ADAPTIVE UNSTRUCTURED GRID GENERATION SCHEME FOR SOLUTION OF THE HEAT EQUATION

K. Mazaheri, A. Banai and S. A. Seyed-Reyhani

*Department of Mechanical Engineering Sharif Univ. of Tech.
Tehran, Iran, mazaheri@sina.sharif.ac.ir*

(Recived: April 14, 1997 - Accepted in Final Form: June 1, 2000)

Abstract An adaptive unstructured grid generation scheme is introduced to use finite volume (FV) and finite element (FE) formulation to solve the heat equation with singular boundary conditions. Regular grids could not acheive accurate solution to this problem. The grid generation scheme uses an optimal time complexity frontal method for the automatic generation and delaunay triangulation of the grid points. The algorithm is incremental, so it is the most appropriate for an adaptive solver. Using adaptive grids, the solution is refined to get enough accuracy in all grid points. Two schemes are applied for the solution of the equations to show the flexibility of the adaptive grid scheme. First a cell-vertex finite volume formulation is used. Then, for the FE scheme, using linear shape functions, a set of linear equations are solved explicitly, with overrelaxation. A sequence of adaptation is applied and appropriate number of grid points are introduced in finite predetermined formats to the existent elements, till convergence in the solution is observed. A postprocess is used to smooth the distribution of the set of nodes. This procedure is applied to a few study cases to show that the method is convergent, and produces accurate solution even in the case of singular boundary conditions.

Key Words Delaunay Grid, Unstructured, Heat Equation

j°kdÇ« ºAq]A ° j°kd« ©\e ºB´{°n pA ²jB´TwA ºAoM »¸¼L¤U ¬B«pBw »M ³ñL{ k¼§±U x °n ð ½ **²k¼ña**
kÇ¯A±U»ªÇ¯ ºA³§BÇv« Ë¼®Ça nj »§±ªí« ³ñL{ /SwA ²k{ »µoí« jkíT« ºpo« £½Ao{ BM RnAoe ³§jBí« ¥e ºAoM
ºAoM An k{BM»« ³®¼´M Ao]A ¬B«p o¨¯ pA ³½ ²k¯°n y ¼Q ³´L] x °n ð ½ ³ñL{ k¼§±U x °n /k½Bª¯ jB\½A ·¼¹j J A±]
ºjkÇî ¥e ºAoM ° ²j±M »z½AqµA ²k{ ²jB´TwA ©T½n±¢§A /jo¼£»« nBñM B´¯C »¯¿j ºk®MW¦X« ° ¢B¸¯nB½j±i k¼§±U
J±¦¤« S¹j ³M ¬k¼wn BU »µB½ nk¹ ³M An ³ñL{ ¬A±U»« »¸¼L¤U ³ñL{ pA ²jB´TwA BM /k{BM»« KwB®« nB¼vM »¸¼L¤U
³ ¦Th« ºjkî x °n °j »¸¼L¤U ³ñL{ ©T½n±¢§A ºo½mQ ² B¤í¯A ¬jAj ¬Bz¯ ºAoM /j±ª¯ q½n ,o¨¯ jn±« ³¤¸¯ oµ nj
¥ñ{ éMA±U pA ²jB´TwA BM kíM ° ²k{ ²jB´TwA »§±¦w t En j°kd« ©\e ³¤MAn ð ½ AkTMA /SwA ²k{ ²jB´TwA ¥e ºAoM
k½k] ¢B¸¯ »µB½ nAk¸« ³M ° ³Tµn nBñM »§A±T« ·¼L¤U ³¦v¦w ð ½ /SwA ²k{ ³Tµo£ nBñM j°kd« ºAq]A x °n ,»¤i
Ë½A /jpBw»« nA±ªµ An ¢B¸¯é½p±U x pAjoQ u Q x °n ð ½ /SwA ²k{ »µoí« ²k{ Ë¼¼íU y ¼Q ºB´¦ñ{ ·L¥ ³ñL{
¬Bz¯ An jo´®« ºpo« ¢o{ S§Be nj ¬C S¹j Hæ±~i ° x °n »½Ao¢ªµ BU ³Tµn nBñM ³¯±ª¯ ³§Bv« k®a ºAoM k®½Coµ
/kµj

INTRODUCTION

Adaptive solution of the Laplace equation in two dimensions has several applications in heat transfer and many other disciplines. It will allow to have fewer number of grid cells, and still keep a maximum acceptable relative truncation error in the whole computational field. Both the unstructured grid and the adaptation procedure have unique features to allow a flexible and robust desirable node distribution. Many investigations are done in both fields of grid generation and solution of the heat equation. Attempts to introduce efficient numerical triangulation schemes are not very old (see [1,2]). Bowyer [3] was the first to introduce an efficient fast Delaunay triangulation method with optimal time complexity. Here we use algorithms following Lawson [4] and Watson [5] which are even better. A comprehensive review is given by Barth [10].

Methods used for numerical solution of Heat equation are very diverse (see [6, 7, 8]. However, these methods are only efficient when gradients in the temperature field are finite. Here we introduce a method which is capable of numerical solution of the heat equation in a domain including regions of infinite gradients of temperature.

In this paper, using a frontal approach, an unstructured grid is generated in the whole field. This grid generation scheme, requires minimum input from the user, and works in optimal time limits. Then finite volume and element formulations are used to discretize the differential equation within the framework of an unstructured grid. Finally, the adaptation allows to have accurate solutions in problems with high solution gradients.

Finite volume methods have been widely used in compressible flow computations. Their main feature is that they conserve mass, momentum and energy, so they provide a robust scheme to capture flow discontinuities [9]. Although discontinuities could not propagate in elliptic fields, the unstructured adaptive grid used here makes it ideal for calculations with discontinuities in the boundary conditions. Here we first review the properties of the Delaunay triangles used in the grid generation procedure. Then, the frontal approach used here will be explained, and finally, the finite volume formulation and the adaptation methods are presented.

GRID GENERATION PROCEDURE

Properties The space discretization could be done in many different ways. If ew select all elements to be triangles, different criteria could be used to establish this discretization. The most famous method, which has shown many theoretical and application properties, is Delaunay triangulation Therefore, it is used in most unstructured mesh computations. It connects each three most closest nodes, in a cloud of points, to each other. The only deficiency of Delaunay triangulation, which is common tomost other methods, is that it requires a points cloud to be given. In many applications, generation of such a cloud of points is not so easy. In this paper, ew will use different ways, including a frontal approach [11] to produce the initial grid and to adapt it to the solution. The Delaunay triangulation is the dual of the *Dirichlet tesselation* of the set of given points. The Dirichlet tesselation is made by determining the *Voroni regions* which are the locus of the closest points of the space, to each point of the colud In two space dimensions, these regions are polygons, mostly with five to seven sides. If we connect each two nodes with neighbor Voroni regions, we will have a Delaunay triangulation. Delaunay triangulation has many different properties [12, 13, 14, 15], which show that it is optimum or suitable for many different applications. some of these properties are:

É Uniqueness: The Delaunay triangulation exists and is unique (except for degenerate cases).

É Out-Circle condition: The circumcircle of a Delaunay triangle respective to three nodes of a given set, does not include any other grid point (except for degenerate cases).

É Max-Min property: Delaunay triangulation maximizes the minimum angle of the triangulation. This property optimizes the grid for many finite element calculations.

É Nearest Neighbor: The closest grid point to grid, is connected to it. This, in fact, shows the solution of the famous Nearest point problem.

É Minimal Roughness: For an arbitrary given data set on grid points, if we project it to the two space dimensions using a linear interpolation function f, the roughness of this surface (its slope in a sense) is minimal if the triangulation is Delaunay.

Numerical Properties to Use Frontal Approach in Delaunay Triangulation The above mentioned properties make it simple, powerful, and optimum to use Delaunay triangulation in many different applications. Given the set of grid points, there are several different methods to (Delaunay) triangulate them. Bowyer [3] introduced an incremental (recursive) algorithm which works optimally in time complexity (i.e. O(n log n)). It uses Voronoi regions for its data structure and triangulation procedure. It starts with an (arbitrary) set of phantom nodes, including the convex hull of the grid points set. Then each new point is added, and necessary changes in the cells with a circumcircle including the new grid point are made. After finding a new Delaunay triangulation, another grid point is added.The algorithm used here is close to those of Lawson [4] and Watso [5]. this uses Delaunay triangles for its data base. It has a preliminary procedure to generate an initial grid, using a frontal approach, and then uses an iterative procedure to adapt the mesh to the solution.

The procedure for generating the initial grid follows:

É Read the data corresponding to the boundary points.

É Make a convex hull using four phantom nodes, including all boundary points, and (Delaunay) triangulate it.

É Add one by one, new boundary points, and retriangulate the field. Each triangle is considered to be a bad triangle, except otherwise marked.

É The common side between bad and good triangles constructs the front. Introduce a new candid node corresponding to each small side of a bad triangle, which is located on the front, and evaluate the spacing function at that point.

É Delete all candid nodes which are too close to permanent nodes.

É Merge all candid nodes which are too close to each other.

É Mark the candid nodes as permanent, and add them one by one to the set of nodes, and retriangulate the field.

É Find the new front, and redo the above procedure, untill there is no front i.e. no bad triangle.

É After construction of the final mesh, use a Laplace filter to smooth the grid points distribution. This involves moving each grid point the area at the center of the neighboring cells.

Data Structure Data structure may strongly affect the overall efficiency of the algorithm. In this work, our records were triangle based, and each record included the triangle number, number of its vertices, its neighbors, and a flag to show if it is a bad or a good triangle. We also used a few small lists and stacks to reduce the time complexity.

Retriangulation This procedure is used both in the frontal method and in the adaptation procedure. Whenever a new node is introduced to the mesh, a local process retriangulate around that new node to make sure that all triangles preserve Delaunay properties. The retriangulation procedure follows:

É Use a search algorithm to find to which old triangle the new node belongs. Walk search is almost the simplest way and shows to be efficient. Other tree search methods are also equally acceptable. Exhaustive search would not be efficient.

É Mark sides of this triangle as the temporary edges of the domain of influence of the new node (we call it TEDI).

É Recursively, check to see if any node neighbor to TEDI satisfies the in-circle condition for the new node and the vertices of the common edge between new node and the neighbor node. If so, that edge is removed from TEDI, and the neighbor node and the two corresponding edges are added to the TEDI.

É When TEDI does not change any more, delete all triangles previously inside TEDI, and connect the new node to all vertices of TEDI.

Goodness Criteria Each new triangle is marked bad, except if the ratio of its smallest to its largest edge is greater than a threshold value b. In this report b ¿ 0.7. If a triangle is marked bad, and in the first trial (firts addition of a new point inside it), it is not deleted, it will be accepted as it is, and will be marked ""good"".

Spacing Function Using spacing functions is standard in grid generation schemes. We define the value of the spacing function for boundary points as the average distance from the neighboring nodes. In each triangle, we assume a linear interpolation of the value of the spacing function on the vertices to find its values inside that triangle. Therefore, whenever a new node is introduced, the value of the spacing function at that point could be easily calculated. If h_0 is the value of the spacing function at a vertex of a triangle, and s_i are outward normals to the edges, the value of the spacing function at the new node ""a"" is equal to

$$h_a = h_0 + \frac{-1}{2s} \overset{3}{\underset{i=1}{õ}} h_i s_i \qquad (1)$$

To add a new node on each small edge of a bad triangle, we assume that the node produces a new good-looking equilateral triangle. One can show that this proposes the distance between the new node and the previous nodes equal to hÅ / [1-(¡3/4)(êh.n)] where, h is the spacing function, hÅ is the average value of the spacing function in nodes on the small edge, and n is the normal to the mentioned small edge. This equation may predict bad values for the spacing function in regions of high gradient. To ensure acceptable values, we impose restriction b< 1/hÅ< 1/b where b is experimentally determined to be about 0.5.

FINITE ELEMENT FORMULATION

To solve the linear equation $ê^2T=0$ on a given Dirichlete or Neumann boundary conditions, it is enough to find T which minimizes the functional

$$t(T) = ßT.ê^2TdW \qquad (2)$$

where W is the whole domain, which is a surface in two dimensional problems. this functional will allow us to be less sensitive in our solution to differentiability of the function T. In this way, it will be easier to work with singular solutions. One can show that with Dirichlete or Neumann boundary condition, this functional has the form

$$t(T) = ß||êT||^2dW \qquad (3)$$

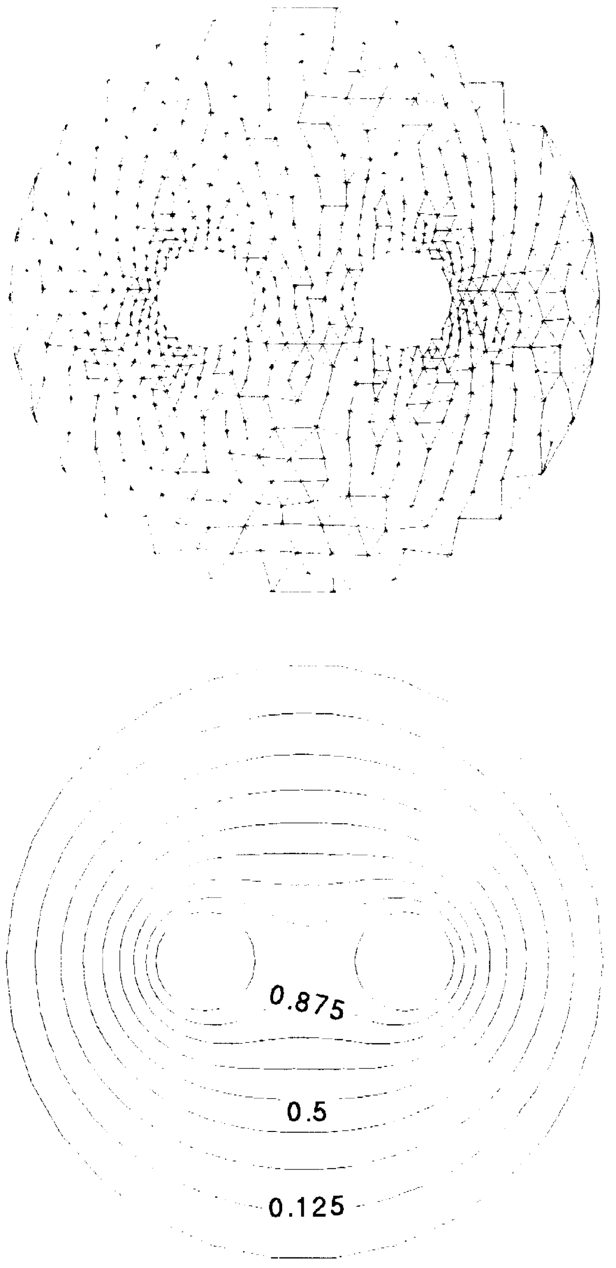
Using the procedure explained in section 2, we discretize the whole domain to N triangular cells. This triangulation for a circular domain with two cavities is shown in Figure 1. We use a cell-vertex method, i.e. we assume the value of T on all cell vertices (except probably those on the boundary) to be unknown. Let's assume linear distribution for function T on each cell, i.e. T on cell i is defined as

$$T_i(x,y)=a_i + b_ix + c_iy \qquad (4)$$

which satisfies values of T_j on all vertices of triangle i. Finally, the value of T on the whole domain is found to be

$$T(x,y) = \overset{N}{\underset{i=1}{õ}} T_i(x,y) \qquad (5)$$

where N is the number of cells.



(b)

Figure 1. Solution of the Laplace equation on an circular region with two holes a) Grid made by frontal approach after adaptation with e=0.2. b) Solution contours.

One can see that the following matrix

$$\begin{bmatrix} a_i \\ b_i \\ c_i \end{bmatrix} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix}^{-1} \begin{bmatrix} T_{i1} \\ T_{i2} \\ T_{i3} \end{bmatrix} \qquad (6)$$

gives the coefficient values for each cell i. Here, indices 1, 2 and 3 are corresponding to three vertices of triangle i. One can show that minimization of the functional introduced in Equation 1 reduces to the equation

A.T=B where

$$A=[C_{ff}] \qquad T=[T_f] \qquad B=-[C_{fp}][T_p] \qquad (7)$$

Here, T_f is the vector of inner vertices temperature, and T_p the vector of boundary vertices temperature. Matrix C is determined so that:

$$t=[T_f T_p] \begin{bmatrix} C_{ff} & C_{fp} \\ C_{pf} & C_{pp} \end{bmatrix} \begin{bmatrix} T_f \\ T_p \end{bmatrix} \qquad (8)$$

and components of matrix C could be geometrically determined, and is a function of the shape functions [16, 17]. To find elements of matrices C_{ff} or C_{fp} for each vertex, we should find contribution of all cells sharing in that vertex. Matrix C is symmetric and sparse. Also, summation of each row and column of this matrix is equal to zero.

Solution of Equation 2 will give the value of T at each vertex. One way is to find the inverse of the sparse matrix A. Another method, which is more popular, is the family of iterative methods. Here, we start with an arbitrary initial condition. To have an accurate initial guess, we use a scheme similar to the algorithm used for finding the values of the space function. The iterative procedure recalculates the value of the temperature at each vertex by

$$T_j = \frac{-1}{C_{jj}} \sum_{i=1, i \neq j}^{n} T_i C_{ji} \qquad (9)$$

Since for all nodes not connected to node j the value of c_{ji} is zero, only a few multiplications are necessary. We save all necessary information related to matrix C in a very efficient way in a

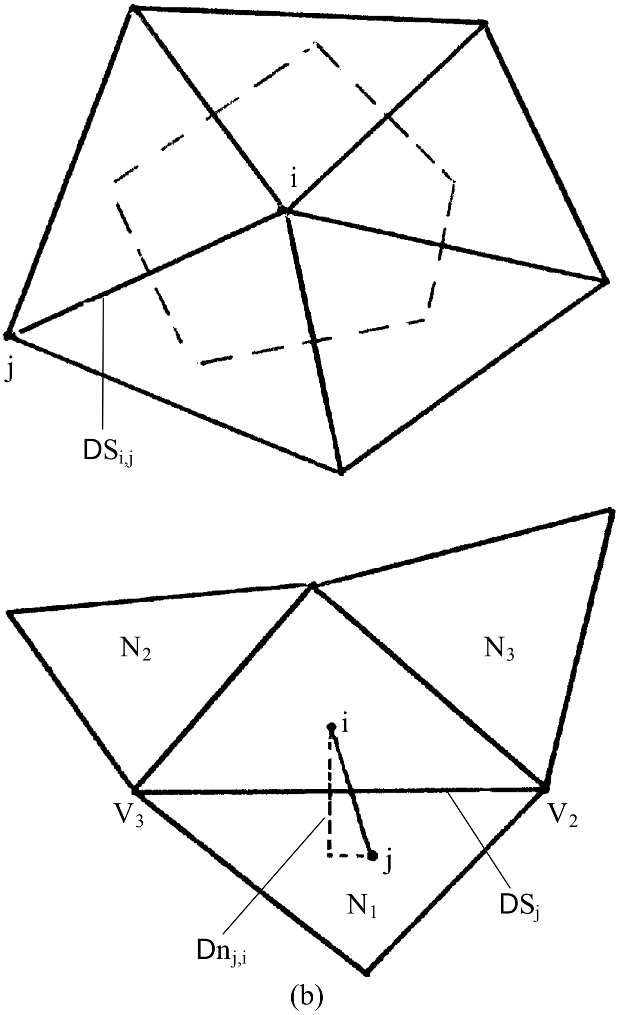


Figure 2. Stencil of computation in (a) a cell-vertex method, (b) a cell-center method.

matrix with space complexity of O(N), and iterate Equation 3 till convergence is acheived (i.e. norm of T alteration is small enough). Since C is a positive definite matrix, this method is always convergent [18]. This procedure is in fact an averaging procedure for the temperature of each vertex.

Overrelaxation is also used to increase the convergence rate. We use

$$T_j^{k+1}=(1-w)T_j^K - \frac{w}{c_{ji}} \sum_{i=1,i \neq j}^{n} T_i c_{ji} \tag{10}$$

using values of $1< w<2$, rate of convergence will improve.

FINITE VOLUME FORMULATION

To solve the conservation equations on an unstructured grid by finite volume, we may use cell-vertex or cell-center methods. In the first method, ew construct our control volumes by connecting the Voronoi vertices around each node (Figure 2a). Integration of the Laplace equation on this element results in

$$\sum_{j=1}^{n} \frac{T_i-T_j}{DS_{i,j}} DS_v = 0 \tag{11}$$

Here, i is the node in the center of volume element, and j denotes different vertices of this element. T is the temperature, $DS_{i,j}$ is the distance between nodes and j, DS_v is the length of the corresponding Voronoi polygon edge and n is the number of the edges of the polygon.

In cell-center methods, the unknown is the temperature at center of our triangular elements. In a first order estimation, one assumes that the temperature is constant in each cell, and Delaunay triangles construct our volume elements (Figure 2b). Integration of the Laplace equation over these elements results in:

$$\sum_{j=1}^{3} \frac{T_i-T_j}{Dn_{j,i}} DS_j = 0 \tag{12}$$

where $Dn_{j,i}$ is the distance of centroids of two neighboring cells with centers i and j in the direction normal to the common edge, DS_j is the length of that edge.

In this work we have used the first formulation. Application of this equation on each cell produces a linear equation. Writing the equations corresponding to all triangular cells will result in a linear system of equations which could be solved with standard methods.

ADAPTATION

To uniformly distribute truncation errors, cell sizes should be made proportional to the

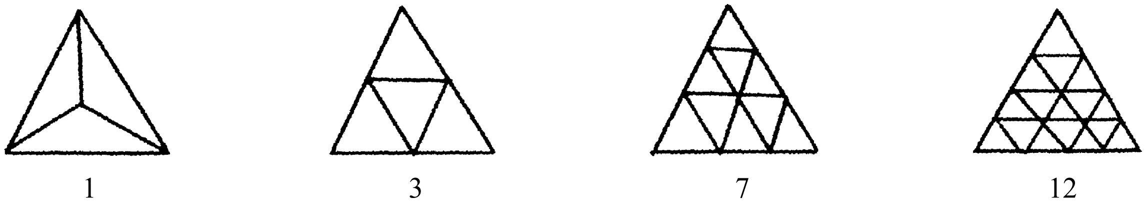


Figure 3. Formats used for node addiction.

inverse of the gradient of the solution at each cell. To do so, using a frontal method, we produce an initial grid and solve the Laplace equation over it to find an initial numerical solution. Since most cells are coarse at this stage, this solution is not accurate enough in points with high gradients.

To find cells with highest relative truncation error, we need a criterion for which we use dimensionless parameter $T^{*} = \frac{\text{¡}s|\bar{\hat{e}}T|}{T_0}$. Here, S is the area of the corresponding cell, and T_0 is a typical (maximum) value of the temperature on the boundaries. Regions with high values of T^{*} should be refined. We have four predetermined formats for node addition (Figure 3). According to the value of T^{*}, one of these formats is selected. Here we use k_1= 1.5, K_2=s, k_3=3, and these values correspond to cell sizes in different levels, e determines how small truncation error we wish. In regions of indefinite solution gradients (singular points) we restrict adaptation levels, or the smallest cell sizes to a predetermined value.

After addition of these new grid points, the retriangulation procedure is repeated, and a new Delaunay triangulation of the field is constructed. Since our node addition does not occur continuously, ew use a postprocessing procedure to smooth the grid distribution. One way of doing this is to assume springs between each two nodes, with stiffness coefficients as a function of their distance, and even the simplest function, i.e. a constant, works very well. We used this simple function, and repeated this procedure two or three times A better control is achieved if we make stiffness coefficients a function of the spacing functions. Then, for each node, we will locally find its equilibrium position (under springs forces) in respect to its neighbor nodes. Afther this process, the Delaunay property may deteriorate, and needs to be locally checked. A more efficient procedure is to use locally this filter after each node addition. The filtering postprocedure shows to be very effective (see results).

TABLE 1. Criteria Used for Node Addition.

new nodes	value of T^{*}
0	$T^{*} <$ û
1	û<T^{*}<k_1û
3	k_1û<T^{*}<k_2û
7	k_2û<T^{*}<k_3û
12	k_3û<T^{*}

RESULTS

Two examples are solved by each method to show features of the method. In the first example for FEM, we solve the Laplace equation on a circular region with two holes on it. The boundary condition is Dirichlete. The temperature on the outer boundary is equal to one, and on internal faces is equal to zero. Using an arbitrary initial set of grid points on the boundaries, we use a frontal approach to make the initial grid. Then we use the adaptation procedure to find the final mesh (Figure 1a) with the final solution shown in Figure 1b. Here e is equal to 0.2. As one sees,

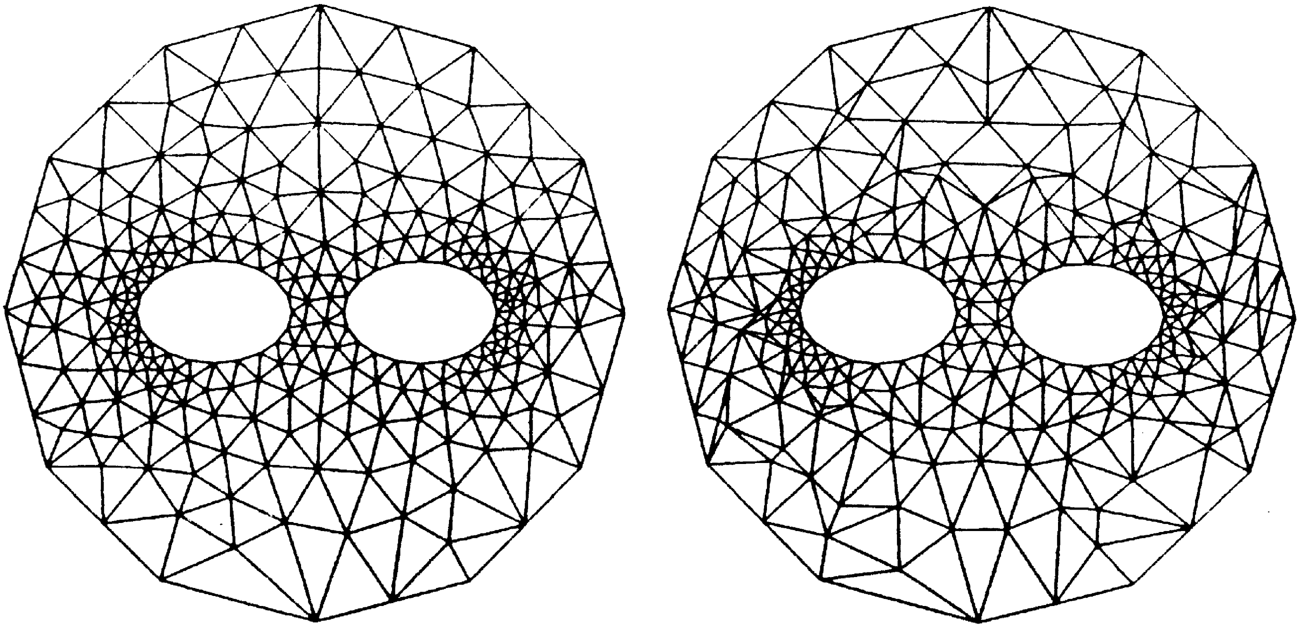


Figure 4. Solution of the Laplace equation on the same region as Figure 1 (a) adapted grid with e=0.1 and (b) grid after filtering postprocess.

cell sizes vary smoothly and rapidly, so some of them are as large as the internal holes.

Figure 4a is a circular domain with two elliptical holes, which is first triangulated with the above mentioned method, but then we have used a Laplace filter to make the grid as smooth as possible. Figure 4b shows the effect of the filtering.

Next, we have solved the Laplace equation on a square region with different constant temperatures on lower and the other three sides of the outer boundary by our FEM solver. The length of each side of the square is six. This example is solved to show both accuracy and effectiveness of the algorithm. Typical solution contours are shown in Figure 5(d). The exact solution of this problem is

$$T(x,y) = \frac{4}{p} \underset{odd\ n}{\tilde{o}} \frac{\sin\,[\frac{np}{a}\ (x+a/2)].\ \ \sinh[\frac{np}{a}\ (a/2-y)]}{n\ \sinh\ np} \qquad (13)$$

First, to show the solution and the effect of the adaptation, we have solved the Laplace equation on that square region on two different meshes. The first mesh is an almost uniform grid with 270 cells. The solution is quite accurate everywhere except the small region close to the lower corners (singular points). The solution contours only near these corners are shown in Figure (6d). To resolve a better solution in these regions. We use the adaptation procedure for several times to get the grid of 2500 cells shown in Figure (6a). The solution contours are now much more accurate even in the regions colse to the singular point (Figure (6c)). This has been verified using the exact solution, Equation 8.

To observe the effectiveness of the algorithm, first we compare results on two uniform and adapted cases with same number of cells (i.e. 970 cells). Results are shown in Figures 5 and 7. As Figure 5 shows, the global results are very much the same. The small glitches on the temperature contours of the adapted case are mostly graphical, since the grid size is much higher in the upper region. The difference is in fact in the lower corners (Figure

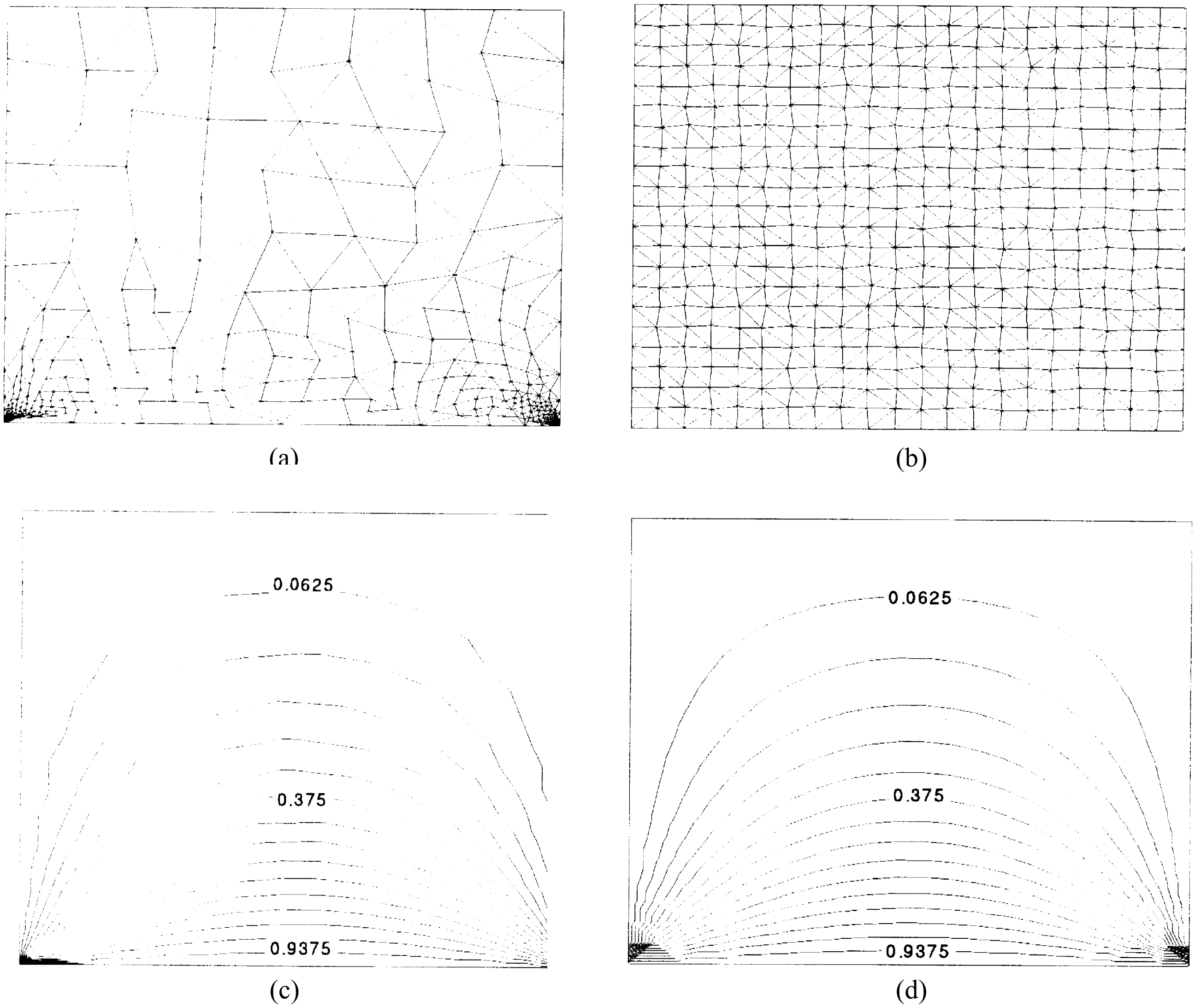


Figure 5. Comparison of solutions on uniform and adapted meshes: (a) adapted grid, (b) uniform grid, (c) solution on the adapted mesh and (d) solution contours on the uniform mesh.

7). This region is almost one hundredth of the total computational field. In this region, the adapted mesh produces a result very colse to the exact solution (Equation 8), but the uniform grid fails to resolve in an acceptable manner.

Finally to assure the accuracy of the solution, using L_2 norm of error (difference of numerical and exact solutions)is computed, and is drawn versus cell sizes for a uniform mesh (Figure 8). This figure shows that the solution has a second order accuracy (Note that the slope of the curve is almost two in the logarithmic scale).

Similar cases are solved by FVM solver to show capabilities of the adaptive algorithm. The first case is the solution of the heat equation in a circular domain with two circular holes and with continuous Dirichlete boundary conditions. Again, temperatures on the outer and inner faces are equal to zero and one, respectively. Results are very much close to those presented in Figure 1, not shown here to prevent repeated similar figures. Since the boundary conditions

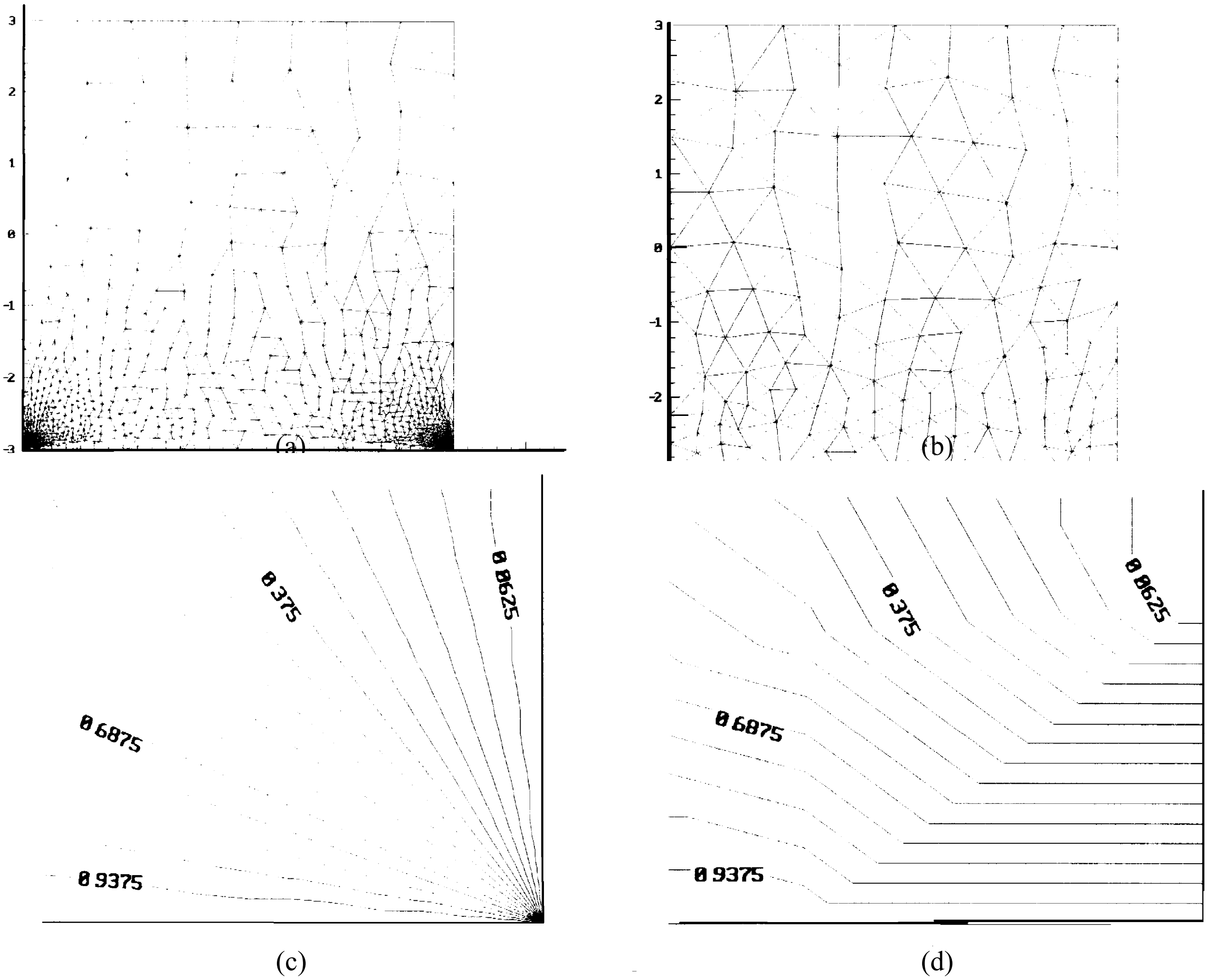


Figure 6. Solution of Laplace equation on a square with constant zero temperature on upper and side boundaries and temperature equal to one on lower side: (a) adapted grid, (b) initial grid, (c) solution on the lower corner in the adapted case and (d) solution contours on the lower corner in the initial mesh.

are continuous, this solution is found to be satisfactory. Many sweeps of adaptation were applied to that solution, and the final grid and solution did not differ in this simple case.

Case two for FVM, is a circular domain, but with temperature equal to 1 on the upper part of the boundary, and equal to zero on the lower part. The grid is several times adapted to the new solution and afterwards it is postprocessed (Figure 9). One can see that how effective the adaptation procedure is in this more difficult case. The solution contours are not reported here.

CONCLUSIONS

An adaptive grid generation scheme was used in a finite element and volume formulation of numerical solution of the Laplace equation. Adaptive grids were used to resolve the best possible solution using a finite number of grid cells. The grid generation scheme starts with a frontal approach to generate mesh points, and

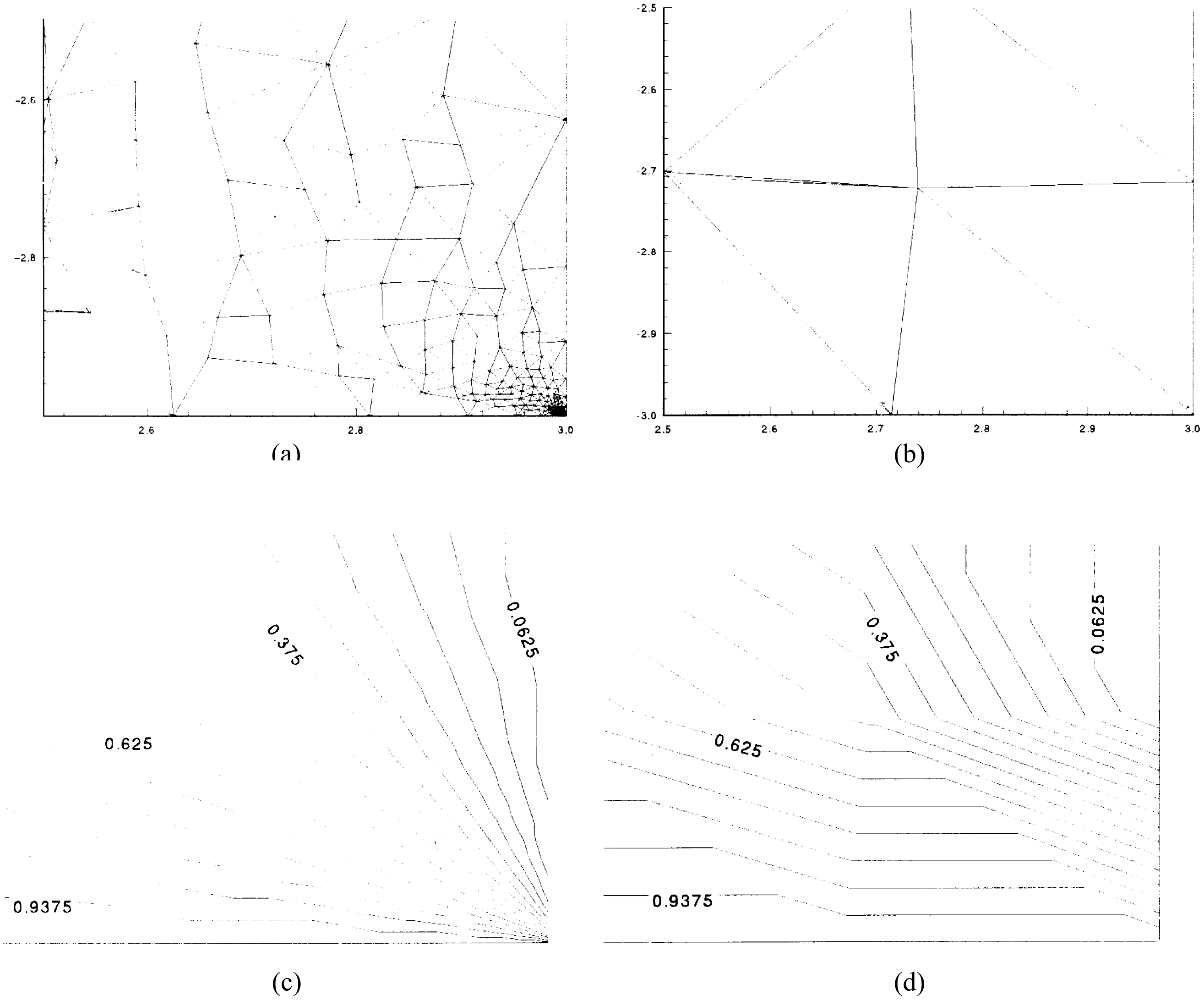


Figure 7. Comparison of solutions on uniform and adapted meshes: (a) lower corner of adapted grid, (b) lower corner of uniform grid, (c) solution on the lower corner in the adapted case, and (d) solution contours on the lower corner in the uniform mesh.

to Delaunay triangulate them. Appropriate merge, delete, and postprocessing procedures were applied to improve the grid quality. After using standard finite element or volume solution of the problem, a criterion is used to find regions with high relative truncation errors. These regions are refined, and after retriangulation, and using filtering process, computation is repeated. The Dirichlete boundary conditions are applied by determining the nodal values on the boundary. The whole process is repeated a few times, until required accuracy, or allowed number of levels of refining is achieved. The method was applied to a few test cases and showed to be convergent, and resulting in an accurate solution with finite number of grid cells, even in the case of singular boundary conditions. This work is going to be extended to other physical equations and three dimensions.

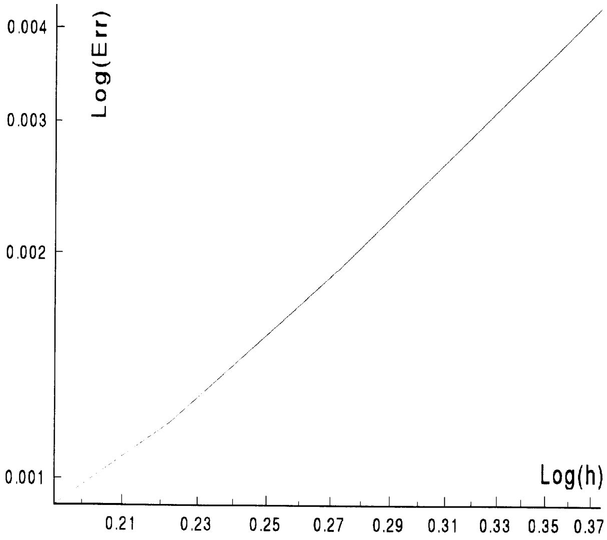


Figure 8. L_2 norm of error versus cell sizes for the uniform mesh computations.

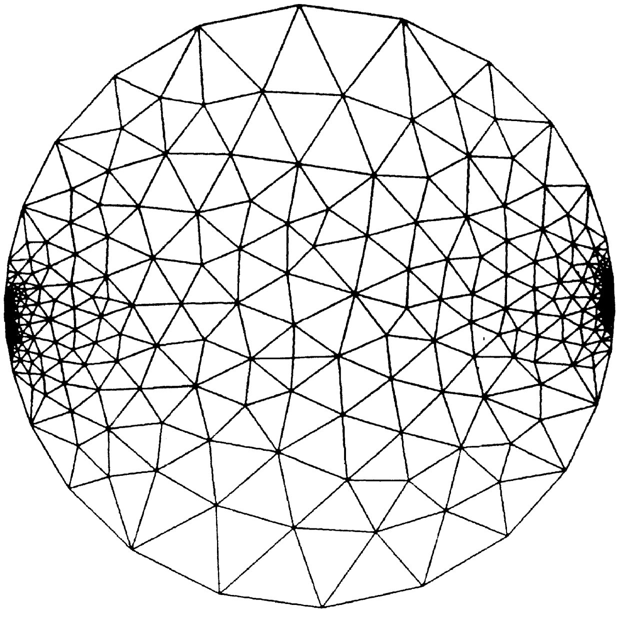


Figure 9. Grid on a circular domain with singular boundary condition/

REFERENCES

1. Lo, S., "Volume discretization into tetrahedra-I", **Computers and Structures**, Vol. 39, (1991), 493-500.
2. Tipper, J., "Fortran Programs to Construct the Planar Voronoi Diagram", **Computers and Geosciences**, Vol. 17, (1991), 597-632.
3. Bowyer, A., "Computing Dirichlet Tesselation", **The Computer Journal**, Vol. 24, No. 2, (1981), 162-166.
4. Lawson, C., "Software for C^1 Surface Interpolation", in **Mathematical Software III** (J. Rice, ed.), Academic Press, (1997).
5. Watson, D., "Computing the n-Dimensional Delaunay Tesselation with Applications to Voronoi Polytopes", **The Computer Journal**, Vol., 24, No.l2, (1981), 167-172.
6. Rausch, R. Batina, J. and Yang, H., "Spatial Adaptation of Unstructured Meshes for Unsteady Aerodynamic Flow Computations", **AIAA Journal**, Vol. 30, (1992), 1243-1250.
7. Struijs, R. Vankeirsblick, and Deconinck, H., "An Adaptive Grid Polygonal Finite Volume Method for the Compressible Flow Equations", **AIAA-89-1959-CP**, (1989).
8. Mavriplis, D. and Jameson, A., "Multigrid Solution of the Two Dimensional Euler Equations on Unstructured Triangular Meshes", **AIAA-87-0353**, (1987).
9. Jameson, A., "The Evolution of Computational Methods in Aerodynamics", **Journal of Applied Mechanics**, Vol. 50, (1983), 1052-70.
10. Barth, T. J., "Analysis of Implicit Local Linearization Techniques for Upwind and TVD Algorithms", in **AIAA Paper 87-0595**, (January 1987).
11. Muller, J. Roe, P. L. and Deconink, H., "A Frontal Approach for Internal Node Generation in Delaunay Triangulations", **International Journal for Numerical Methods in Fluids**, (1993).
12. Preparata, F. and Shamos, M., "Computational Geometry", Springer-Verlag, (1985).
13. Barth, "On Unstructured Grids and Solvers", **VKI Lecture Series 1990-03**, (1990).
14. Baker, T., "Element Quality in Tetrahedral Meshes", in **7th International Conference on Finite Element Methods in Flow Problems**, (1989).
15. Rippa, S., "Minimal Roughness Property of the Delaunay Triangulation", PhD Thesis, Tel-Aviv University, (1989).
16. Sadiku, M., "Numerical Techniques in Electromagnetics", CRC Press, (1992).
17. Hughes, T., "The Finite Element Method", Prentice Hall, (1987).
18. Press, W. Flannery, B. Teukolsky, S. and Vetterling, W., "Numerical Recipes", Cambridge University Press, (1992).