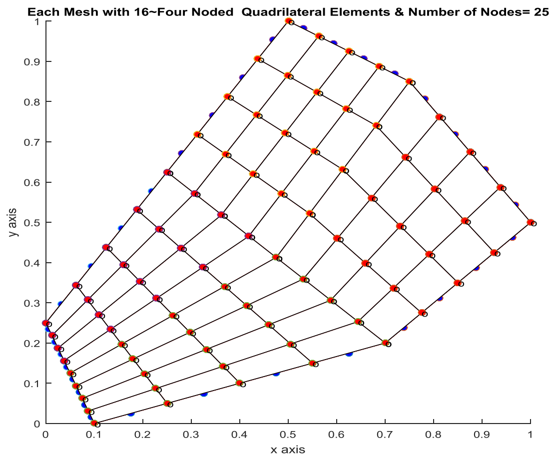
- M.Okabe,Complete Lagrange family for the cube in Finite Element interpolations,Comput.Meth.Appl.Mech.Engng.29(1981)51-56.
- H.T,Rathod and Sridevi.Kilari,General complete Lagrange family for the cube in finite element interpolations,Comput.Meth.Appl.Mech and Engrg 181(2000)295-344.

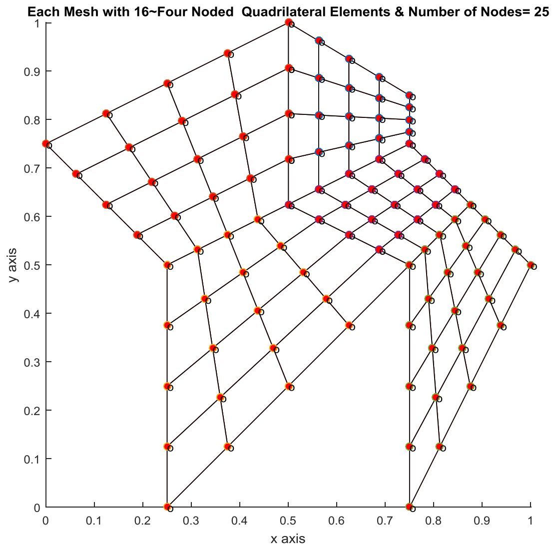
FIGURES

(1)FOUR NODE QUADRILATERALS

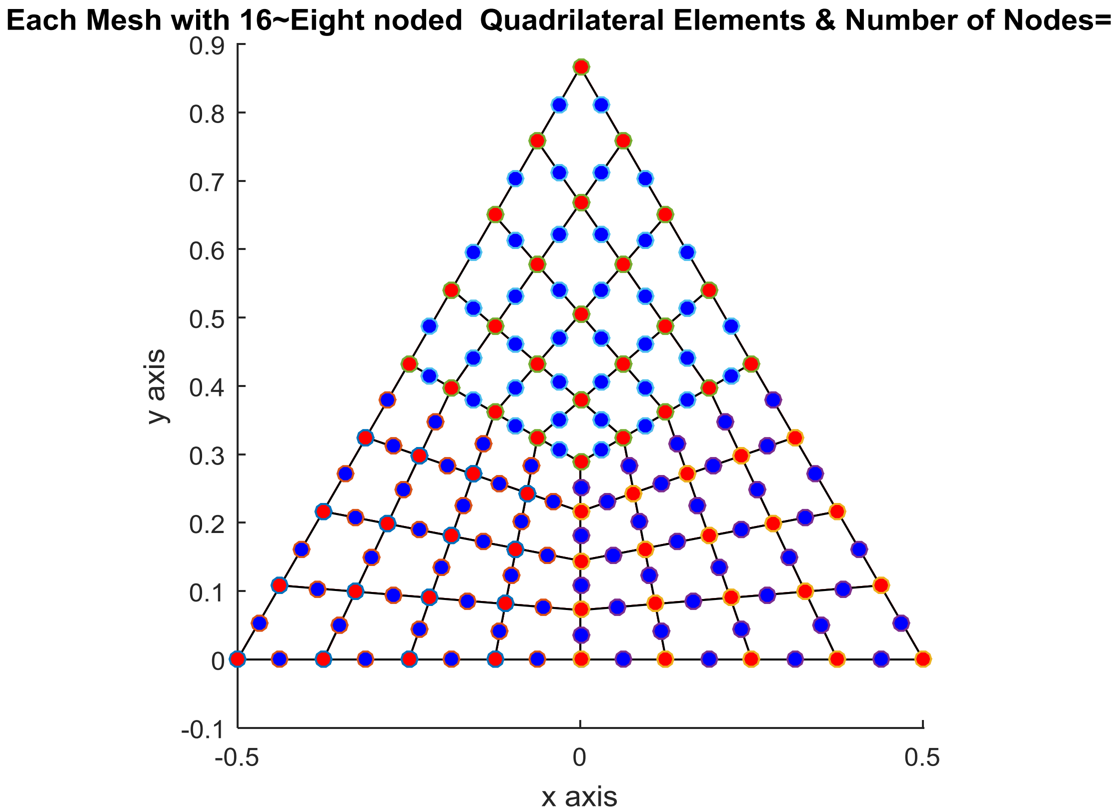


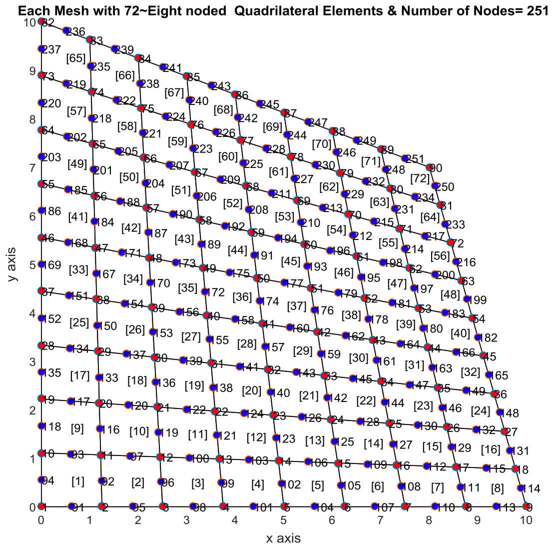


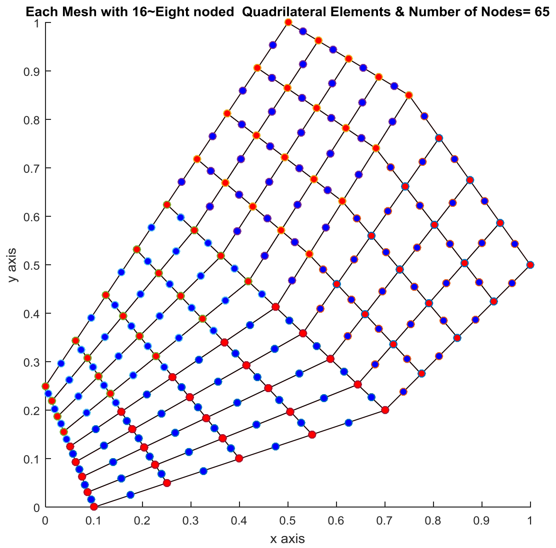


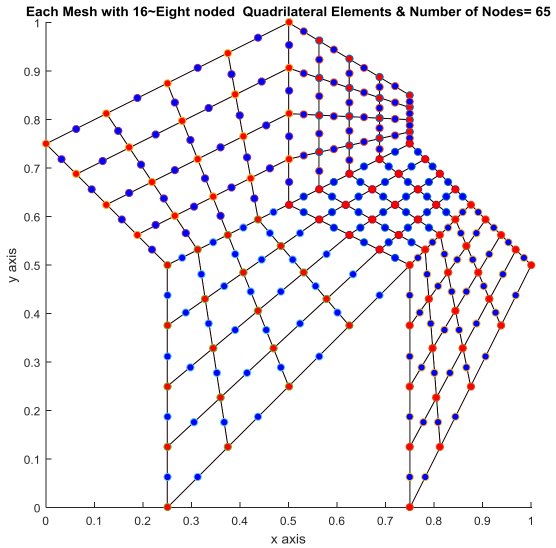


(2) 8- NODE SERENDIPITY QUADRILATERALS

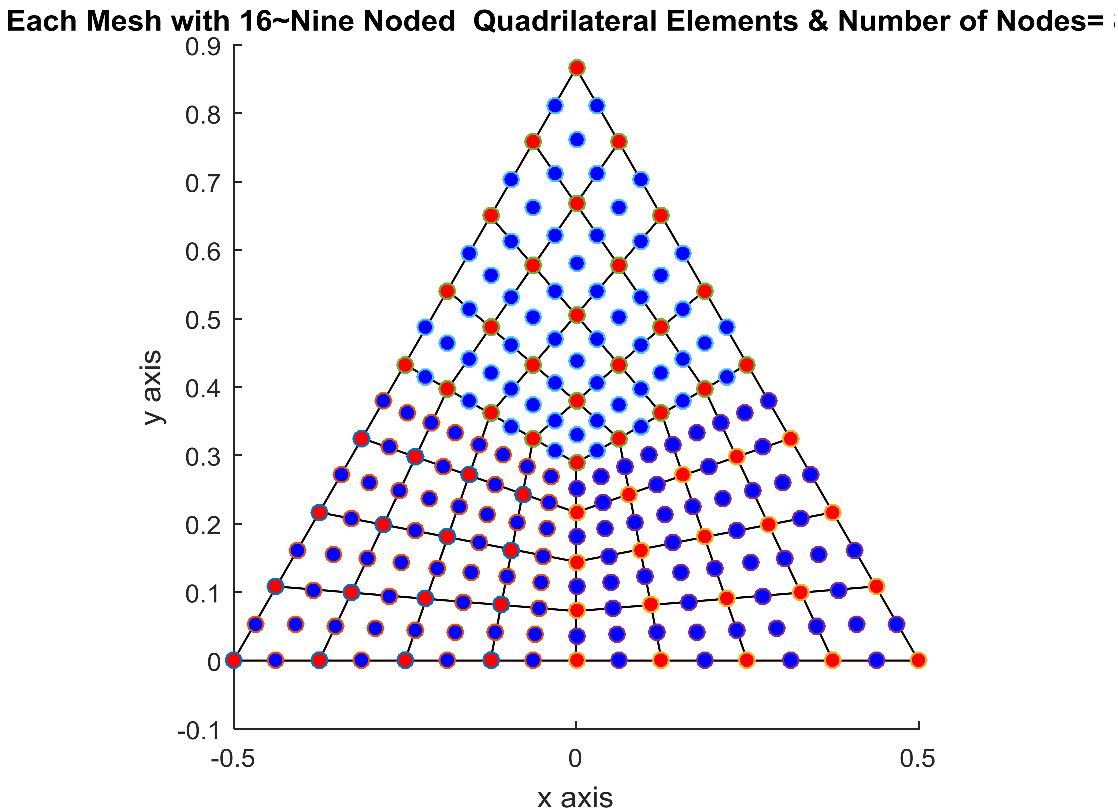


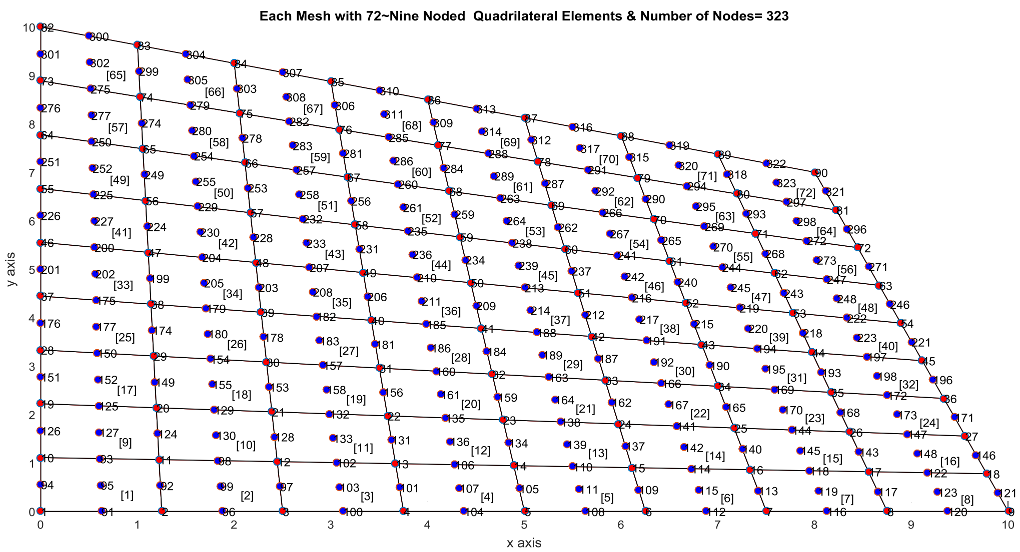

Each Mesh with 72~Eight noded Quadrilateral Elements & Number of Nodes= 251
x axis
y axis

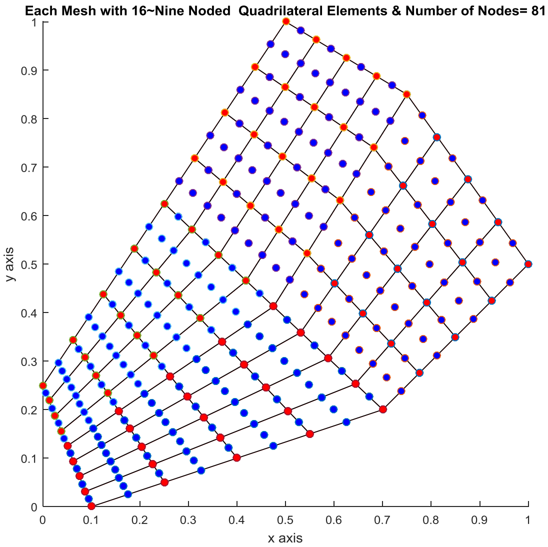

Each Mesh with 16~Eight noded Quadrilateral Elements & Number of Nodes= 65
x axis
y axis

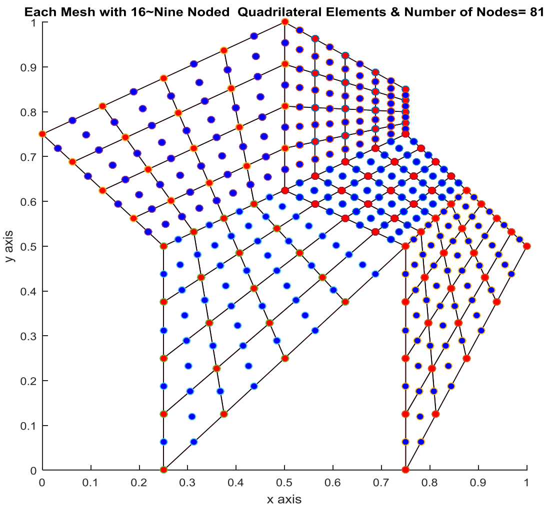

Each Mesh with 16~Eight noded Quadrilateral Elements & Number of Nodes= 65
x axis
y axis

(3) 9- NODE LAGRANGE QUADRILATERALS

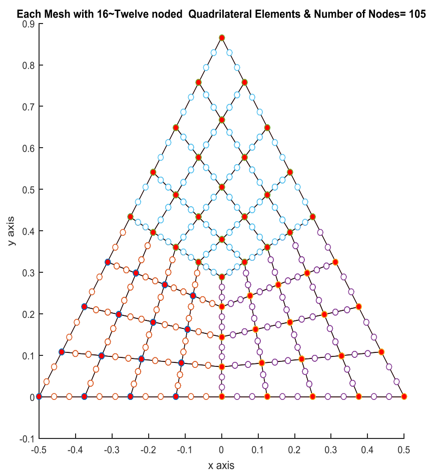


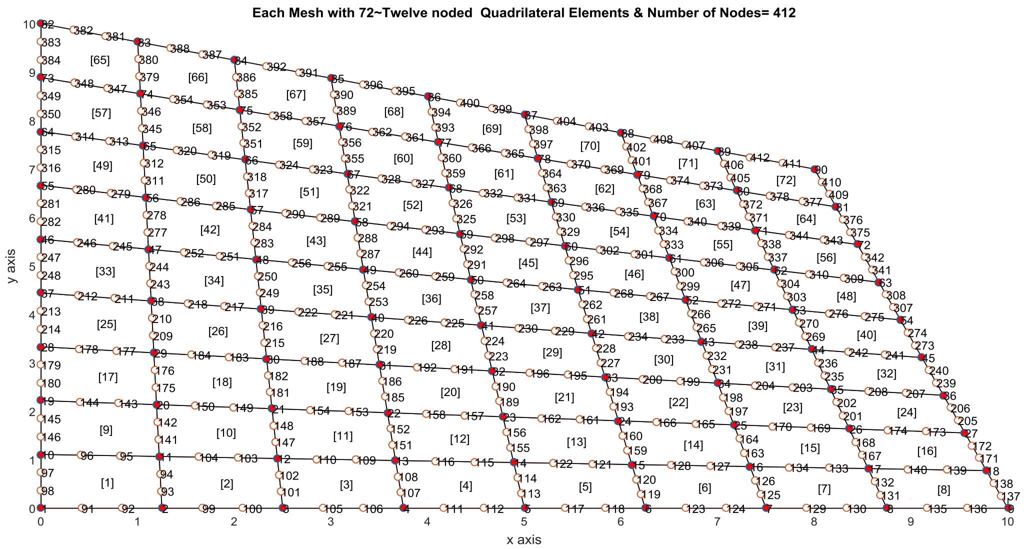




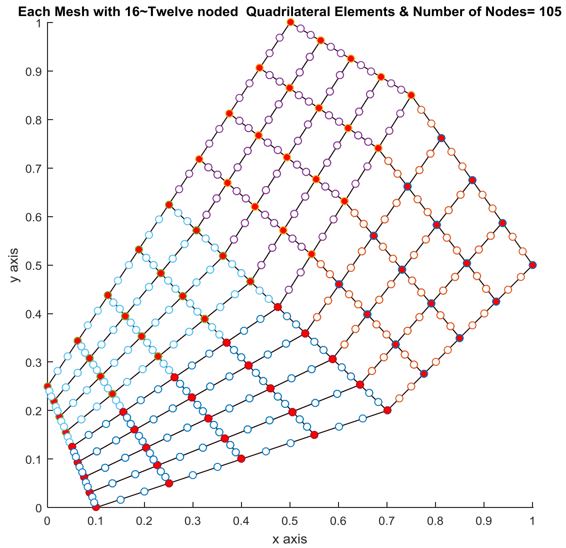


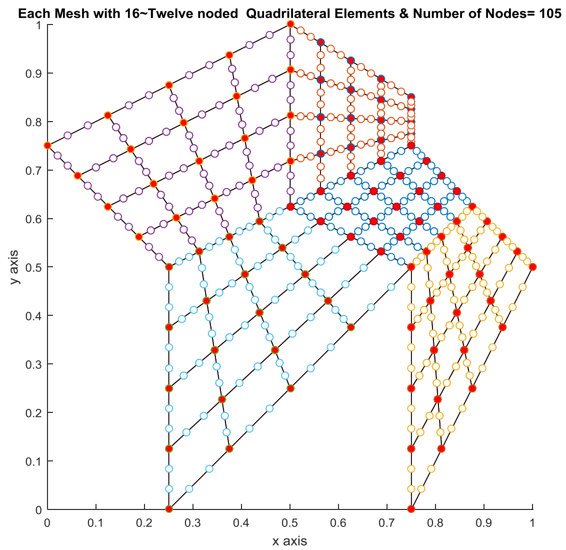
(4) 12-NODE SERENDIPITY ELEMENTS



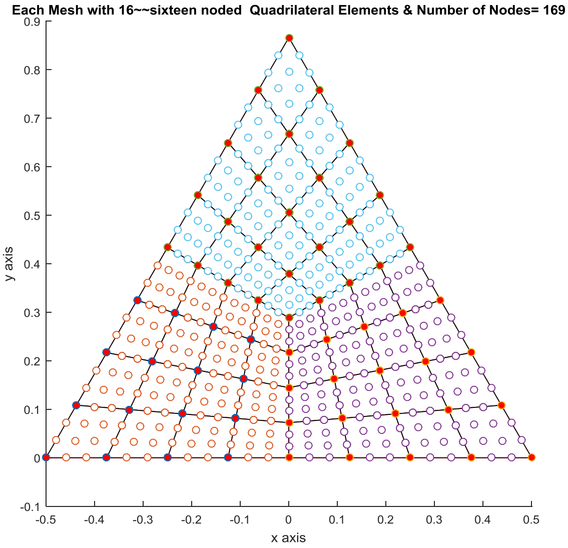

Each Mesh with 72~Twelve noded Quadrilateral Elements & Number of Nodes= 412
y axis
x axis

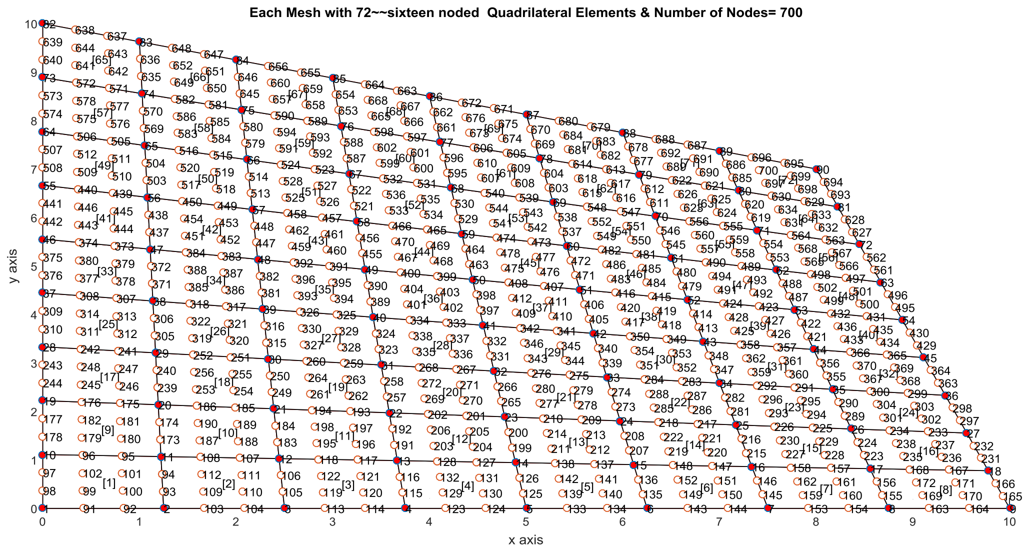
f

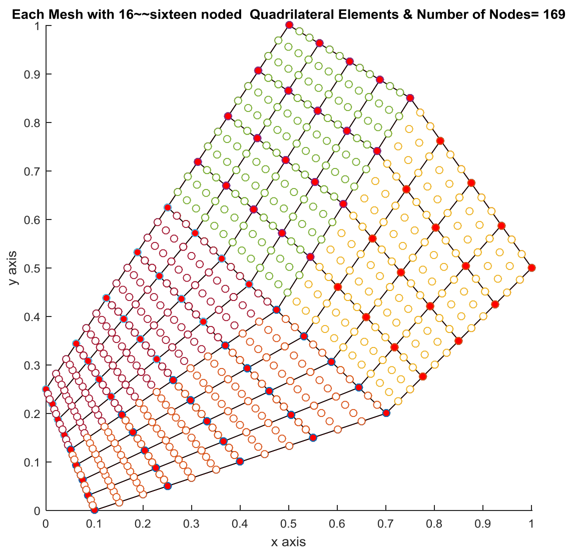

Each Mesh with 16~Twelve noded Quadrilateral Elements & Number of Nodes= 105
y axis
x axis

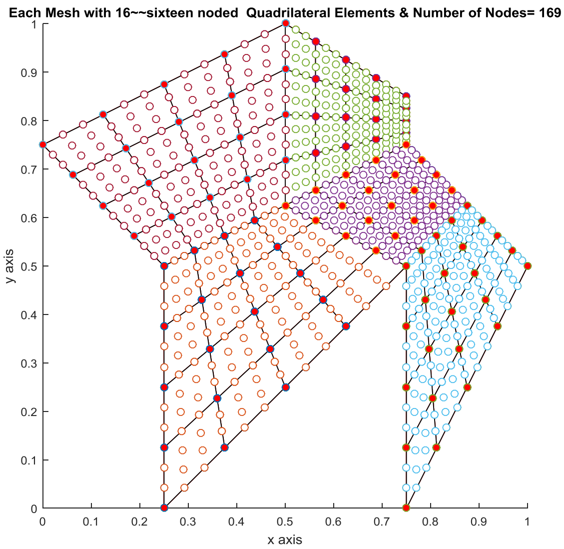

Each Mesh with 16~Twelve noded Quadrilateral Elements & Number of Nodes= 105
y axis
x axis

(5) 16-node Lagrange quadrilateral

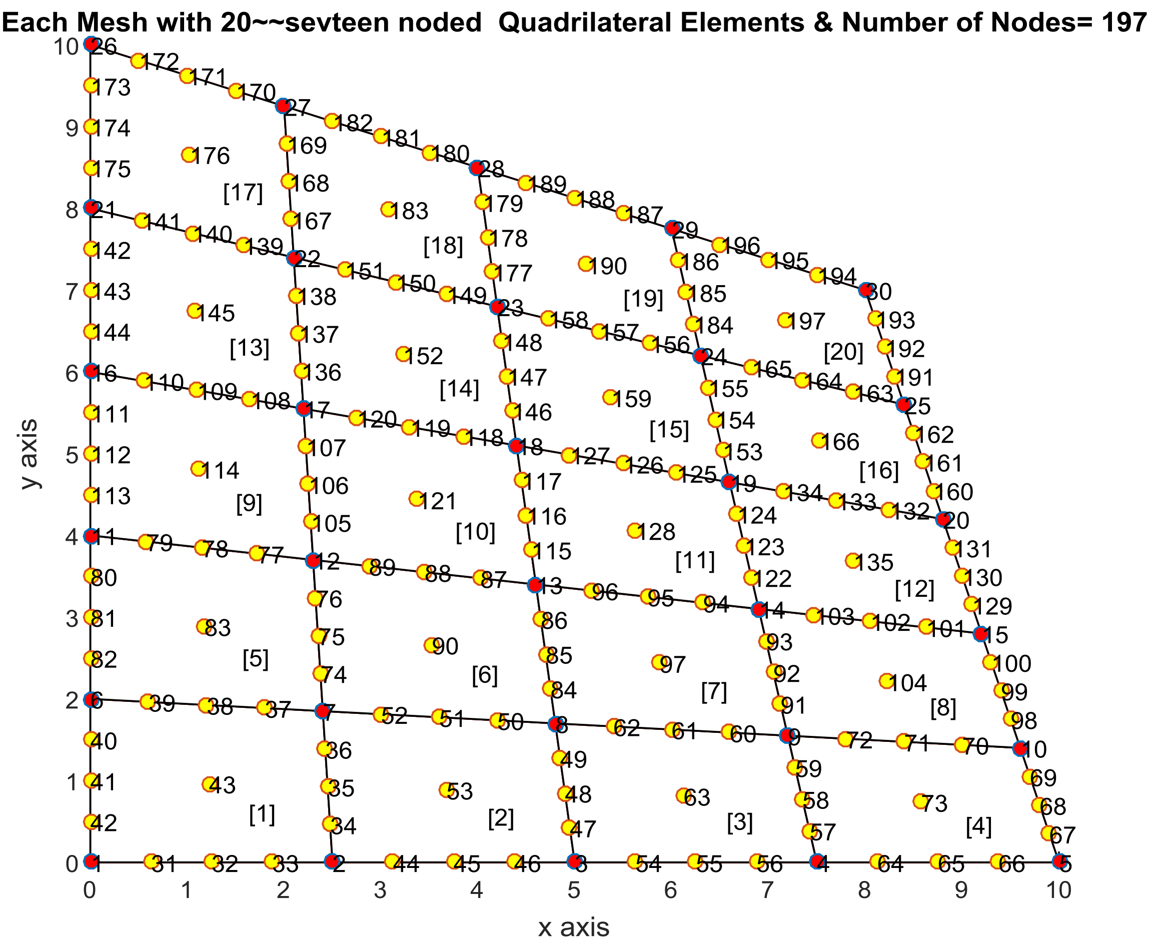


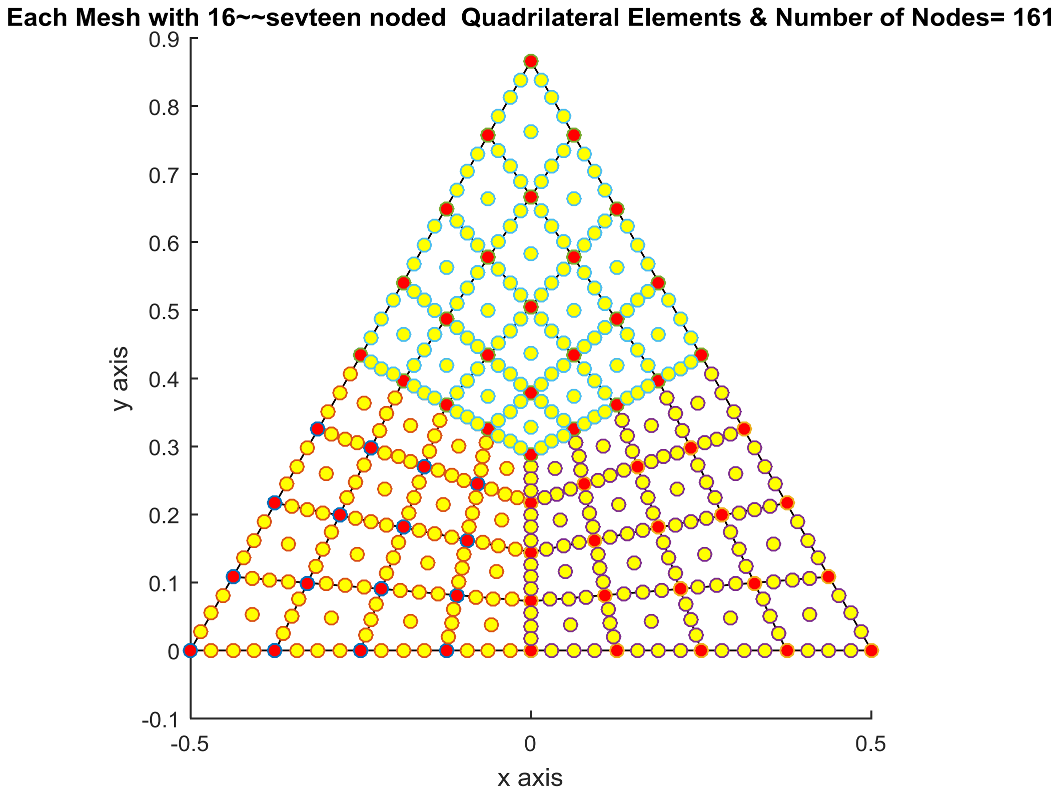


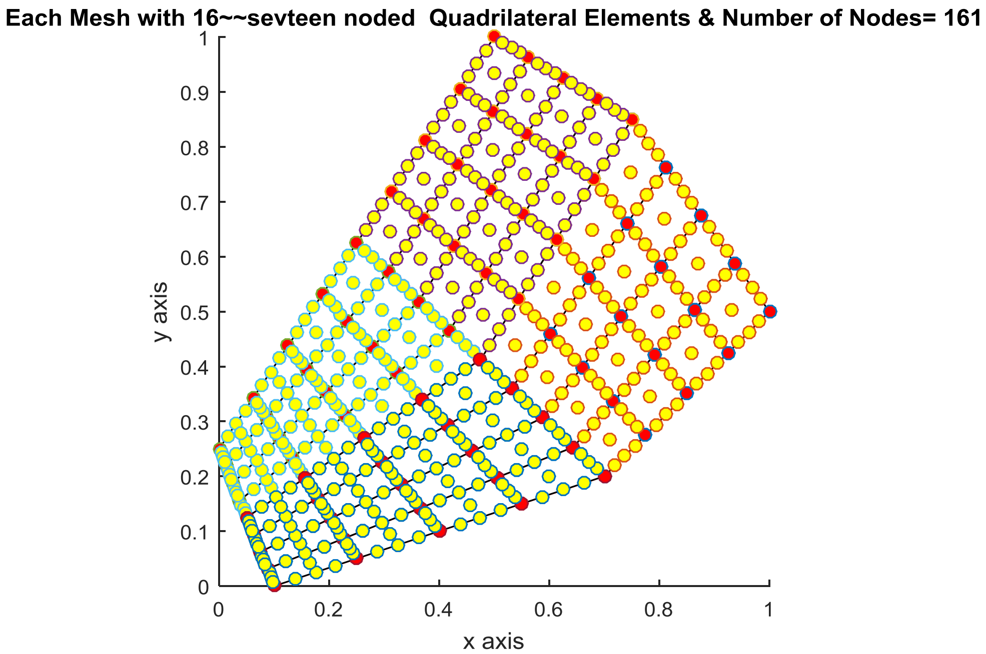


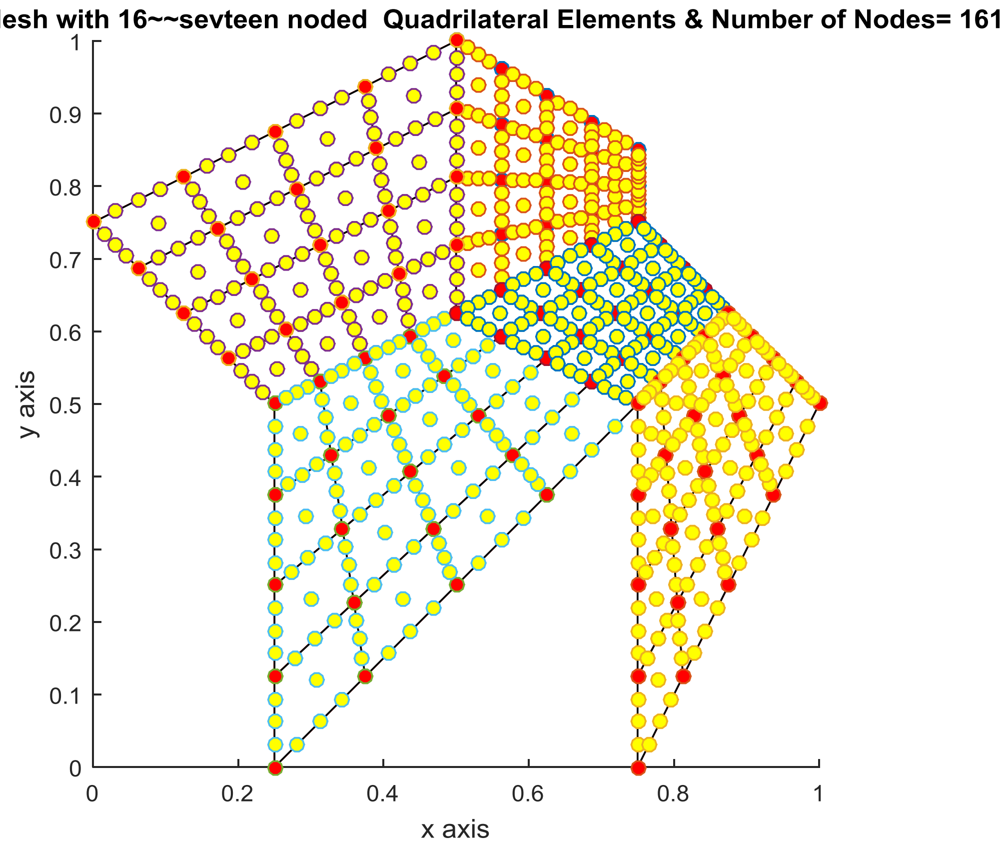


(6) 17-node Complete Lagrange quadrilateral

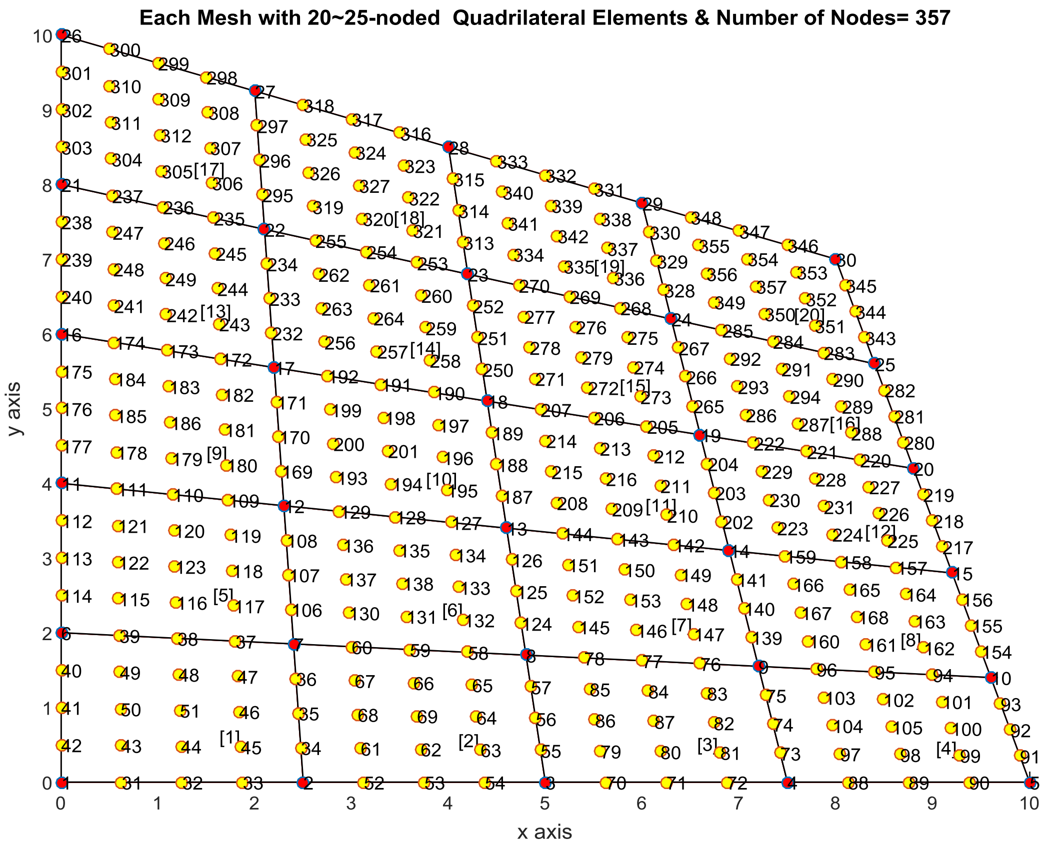


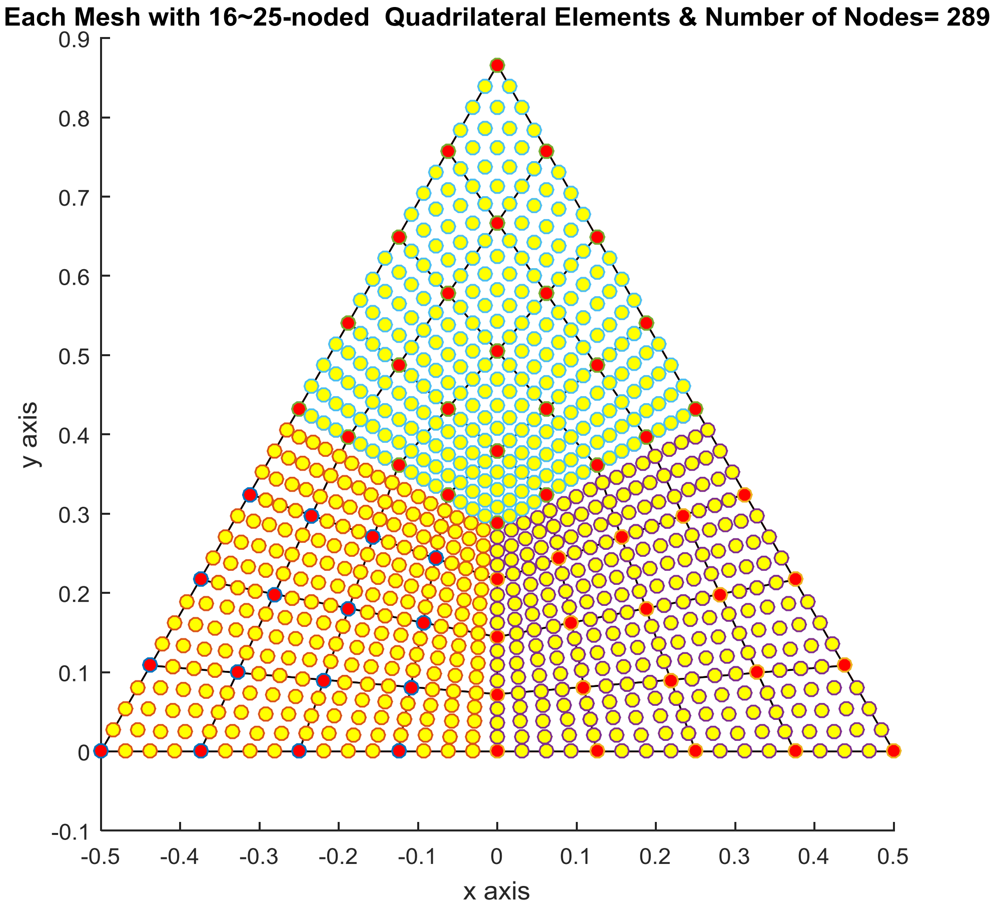

Each Mesh with 16~~sevteen noded Quadrilateral Elements & Number of Nodes= 161
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
y axis
-0.5
0
0.5
x axis

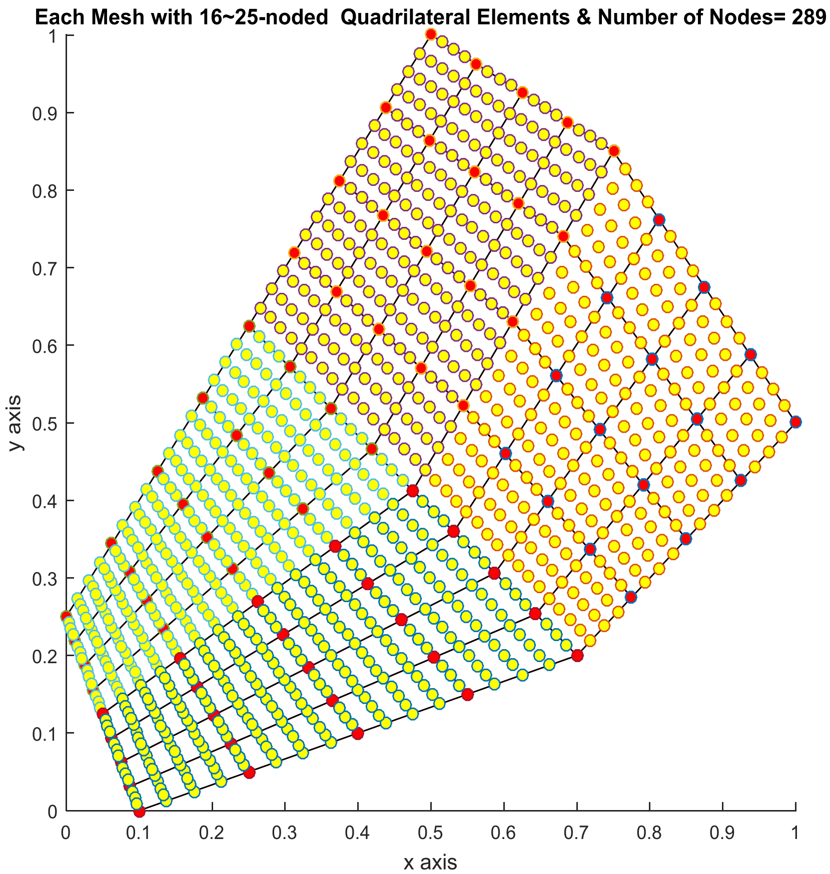

Each Mesh with 16~~sevteen noded Quadrilateral Elements & Number of Nodes= 161
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
y axis
0
0.2
0.4
0.6
0.8
1
x axis

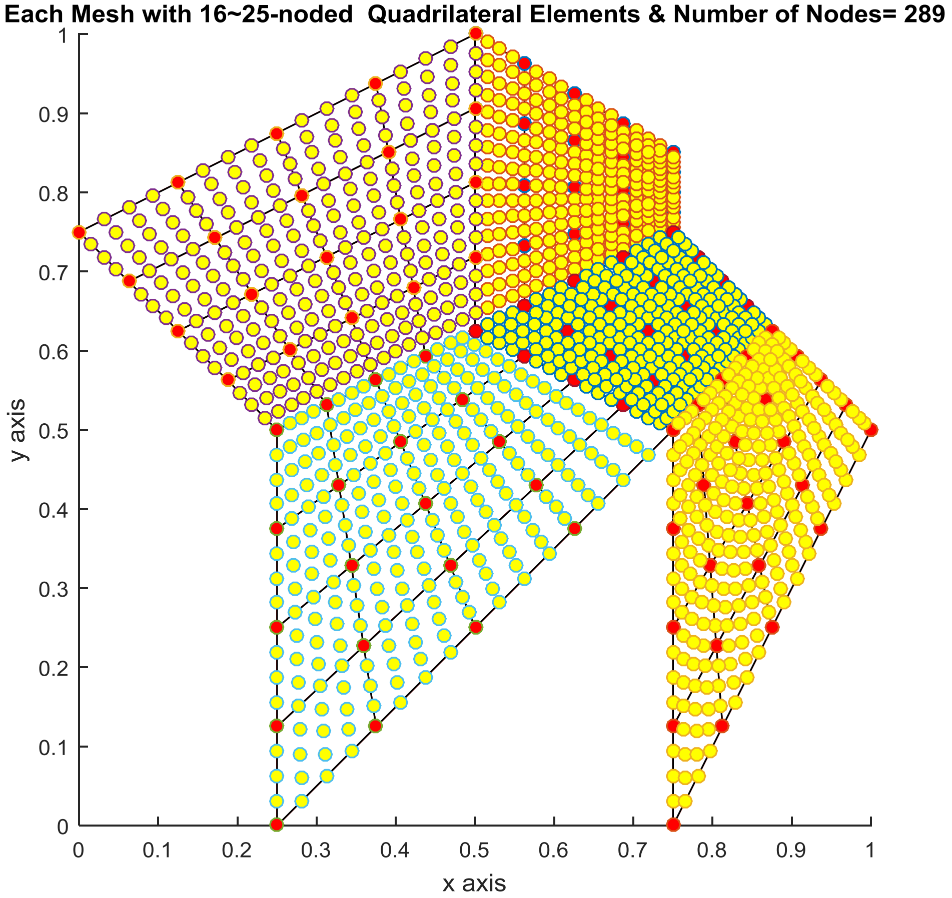


(7) 25-node Lagrange Quadrilateral Elements




Each Mesh with 16~25-noded Quadrilateral Elements & Number of Nodes= 289
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
-0.1
y axis
-0.5
-0.4
-0.3
-0.2
-0.1
0
0.1
0.2
0.3
0.4
0.5
x axis


Each Mesh with 16~25-noded Quadrilateral Elements & Number of Nodes= 289
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
y axis
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
x axis



MATLAB CODES

Code (1)

```
function[]=FEMmeshExample4triangleNquadrilateral17node(gdata)
%This code generates NE triangular elements  NE/2 Quadrilateral elements
%length = Ly units and width = Lx units with Nx divisions on the x axis
% Ny=NE/(2*Nx);    %Divisions on y axis
% cla
    N=0;
   switch gdata
   case 1
   Lx=1;
   Ly=1;
   Nx=8;
   NE=144;
   X=[0;10;8; 0]
   Y=[0; 0;7;10]
   hdata=gdata
   case 2
   Lx=1;
   Ly=1;
   Nx=4;
   NE=40;
   X=[0;10;8; 0]
   Y=[0; 0;7;10]
   hdata=gdata
   case 3
   Lx=1;
   Ly=1;
   Nx=2;
   NE=8;
   X=[0;10;8; 0]
   Y=[0; 0;7;10]
   hdata=gdata
   case 4
   Lx=1;
   Ly=1;
   Nx=16;
```

```
    NE=288;
    X=[0;10;8; 0]
    Y=[0; 0;7;10]
    hdata=gdata
     case 5
    Lx=1;
    Ly=1;
    Nx=16;
    NE=288;
    X=[0;1;1;0]
    Y=[0;0;1;1]
    hdata=gdata
    case 6
    Lx=1;
    Ly=1;
    Nx=8;
    NE=144;
    X=[0;1;1;0]
    Y=[0;0;1;1]
    hdata=gdata
   case 7
    Lx=1;
    Ly=1;
    Nx=4;
    NE=40;
    X=[0;1;1;0]
    Y=[0;0;1;1]
    hdata=gdata
    case 8
    Lx=1;
    Ly=1;
    Nx=2;
    NE=8;
    X=[0;1;1;0]
    Y=[0;0;1;1]
    hdata=gdata
        case 9%beginning-Q1
    Lx=1;
    Ly=1;
    Nx=10;
    NE=160;
    X=[0;10;5;0]
    Y=[0;0;10;10]
    hdata=9
         case 10%beginning-Q2
    Lx=1;
    Ly=1;
    Nx=10;
    NE=160;
    X=[-10;0;0;-5]
    Y=[  0;0;10;10]
    hdata=9
          case 11%beginning-Q3
    Lx=1;
    Ly=1;
    Nx=10;
    NE=160;
    X=[0;-10; -5;  0]
    Y=[0;  0;-10;-10]
    hdata=9
           case 12%beginning-Q4
    Lx=1;
    Ly=1;
    Nx=10;
    NE=160;
    X=[10;0;  0;  5]
    Y=[ 0;0;-10;-10]
    hdata=9
        case 13%6-node convex polygonQ1
     Lx=1;    Ly=1;    Nx=2;    NE=8;
     Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
Lx=1;    Ly=1;    Nx=4;    NE=32;N=4
     A1=   0;A2=  .05;A3=0.1;A4=0.7;A5=  1;  A6=0.75;A7=0.5;A8= 0.25;A9=0.95/2 ;
     B1=0.25;B2=0.125;B3=  0;B4=0.2;B5=0.5;  B6=0.85;B7=  1;B8=0.625;  ;B9=0.825/2 ;
    X=[A9;A4;A5;A6]
    Y=[B9;B4;B5;B6]
    hdata=10
       case 14%6-node convex polygonQ2
     Lx=1;    Ly=1;    Nx=2;    NE=8;
     Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
Lx=1;    Ly=1;    Nx=4;    NE=32;N=4
     A1=   0;A2=  .05;A3=0.1;A4=0.7;A5=  1;  A6=0.75;A7=0.5;A8= 0.25;A9=0.95/2 ;
```

```
      B1=0.25;B2=0.125;B3=  0;B4=0.2;B5=0.5;  B6=0.85;B7=  1;B8=0.625;  ;B9=0.825/2 ;
     X=[A8;A9;A6;A7]
     Y=[B8;B9;B6;B7]
     hdata=10
        case 15%6-node convex polygonQ3
      Lx=1;    Ly=1;    Nx=2;    NE=8;
      Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
Lx=1;    Ly=1;    Nx=4;    NE=32;N=4
      A1=   0;A2=  .05;A3=0.1;A4=0.7;A5=  1;  A6=0.75;A7=0.5;A8= 0.25;A9=0.95/2 ;
      B1=0.25;B2=0.125;B3=  0;B4=0.2;B5=0.5;  B6=0.85;B7=  1;B8=0.625;  ;B9=0.825/2 ;
     X=[A9;A8;A1;A2]
     Y=[B9;B8;B1;B2]
     hdata=10
         case 16%6-node convex polygonQ4
      Lx=1;    Ly=1;    Nx=2;    NE=8;
      Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
Lx=1;    Ly=1;    Nx=4;    NE=32;N=4
      A1=   0;A2=  .05;A3=0.1;A4=0.7;A5=  1;  A6=0.75;A7=0.5;A8= 0.25;A9=0.95/2 ;
      B1=0.25;B2=0.125;B3=  0;B4=0.2;B5=0.5;  B6=0.85;B7=  1;B8=0.625;  ;B9=0.825/2 ;
     X=[A4;A9;A2;A3]
     Y=[B4;B9;B2;B3]
     hdata=10
  %=========6-node convex polygon discritised into a coarse mesh of four quadrilateral with no 
subdivisions==============
         case 17%6-node convex polygonQ1
      Lx=1;    Ly=1;    Nx=1;    NE=2;
      A1=   0;A2=  .05;A3=0.1;A4=0.7;A5=  1;  A6=0.75;A7=0.5;A8= 0.25;A9=0.95/2 ;
      B1=0.25;B2=0.125;B3=  0;B4=0.2;B5=0.5;  B6=0.85;B7=  1;B8=0.625;  ;B9=0.825/2 ;
     X=[A9;A4;A5;A6]
     Y=[B9;B4;B5;B6]
     hdata=11
        case 18%6-node convex polygonQ2
      Lx=1;    Ly=1;    Nx=1;    NE=2;
      A1=   0;A2=  .05;A3=0.1;A4=0.7;A5=  1;  A6=0.75;A7=0.5;A8= 0.25;A9=0.95/2 ;
      B1=0.25;B2=0.125;B3=  0;B4=0.2;B5=0.5;  B6=0.85;B7=  1;B8=0.625;  ;B9=0.825/2 ;
     X=[A8;A9;A6;A7]
     Y=[B8;B9;B6;B7]
     hdata=11
        case 19%6-node convex polygonQ3
      Lx=1;    Ly=1;    Nx=1;    NE=2;
      A1=   0;A2=  .05;A3=0.1;A4=0.7;A5=  1;  A6=0.75;A7=0.5;A8= 0.25;A9=0.95/2 ;
      B1=0.25;B2=0.125;B3=  0;B4=0.2;B5=0.5;  B6=0.85;B7=  1;B8=0.625;  ;B9=0.825/2 ;
     X=[A9;A8;A1;A2]
     Y=[B9;B8;B1;B2]
     hdata=11
         case 20%6-node convex polygonQ4
      Lx=1;    Ly=1;    Nx=1;    NE=2;
      A1=   0;A2=  .05;A3=0.1;A4=0.7;A5=  1;  A6=0.75;A7=0.5;A8= 0.25;A9=0.95/2 ;
      B1=0.25;B2=0.125;B3=  0;B4=0.2;B5=0.5;  B6=0.85;B7=  1;B8=0.625;  ;B9=0.825/2 ;
     X=[A4;A9;A2;A3]
     Y=[B4;B9;B2;B3]
     hdata=11
    %=====================================================
         case 21%standard square
     Lx=1;    Ly=1;    Nx=1;    NE=2;
    X=[-1; 1;1; -1]
    Y=[-1;-1;1;  1]
    hdata=12
         case 22%arbitrary quadrilateral
     Lx=1;    Ly=1;    Nx=1;    NE=2;
    X=[-1;2;3;1]
    Y=[ 2;1;3;4]
    hdata=13
    %+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
    case 23%Q1<11,5,6,7 >==>FIRST QUADRILATERAL OF NINE NODE NONCONVEX POLYGON
Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
Lx=1;    Ly=1;    Nx=4;    NE=32;N=4;
%Lx=1;    Ly=1;    Nx=8;    NE=256;N=8;

 A1=0.25;A2=0.75;A3=0.75;A4=  1;A5=0.75;A6=0.75;A7=0.5;A8=   0;A9=0.25;A10=1.75/2;A11=   0.5;
 B1=   0;B2= 0.5;B3=   0;B4=0.5;B5=0.75;B6=0.85;B7=  1;B8=0.75;B9= 0.5;B10=1.25/2;B11=1.25/2;
    X=[A11; A5; A6; A7];
    Y=[B11; B5; B6; B7];
hdata=14

case 24%Q2<9,11,7,8>==>SECOND QUADRILATERAL OF NINE NODE NONCONVEX POLYGON
    Lx=1;    Ly=1;    Nx=2;    NE=8;
    Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
Lx=1;    Ly=1;    Nx=4;    NE=32;N=4 ;
%Lx=1;    Ly=1;    Nx=8;    NE=256;N=8;
 A1=0.25;A2=0.75;A3=0.75;A4=  1;A5=0.75;A6=0.75;A7=0.5;A8=   0;A9=0.25;A10=1.75/2;A11=   0.5;
 B1=   0;B2= 0.5;B3=   0;B4=0.5;B5=0.75;B6=0.85;B7=  1;B8=0.75;B9= 0.5;B10=1.25/2;B11=1.25/2;
```

```
    X=[A9; A11; A7; A8];
    Y=[B9; B11; B7; B8];
hdata=14

case 25%Q3<11,9,1,2>==>THIRD QUADRILATERAL OF NINE NODE NONCONVEX POLYGON
    Lx=1;    Ly=1;    Nx=2;    NE=8;
    Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
Lx=1;    Ly=1;    Nx=4;    NE=32;N=4
    A1=0.25;A2=0.75;A3=0.75;A4=  1;A5=0.75;A6=0.75;A7=0.5;A8=   0;A9=0.25;A10=1.75/2;A11=   0.5;
    B1=   0;B2= 0.5;B3=   0;B4=0.5;B5=0.75;B6=0.85;B7=  1;B8=0.75;B9= 0.5;B10=1.25/2;B11=1.25/2;
   X=[A11; A9; A1; A2];
   Y=[B11; B9; B1; B2] ;
hdata=14
case 26%Q4<5;11;2;10>==>FOURTH QUADRILATERAL OF NINE NODE NONCONVEX POLYGON
    Lx=1;    Ly=1;    Nx=2;    NE=8;
    Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
Lx=1;    Ly=1;    Nx=4;    NE=32;N=4 ;
%Lx=1;    Ly=1;    Nx=4;    NE=256;N=8;
  A1=0.25;A2=0.75;A3=0.75;A4=  1;A5=0.75;A6=0.75;A7=0.5;A8=   0;A9=0.25;A10=1.75/2;A11=   0.5;
  B1=   0;B2= 0.5;B3=   0;B4=0.5;B5=0.75;B6=0.85;B7=  1;B8=0.75;B9= 0.5;B10=1.25/2;B11=1.25/2;
   X=[A5; A11; A2; A10];
   Y=[B5; B11; B2; B10];
   hdata=14

case 27%Q5<10;2;3;4>==>FIFTH QUADRILATERAL OF NINE NODE NONCONVEX POLYGON
    Lx=1;    Ly=1;    Nx=2;    NE=8;
    Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
Lx=1;    Ly=1;    Nx=4;    NE=32;N=4 ;
%Lx=1;    Ly=1;    Nx=4;    NE=256;N=8;
  A1=0.25;A2=0.75;A3=0.75;A4=  1;A5=0.75;A6=0.75;A7=0.5;A8=   0;A9=0.25;A10=1.75/2;A11=   0.5;
  B1=   0;B2= 0.5;B3=   0;B4=0.5;B5=0.75;B6=0.85;B7=  1;B8=0.75;B9= 0.5;B10=1.25/2;B11=1.25/2;
    X=[A10; A2; A3; A4];
    Y=[B10; B2; B3; B4];
    hdata=14

 case 28%arbitrary quadrilateral
    Lx=1;    Ly=1;    Nx=4;    NE=40;
   X=[-1;2;3;1]
   Y=[ 2;1;3;4]
   hdata=15
 case 29
    Lx=1;
    Ly=1;
    Nx=4;
    NE=32;
    X=[0;1;1;0]
    Y=[0;0;1;1]
    hdata=16
 case 30%Q1<7;6;1;4>==>FIRST QUADRILATERAL OF EQUILATERAL  TRIANGLE
    Lx=1;    Ly=1;    Nx=2;    NE=8;
    Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
    Lx=1;    Ly=1;    Nx=4;    NE=32;N=4 ;
%Lx=1;    Ly=1;    Nx=4;    NE=256;N=8;
  A1=-0.5;A2=0.5;A3=        0;A4=0;A5=    0.25;A6=    -0.25;A7=0;
  B1=   0;B2=  0;B3=sqrt(3)/2;B4=0;B5=sqrt(3)/4;B6=sqrt(3)/4;B7=sqrt(3)/6;
    X=[A7; A6; A1; A4];
    Y=[B7; B6; B1; B4];
    hdata=17
case 31%Q2<7;4;2;5>==>SECOND QUADRILATERAL OF EQUILATERAL  TRIANGLE
    Lx=1;    Ly=1;    Nx=2;    NE=8;
    Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
    Lx=1;    Ly=1;    Nx=4;    NE=32;N=4 ;
%Lx=1;    Ly=1;    Nx=4;    NE=256;N=8;
  A1=-0.5;A2=0.5;A3=        0;A4=0;A5=    0.25;A6=    -0.25;A7=0;
  B1=   0;B2=  0;B3=sqrt(3)/2;B4=0;B5=sqrt(3)/4;B6=sqrt(3)/4;B7=sqrt(3)/6;
    X=[A7; A4; A2; A5];
    Y=[B7; B4; B2; B5];
    hdata=17
      case 32%Q3<7;5;3;6>==>THIRD QUADRILATERAL OF EQUILATERAL  TRIANGLE
    Lx=1;    Ly=1;    Nx=2;    NE=8;
    Lx=1;    Ly=1;    Nx=2;    NE=8;N=2;
    Lx=1;    Ly=1;    Nx=4;    NE=32;N=4 ;
%Lx=1;    Ly=1;    Nx=4;    NE=256;N=8;
  A1=-0.5;A2=0.5;A3=        0;A4=0;A5=    0.25;A6=    -0.25;A7=0;
  B1=   0;B2=  0;B3=sqrt(3)/2;B4=0;B5=sqrt(3)/4;B6=sqrt(3)/4;B7=sqrt(3)/6;
    X=[A7; A5; A3; A6];
    Y=[B7; B5; B3; B6];
    hdata=17
    end
    %[gcoord,coords,cT,nNodes]=femTriangularMeshGenerator(Lx,Ly,Nx,NE);
```

```
[xygcoord,xycoords,xycoordrgqd,xycoordrgqdm,cT,qT,nNodes]=femTriangularMeshGenerator4triangleNquadrilateral17
node(Lx,Ly,Nx,NE,X,Y)
   % return
    [nnode,dimension]=size(xygcoord)
    disp(['Number of nodes =  ',num2str(nnode)])
    disp('Connectivity Table')
    disp(cT)
    disp(qT)
    axis square
   %axis equal
    z=1;
    for i=1:NE
        figure(2*hdata-
1),patch('Vertices',xycoords(z:z+2,:),'Faces',[1,2,3],'FaceColor','none','EdgeColor','g')
        if Nx<9
        midx=mean(xycoords(z:z+2,1));
        midy=mean(xycoords(z:z+2,2));
        text(midx,midy,['[',num2str(i),']']);
        end
        hold on
        z=z+3;
    end
    hold on
xlabel('x axis')
ylabel('y axis')
st1='FEM MESH WITH  ';
st2=num2str(NE);
st3='; 3-node Linear ';
st4='Triangular';
st5=' Elements'
st6='& Nodes='
st7=num2str(nNodes);
title([st1,st2,st3,st4,st5,st6,st7])

figure(2*hdata-1),scatter(xycoords(:,1),xycoords(:,2),'MarkerFaceColor','r')

    hold on
    %put node numbers
if Nx<9
for jj=1:nNodes
text(xygcoord(jj,1),xygcoord(jj,2),['.',num2str(jj)]);
end
end
disp('nodal  connectivity  for seventeen noded quartic convex quadrilaterals ')
disp(qT) %

  z=1;
    for i=1:NE/2

figure(2*hdata),patch('Vertices',xycoordrgqd(z:z+3,:),'Faces',[1,2,3,4],'FaceColor','none','EdgeColor','r')
        xx=xycoordrgqd(z:z+3,1); yy=xycoordrgqd(z:z+3,2);
        hold on
        patch(xx,yy,'w')
        if (Nx<9)&(Nx~=N)
        midx=mean(xycoordrgqd(z:z+2,1));
        midy=mean(xycoordrgqd(z:z+2,2));

        text(midx,midy,['[',num2str(i),']']);
        end
        hold on
        z=z+4;
    end


    figure(2*hdata),scatter(xycoordrgqd(:,1),xycoordrgqd(:,2),'MarkerFaceColor','r')
    figure(2*hdata),scatter(xycoordrgqdm(:,1),xycoordrgqdm(:,2),'MarkerFaceColor','y')
    hold on
    %put node numbers


if (Nx<9)&(Nx~=N)
for jj=1:nnode
text(xygcoord(jj,1),xygcoord(jj,2),num2str(jj));
end
end

hold on
xlabel('x axis')
ylabel('y axis')
```

```
st1='Each Mesh with ';
st2=num2str(NE/2);
st3='~~sevteen noded  ';
st4='Quadrilateral';
st5=' Elements ';
st6='& Number of Nodes= ';
st7=num2str(nnode);
title([st1,st2,st3,st4,st5,st6,st7])
end
%===========================
```

Code (2)

```
function [xygcoord,xycoords,xycoordrgqd,xycoordrgqdm,cT,qT,nNodes]=femTriangularMeshGenerator4triangleNquadrilateral17node(Lx,Ly,Nx,NE,X,Y)
    %   This function generates triangular mesh for a rectangular
    %   shape structure for finite element analysis
    %   [coords cT nNodes ]=femTriangularMeshGenerator(Lx,Ly,Nx,NE)
    %   coords  =   x and y coordinates of each element nodes
    %   cT      =   nodal connectivity
    %   nNodes  =   Number of nodes
    %   Lx      =   width of the rectangular structure
    %   Ly      =   Height of the rectangular structure
    %   Nx      =   Number of divisions on x- axis
    %   NE      =   Number of elements
    %

    x1=X(1,1);
    x2=X(2,1);
    x3=X(3,1);
    x4=X(4,1);
    y1=Y(1,1);
    y2=Y(2,1);
    y3=Y(3,1);
    y4=Y(4,1);

    if mod((NE/Nx),2)~=0
        errordlg('The No of divisions on X axis must divide No of Elements twice')

    end

    Ny=NE/(2*Nx);   %Divisions on y axis

    nNodes =(Nx+1)*(Ny+1);  %No of nodes

     m=1;
    j=(1:Nx);
    k=linspace(Nx*2,NE,Ny);

    for i=1:Ny
        cT(m:2:k(i),1)= j';  %node 1 of 1st element
        cT(m+1:2:k(i),1)= j';  %node 1 of 2nd element
        cT(m:2:k(i),2)=(j+1)'; %%node 2 of 1st element
        cT(m+1:2:k(i),2)=(j+Nx+2)';%node 2 of 2nd element
        cT(m:2:k(i),3)= (j+Nx+2)';  %node 3 of 1st element
        cT(m+1:2:k(i),3)=(j+1+Nx)';    %%node 3 of 1st element

        m=k(i)+1;
        j=j+Nx+1;
    end
    for m=1:NE/2
        qT(m,1)=cT(2*m-1,1);
        qT(m,2)=cT(2*m-1,2);
        qT(m,3)=cT(2*m-1,3);
        qT(m,4)=cT(2*m,3);
    end


    %%%%%%%%%%%%%%%COORDINATES GENERATION%%%%%%%%%%%%%%%%%%%

    ax=linspace(0,Lx,Nx+1); %%%x coordinates
    by=linspace(0,Ly,Ny+1); %%%y coordinates
    X1=[];
    Y1=[];
    for i1=1:Ny+1
        %    General Nodal Coordinates layer by layer
        by1(1:Nx+1)=by(i1);
        X1=[X1 ax];
        Y1=[Y1 by1];
    end
   % disp('X1=')
   %[X1]
```

```
   %disp('Y1=')
   %[Y1]
   gcoord(:,1)=X1';
   gcoord(:,2)=Y1';
   NN=(1:nNodes)';
   [NN gcoord]



    j=1:3;
    %each element coordinates for triangles
    for n=1:NE
        X(j,1) = X1(cT(n,:));
        Y(j,1)=Y1(cT(n,:));
        j=j+3;
    end
    coords=[X Y];   %x and y coordinates for triangles
      j=1:4;
    %each element coordinates for quadrilaterals
    for n=1:NE/2
        XX(j,1) = X1(qT(n,:));
        YY(j,1)=Y1(qT(n,:));
        j=j+4;
    end
    coord=[XX YY];   %x and y coordinates for quadrilaterals
  %^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
 % mesh generation of 16-node special quadrilaterals
nnd=nNodes;
for inum=1:nnd
    for jnum=1:nnd
        trisect(inum,jnum)=0;
        mdpt(inum,jnum)=0;
    end
end
nd=nnd;mm=NE/2;
for mmm=1:mm
mmm1=qT(mmm,1);
mmm2=qT(mmm,2);
mmm3=qT(mmm,3);
mmm4=qT(mmm,4);

%trisectional points: side-1 of 4-node  quadrilateral
if((trisect(mmm1,mmm2)==0))
    nd=nd+1;
   trisect(mmm1,mmm2)=nd;
end
if((mdpt(mmm1,mmm2)==0)&&(mdpt(mmm2,mmm1)==0))
    nd=nd+1;
   mdpt(mmm1,mmm2)=nd;
   mdpt(mmm2,mmm1)=nd;
end
if((trisect(mmm1,mmm2)~=0)&&(trisect(mmm2,mmm1)==0))
    nd=nd+1;
   trisect(mmm2,mmm1)=nd;
end


% trisectional points: side-2 of 4-node  quadrilateral

if((trisect(mmm2,mmm3)==0))
    nd=nd+1;
   trisect(mmm2,mmm3)=nd;
end
if((mdpt(mmm2,mmm3)==0)&&(mdpt(mmm3,mmm2)==0))
    nd=nd+1;
   mdpt(mmm2,mmm3)=nd;
   mdpt(mmm3,mmm2)=nd;
end
if((trisect(mmm2,mmm3)~=0)&&(trisect(mmm3,mmm2)==0))
    nd=nd+1;
   trisect(mmm3,mmm2)=nd;
end

% trisectional points: side-3 of 4-node  quadrilateral

if((trisect(mmm3,mmm4)==0))
    nd=nd+1;
   trisect(mmm3,mmm4)=nd;
end
if((mdpt(mmm3,mmm4)==0)&&(mdpt(mmm4,mmm3)==0))
```

```
    nd=nd+1;
   mdpt(mmm3,mmm4)=nd;
   mdpt(mmm4,mmm3)=nd;
end
if((trisect(mmm3,mmm4)~=0)&&(trisect(mmm4,mmm3)==0))
    nd=nd+1;
   trisect(mmm4,mmm3)=nd;
end

% trisectional points: side-4 of 4-node  quadrilateral

if((trisect(mmm4,mmm1)==0))
    nd=nd+1;
   trisect(mmm4,mmm1)=nd;
end
if((mdpt(mmm4,mmm1)==0)&&(mdpt(mmm1,mmm4)==0))
    nd=nd+1;
   mdpt(mmm4,mmm1)=nd;
   mdpt(mmm1,mmm4)=nd;
end
if((trisect(mmm4,mmm1)~=0)&&(trisect(mmm1,mmm4)==0))
    nd=nd+1;
   trisect(mmm1,mmm4)=nd;
end



   qT(mmm,5)=trisect(mmm1,mmm2);
   qT(mmm,6)=mdpt(mmm1,mmm2);
   qT(mmm,7)=trisect(mmm2,mmm1);
   %
   qT(mmm,8)=trisect(mmm2,mmm3);
   qT(mmm,9)=mdpt(mmm2,mmm3);
   qT(mmm,10)=trisect(mmm3,mmm2);

   %
   qT(mmm,11)=trisect(mmm3,mmm4);
   qT(mmm,12)=mdpt(mmm3,mmm4);
   qT(mmm,13)=trisect(mmm4,mmm3);
   %
   qT(mmm,14)=trisect(mmm4,mmm1);
   qT(mmm,15)=mdpt(mmm4,mmm1);
   qT(mmm,16)=trisect(mmm1,mmm4);
   %
   nd=nd+1;qT(mmm,17)=nd;


end%for

nnode=nd;
nel=mm;
% % spqd=nodes;
 MM=(1:mm)';
  disp([MM qT])
  [nel,nnel]=size(qT)
 % return
for mmm=1:nel
   mmm1=qT(mmm,1);
   mmm2=qT(mmm,2);
   mmm3=qT(mmm,3);
   mmm4=qT(mmm,4);
   mmm5=qT(mmm,5) ;
   mmm6=qT(mmm,6);
   mmm7=qT(mmm,7);
   mmm8=qT(mmm,8);
   mmm9=qT(mmm,9);
   mmm10=qT(mmm,10);
   mmm11=qT(mmm,11);
   mmm12=qT(mmm,12);    %
   mmm13=qT(mmm,13);
   mmm14=qT(mmm,14);
   mmm15=qT(mmm,15);
   mmm16=qT(mmm,16);
   mmm17=qT(mmm,17);

   %

   xi1=gcoord(mmm1,1);
   xi2=gcoord(mmm2,1);
   xi3=gcoord(mmm3,1);
```

```
   xi4=gcoord(mmm4,1);
   %
   yi1=gcoord(mmm1,2);
   yi2=gcoord(mmm2,2);
   yi3=gcoord(mmm3,2);
   yi4=gcoord(mmm4,2);
   %
 gcoord(mmm5,1)=xi1+(xi2-xi1)/4;gcoord(mmm6,1)=xi1+(xi2-xi1)/2; gcoord(mmm7,1)=xi1+3*(xi2-xi1)/4;
 gcoord(mmm5,2)=yi1+(yi2-yi1)/4;gcoord(mmm6,2)=yi1+(yi2-yi1)/2; gcoord(mmm7,2)=yi1+3*(yi2-yi1)/4;
 %
 gcoord(mmm8,1)=xi2+(xi3-xi2)/4; gcoord(mmm9,1)=xi2+(xi3-xi2)/2;gcoord(mmm10,1)=xi2+3*(xi3-xi2)/4;
 gcoord(mmm8,2)=yi2+(yi3-yi2)/4; gcoord(mmm9,2)=yi2+(yi3-yi2)/2;gcoord(mmm10,2)=yi2+3*(yi3-yi2)/4;
 %
 gcoord(mmm11,1)=xi3+(xi4-xi3)/4;gcoord(mmm12,1)=xi3+(xi4-xi3)/2; gcoord(mmm13,1)=xi3+3*(xi4-xi3)/4;
 gcoord(mmm11,2)=yi3+(yi4-yi3)/4;gcoord(mmm12,2)=yi3+(yi4-yi3)/2; gcoord(mmm13,2)=yi3+3*(yi4-yi3)/4;
 %
 gcoord(mmm14,1)=xi4+(xi1-xi4)/4;gcoord(mmm15,1)=xi4+(xi1-xi4)/2;gcoord(mmm16,1)=xi4+3*(xi1-xi4)/4;
 gcoord(mmm14,2)=yi4+(yi1-yi4)/4;gcoord(mmm15,2)=yi4+(yi1-yi4)/2;gcoord(mmm16,2)=yi4+3*(yi1-yi4)/4;
 %
 gcoord(mmm17,1)=(xi1+xi2+xi3+xi4)/4;gcoord(mmm17,2)=(yi1+yi2+yi3+yi4)/4;
end%for nel
disp(gcoord)
[nnode,dimension]=size(gcoord)
%============================================================================================
  j=1:13;
    %each element coordinates for quadrilaterals at midside nodes
    for n=1:NE/2
        XM(j,1)=gcoord(qT(n,5:17),1);
        YM(j,1)=gcoord(qT(n,5:17),2);
        j=j+13;
    end
    coordm=[XM YM];   %x and y coordinates for quadrilaterals  at midside  nodes

    for ii=1:nnode
        r=gcoord(ii,1);s=gcoord(ii,2);
        xygcoord(ii,1)=x1+r*(x2-x1)+s*(x4-x1)+r*s*(x1-x2+x3-x4);
        xygcoord(ii,2)=y1+r*(y2-y1)+s*(y4-y1)+r*s*(y1-y2+y3-y4);

    end
   kk=0;
    for n=1:NE
        for jj=1:3
            kk=kk+1
            rr=coords(kk,1);ss=coords(kk,2);
            xycoords(kk,1)=x1+rr*(x2-x1)+ss*(x4-x1)+rr*ss*(x1-x2+x3-x4);
            xycoords(kk,2)=y1+rr*(y2-y1)+ss*(y4-y1)+rr*ss*(y1-y2+y3-y4);
        end

    end

    kk=0;
    for n=1:NE/2
        for jj=1:4
            kk=kk+1
            rr=coord(kk,1);ss=coord(kk,2);
            xycoordrgqd(kk,1)=x1+rr*(x2-x1)+ss*(x4-x1)+rr*ss*(x1-x2+x3-x4);
            xycoordrgqd(kk,2)=y1+rr*(y2-y1)+ss*(y4-y1)+rr*ss*(y1-y2+y3-y4);
        end

    end
    %coordinates for the element midside nodes
   kk=0;
    for n=1:NE/2
        for jj=1:13
            kk=kk+1;
            rr=coordm(kk,1);ss=coordm(kk,2);
            xycoordrgqdm(kk,1)=x1+rr*(x2-x1)+ss*(x4-x1)+rr*ss*(x1-x2+x3-x4);
            xycoordrgqdm(kk,2)=y1+rr*(y2-y1)+ss*(y4-y1)+rr*ss*(y1-y2+y3-y4);
        end

    end



end
%=============================================================================
```