An Unstructured Grid Generation Approach for Inviscid Flow Solutions

BAHRAINIAN SEYED SAIED, DANEH DEZFULI ALIREZA
Mechanical Engineering Department
Shahid Chamran University
AHWAZ, IRAN
bahrainian@scu.ac.ir

Abstract: -The unsteady and inviscid compressible flow around a circular cylinder and airfoil has been solved using a cell-centered finite volume method. A new algorithm for the generation of unstructured triangular grids was used to discretize the computational flow domain. Automatic grid generation, cell size distribution, and geometry treatment are some of the grid algorithm capabilities. Euler flow solutions have been obtained for subsonic and transonic flow conditions. The subsonic solution of flow around a circular cylinder grid verified the unstructured grid techniques, and was compared with an analytical solution. Transonic solution over the same grid fully converged after only 2791 iterations. The grid quality resulted in smooth pressure contours. Transonic flow solutions were also obtained for airfoil at free stream Mach number of 0.8 and attack angle of 1.25. The true position of the shock is illustrated in the pressure coefficient plots, and shows the shock capturing capability of the solution method. The convergence history obtained, points to the accuracy of the numerical solution.

Key-Words: - Unsteady - Inviscid - Geometry Movement - Unstructured Grid - Edge Swapping.

1 Introduction

The governing equations of the fluid flow are partial differential equations. Analytical solutions of these equations for complex geometry are difficult if not impossible. One approach is to use computational fluid dynamics (CFD) to solve the governing equations. In most CFD analysis, the first step is to generate a computational grid around the domain of interest. Structured and Unstructured grid generation methods are two dominant categories for space descretization of flow domains. Structured grids are generated based on a specific mathematical relation, while unstructured grid generation does not follow a specific equation for allocation of grid points. Structured grids produce equal number of cells around each point, and have been extensively investigated by Thompson [1].

Unstructured grids require a connectivity matrix for the definition of points and point connectivity. This may require a large amount of data storage, and can make the generation of unstructured grids more difficult than the structured grid generation methods. However, the ability of unstructured grids in handling complex geometry and solution adaptation has made them the subject of interest for many researchers. Lohner [2] has used the unstructured advancing front method to generate grids for medical applications. Marcum [3] has generated unstructured grids using iterative point insertion and local reconnection. Hexahedral blocks were employed to generate tetrahedral grids for aerodynamic applications [4].

In this paper, a new unstructured grid generation algorithm is used to discretize the flow domain with triangular cells. This algorithm uses a combination of point insertion and cell subdivision techniques whereby new grid points and point connectivities are created simultaneously. Automatic grid generation, the ability to change the geometry of the solid domain, grid quality enhancement, and cell size distribution are some of the issues addressed in this algorithm. Using this method, sample computational grids are generated around a circular cylinder and airfoil geometry applicable to external flow.

To verify the computational unstructured grids, Euler flow solutions are obtained. Jameson and Mavriplis solved Euler equations numerically on a regular triangular mesh [5]. In the present work, solutions of the unsteady Euler equations by a cell-centered finite volume method and unstructured grids are presented. In order to prevent oscillations near shock waves, extra dissipative terms were added to the Euler equations. A modified Rung-Kutta scheme is used to update the flow variables in

each time step. Far field boundary condition can affect the outgoing disturbances. To minimize the reflection of these disturbances, Riemann invariants are used for the outer boundary condition.

2 Unstructured Grid Generation

A novel algorithm is used to generate triangular unstructured grids. Grid refinement is based on a combination of point insertion and cell-subdivision methods. An edge-based and a cell-based connectivity matrix were employed simultaneously to prevent long searches through the list of edges. The algorithm starts with a very coarse starting grid. Surface and field grids are then generated simultaneously. The art of this grid generation algorithm is in its geometry movement capability, where the newly generated surface points are moved to a prescribed geometry. The following sections further describe the grid generation procedure.

2.1 Initial Grid

The grid generation algorithm developed in the current research, starts with a very coarse initial grid. A sample initial grid is shown in Figure 1, where, the domain is arbitrarily triangulated.

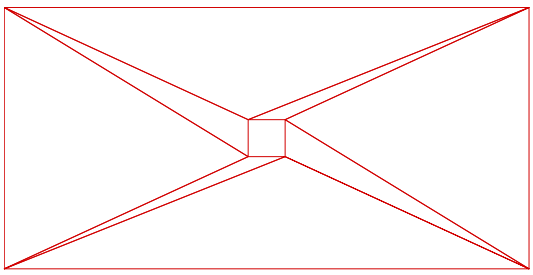

Fig 1: Initial coarse grid

This initial grid contains 8 vertices and 16 edges that form the 8 triangular cells around the square shape solids geometry within a rectangular domain. It should be noted that the 4 edges forming the solid boundary are designated as surface edges.

2.2 Connectivity Matrices

As noted earlier, unstructured grids are associated with a data structure known as the connectivity matrix. The current approach uses an edge based data structure and a cell based data structure simultaneously. This reduces the long searches through the list of cells, edges, or vertices.

In addition to the grid connectivity data, other information is also stored within the matrices. For example, each edge is designated with an integer value to represent a surface, interior or outer boundary edge. This integer value is stored as a column within the edge based data structure.

2.3 Grid Refinement Procedure

Once the data connectivity matrices are constructed for the initial grid, the remainder of the grid generation procedure is performed automatically. The algorithm starts by sorting the cells based on their area with the largest cell placed on top of the stack. The longest edge of the cell on top of the stack is then selected for refinement. A new grid point is defined and inserted at the middle of this edge. This point is then connected to the opposite vertices of the triangular cells sharing the selected edge. Figure 2 shows the initial grid after one cycle of refinement. It can be seen that the largest edges within the original grid are subdivided.

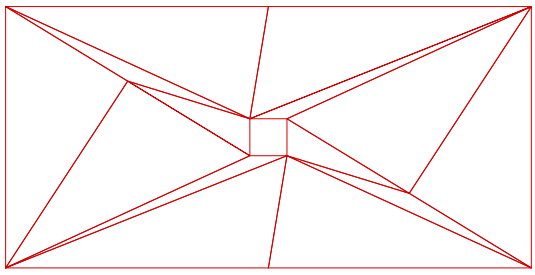

Fig 2: Grid after 1st cycle of refinement

The newly generated vertices, edges, and cells are recorded by updating the connectivity matrices. The program is then ready to further refine the initial grid. To illustrate this procedure, Figures 3 and 4 show the grid after 2nd and 3rd cycles of refinement.

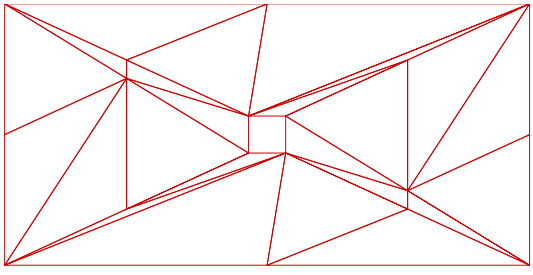

Fig 3: Grid after 2nd cycle of refinement

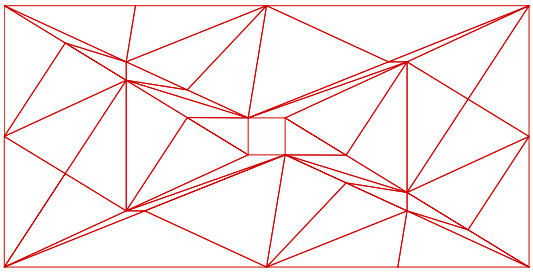

Fig 4: Grid after 3rd cycle of refinement

The grid displayed in Figure 4, although numerically acceptable, contains many elongated triangular cells. These elongated cells can deteriorate Euler solutions which require isotropic grids. To improve the shape of elongated triangular cells additional algorithms for grid quality must be employed.

2.4 Grid Quality

Two grid quality algorithms were used to improve the overall shape of the triangular cells. The first algorithm involved an edge swapping procedure. This procedure is better described by Figure 5, where edge AB is removed and swapped for edge CD. By doing so, the resulting triangles ACD and BDC are closer in shape to an equilateral triangle.

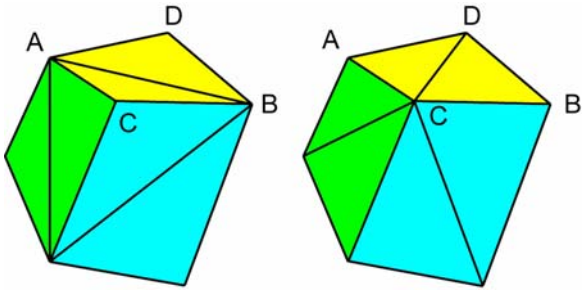

Fig 5: Edge swapping procedure, where edge AB is removed and edge CD is generated

The second algorithm used for improving the grid quality involved a grid point smoothing procedure. This procedure is better described by Figure 6, where point B is moved to a new location for optimal quality of the grids surrounding point B. The smoothing weighted coefficient can be adjusted to satisfy a variety of criteria. Equal smoothing coefficient is used for all of the grids illustrated here.

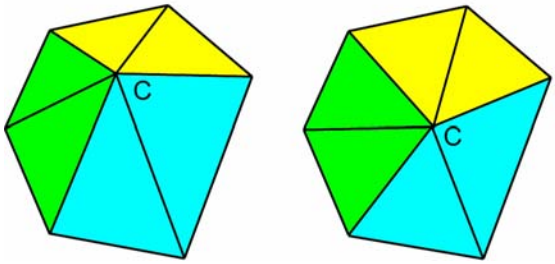

Fig 6: Grid smoothing procedure, where point C is moved to a new location for the optimal grid quality

By applying the grid quality algorithms while refining the initial grid, triangular cells are clearly improved. This can be seen in Figure 7, where final grid outputs are compared with and without quality improvement.

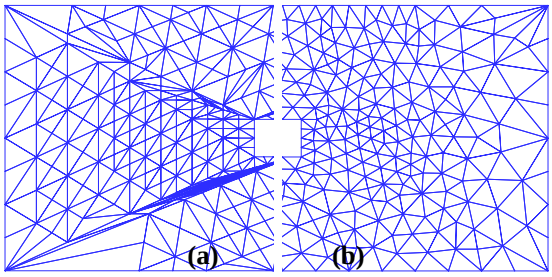

Fig 7: Final grid output (a) without quality improvement, (b) with quality improvement

The amount of improvement with respect to an equilateral triangle depends on the user defined quality parameters. The elongated cells of Figure 7 (a) have vanished after employment of quality improvement algorithms. The final grids displayed in Figure 7 (b) dose not have any elongated triangular cells. But, how close are they when compared to an equilateral triangle? Figure 8 shows the distribution of angles between adjacent triangle edges for the final grid output of Figure 7 before and after quality improvement. It is shown that the distributions of skewed angles are removed after employing grid quality algorithms.

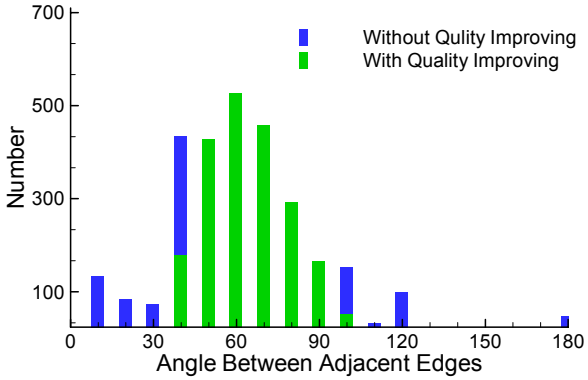

Fig 8: Distribution of angles between adjacent triangle edges for the final grid output before and after quality improvement

2.4 Geometry Treatment

The last and the most important function of the triangular unstructured grid generation procedure is its capability to adapt a coarse initial geometry to a prescribed solid geometry. To show this procedure the initial coarse grid of Figure 1 is refined again with utilizing the geometry treatment algorithm. In this case, the objective is to match the initial square shape solid geometry of Figure 1 to that of a circular cylinder. By doing so and simultaneously employing grid quality algorithms, final grid around a circular cylinder is generated and is shown in Figure 9.

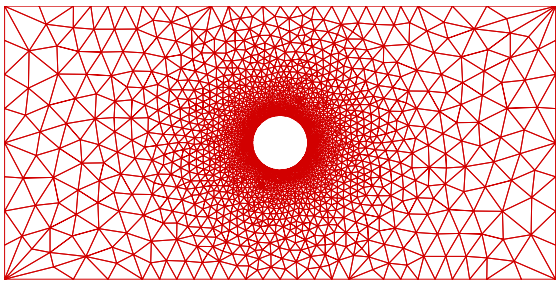

Figure 9: Final computational grid output around a circular cylinder within a rectangular domain

To examine the quality of the final triangular cells near the solid geometry, a close view of the final grid output of Figure 9 is shown in Figure 10.

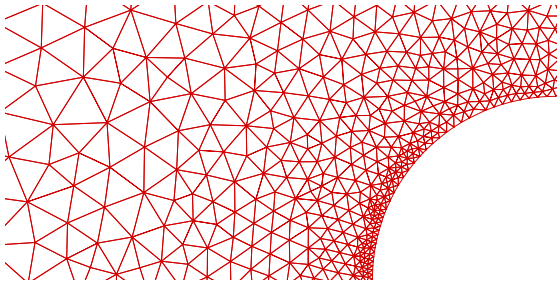

Figure 10: A close view of the final computational grid shown in Figure 9

To verify the grid generation procedure illustrated in this section, Euler flow solutions are obtained over the computational grid of Figure 9. The mathematical formulations for the flow solver are given in the following section.

3 Mathematical Formulation

The finite volume representation of the governing flow equations, a multi-step time marching scheme, initial conditions, and the appropriate boundary conditions constitute the numerical mathematical formulation. The governing equations used are the two-dimensional Euler equations. The Euler equations govern inviscid, compressible fluid motion and can be written in integral form for a region Ω with boundary $\partial\Omega$ as:

$$\frac{\partial}{\partial t}\iint_{\Omega}\vec{w}dxdy+\oint_{\partial\Omega}\left(\vec{f}dy-\vec{g}dx\right)=0\cdot \tag{1}$$

Where t is the time variable, x and y are Cartesian coordinates, and vectors $\vec{w}$, $\vec{f}$, and $\vec{g}$ are defined by the following matrices.

$$\vec{w}=\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ E\end{pmatrix},\ \vec{f}=\begin{pmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ u(E+p)\end{pmatrix},\ \vec{g}=\begin{pmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ v(E+p)\end{pmatrix}\cdot \tag{2}$$

For a perfect gas, the total energy E is given by:

$$E=\frac{p}{(\gamma-1)}+\frac{1}{2}\rho\left(u^2+v^2\right)\cdot \tag{3}$$

Where; ρ, u, v, p, and γ are definitions of density, Cartesian velocity components, static pressure, and ratio of specific heats respectively.

Equation (1) is in vector form and includes the set of continuity, momentum and energy equations. This set of equations, apply to each cell individually and can also be written as:

$$A_i \frac{d\vec{w}_i}{dt} + \vec{Q}_i = 0 \cdot \quad (4)$$

Where, A_i is the cell area and $\vec{Q}_i$ is the net flux out of the cell. The net flux can be computed by taking the sum over the three sides of the triangular cell and is given by:

$$\vec{Q}_i = \sum_{k=1}^{3} \left(\vec{f}_k \Delta y_k - \vec{g}_k \Delta x_k \right) \cdot \quad (5)$$

Where, Δx and Δy are spatial increment in the x and y directions.

An adaptive scheme for adding dissipative terms to the Euler equations has proven to be effective in practice in numerous calculations of complex unsteady flows. Thus, to consider the dissipative terms, Equation (4) is replaced by:

$$A_i \frac{d\vec{w}_i}{dt} + \vec{Q}_i - \vec{D}_i = 0 \cdot \quad (6)$$

Where $\vec{D}_i$ contains the dissipative values for each cell and is calculated as shown below.

$$\vec{D}_i = \sum_{j=1}^{3} R_{ij} [\varepsilon_{ij}^{(2)} (\vec{w}_i - \vec{w}_j) + \varepsilon_{ij}^{(4)} (\nabla^2 \vec{w}_i - \nabla^2 \vec{w}_j)] \cdot \quad (7)$$

with

$$R_{ij} = \frac{1}{2} \left(\frac{A_i}{\Delta t_i} + \frac{A_j}{\Delta t_j} \right) \cdot \quad (8)$$

A pressure sensor ν_i, first defined by Jameson [6], is utilized to control the amount of dissipation to be added throughout the flow field. This pressure sensor for a triangular cell is given by:

$$\nu_i = \sum_{j=1}^{3} \left| \frac{p_i - p_j}{p_i + p_j} \right| \cdot \quad (9)$$

where, p_i and p_j are pressures of adjacent cells. The calculated value of the pressure sensor is then used to predict the amount of pressure switches $\varepsilon_i^{(2)}$ and $\varepsilon_i^{(4)}$. These first- and third order pressure switches are calculated as a function of the pressure sensor ν_i, and the adjustable dissipation coefficients k_2 and k_4 as follows.

$$\varepsilon_{ij}^{(2)} = k_2 \nu \cdot \quad (10)$$

$$\varepsilon_{ij}^{(4)} = \max(0.0, k_4 - \varepsilon_{ij}^{(2)}) \cdot \quad (11)$$

A modified four step Rung-Kutta scheme is used to update flow variables after each time step. The updating procedure is performed in the following manner.

$$\begin{aligned} \vec{w}^{(0)} &= \vec{w}^n \\ \vec{w}^{(l)} &= \vec{w}^{(0)} - \alpha_l \frac{\Delta t}{A} \left[\vec{Q}^{(l-1)} - \vec{D}^{(l-1)} \right] \cdot \\ \vec{w}^{n+1} &= \vec{w}^{(4)} \end{aligned} \quad (12)$$

Where l represents arbitrary level index for each stage, α is coefficient of Rung-Kutta, and Δt is the time step.

At the inner boundary, there is no flux through the solid wall. Therefore, the solid wall flux is set to zero ($Q_k = 0$). The far field boundary condition of the computational domain is constructed using the Riemann invariants [6].

4 Computational Flow Solutions

Three computational flow solutions are given in this section to both verify the unstructured grid method's robustness, and to show the accuracy of the Euler flow solver. First and second cases include flow over a circular cylinder at free stream Mach numbers of 0.3 and 0.45. The third case illustrates the transonic flow over an airfoil at a free stream Mach number of 0.8.

4.1 Circular cylinder, $M_\infty = 0.25$

The first computational case involved the inviscid flow solution around circular cylinder at $M_\infty = 0.25$. The computational grid output around a circular cylinder given in Figure 9 was used with the Euler flow solver for 6000 iterations. Average density residuals are plotted verses iteration numbers and are shown in Figure 11. The distribution of pressure coefficients on the cylinder surface are plotted against an inviscid analytical solution. This is shown in Figure 12, where, exceptional agreement with

analytical solution is achieved for this subsonic test run.

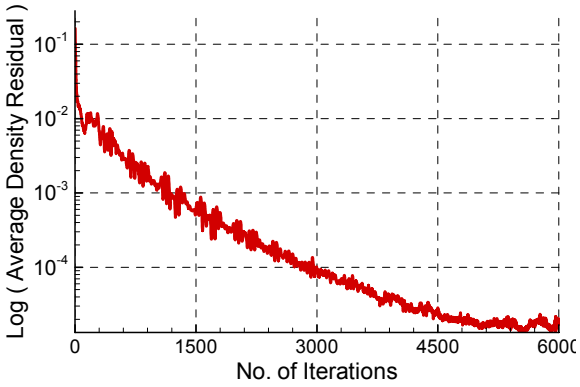

Fig 11: Convergence history for the solution of flow around circular cylinder at $M_\infty = 0.25$

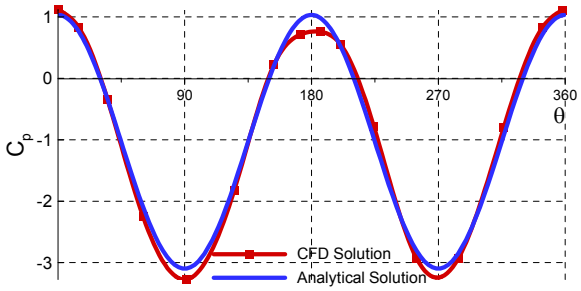

Fig 12: Distribution of pressure coefficients on a circular cylinder surface at $M_\infty = 0.25$

The Mach number and pressure contours for the solution around circular cylinder at $M_\infty = 0.25$ are plotted over the cylinder surface and are shown in Figures 13 and 14.

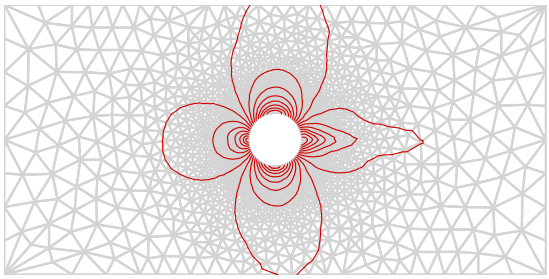

Fig 13: Mach number contours for the solution of flow around circular cylinder at $M_\infty = 0.25$

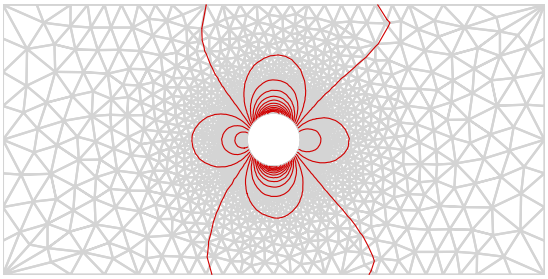

Fig 14: Pressure contours for the solution of flow around circular cylinder at $M_\infty = 0.25$

4.2 Circular cylinder, $M_\infty = 0.45$

The second case uses the final grid of the previous case with a free stream Mach number of 0.45. This was selected to enable the authors to compare the results with available published data. Again, the Euler flow solver was employed for 6000 iterations and the convergence history is shown in Figure 15.

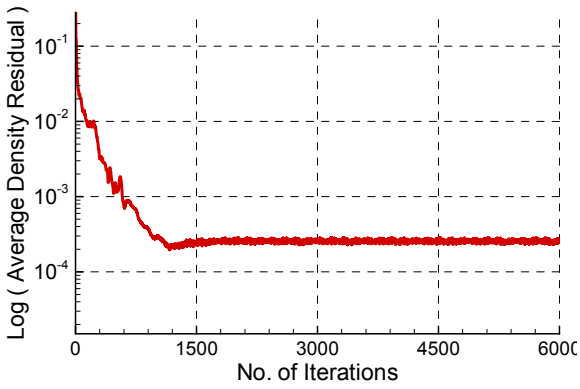

Fig 15: Convergence history for the solution of flow around circular cylinder at $M_\infty = 0.45$

The distribution of pressure coefficients on the cylinder surface are plotted against Jameson [6] and are shown in Figure 16. It can be seen that the CFD solution results are comparable with the published data. The increase in the free stream Mach number from 0.25 of the solution case 1 to 0.45 of this solution have led to development of shocks on the cylinder shoulders. This can be seen by comparing the pressure coefficient curves of Figures 12 and 16 at locations about 90º and 270º from the leading edge on the cylinder surface.

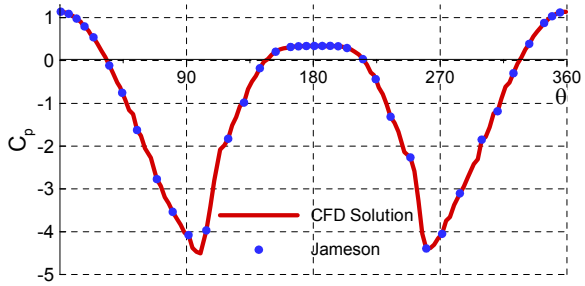

Fig 16: Distribution of pressure coefficients on a circular cylinder surface at $M_\infty = 0.45$

Similarly, Mach number and pressure contours for this solution around the circular cylinder at $M_\infty = 0.45$ are provided by Figures 17 and 18.

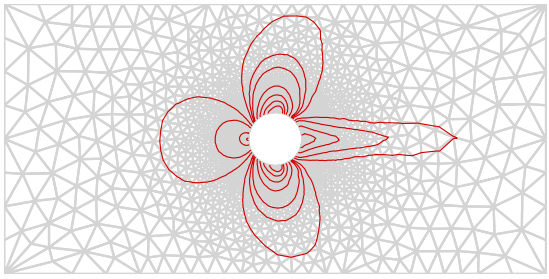

Fig 17: Mach number contours for the solution of flow around circular cylinder at $M_\infty = 0.45$

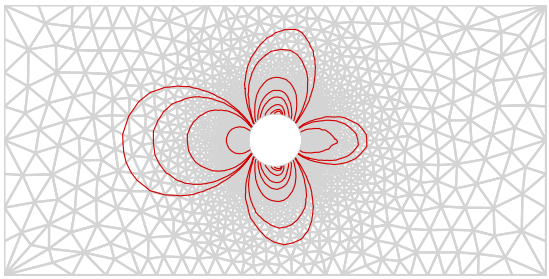

Fig 18: Pressure contours for the solution of flow around circular cylinder at $M_\infty = 0.45$

Distribution of pressure coefficients on a circular cylinder surface for this case was given by Figure 16. To show the location of pressure coefficients with respect to the cylinder surface geometry, pressure coefficients are compared with Jameson [6] results and are shown in Figure 19.

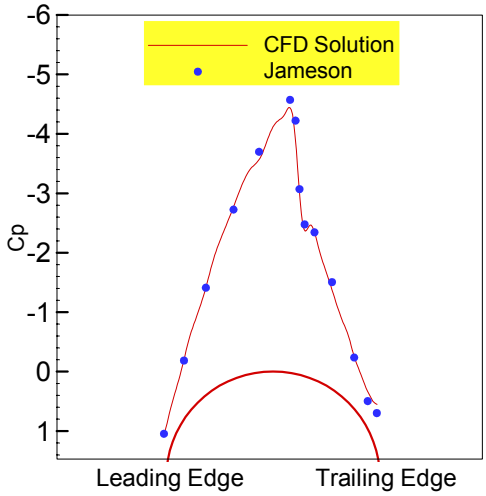

Fig 19: Distribution of pressure coefficients on a circular cylinder surface geometry at $M_\infty = 0.45$

4.3 NACA0012 Airfoil, $M_\infty = 0.8$

The third computational case considers transonic flow on an airfoil with NACA0012 cross section. To do this, the vertices of the initial coarse grid of Figure 1 are modified to suite the problem at hand, and are shown in Figure 20.

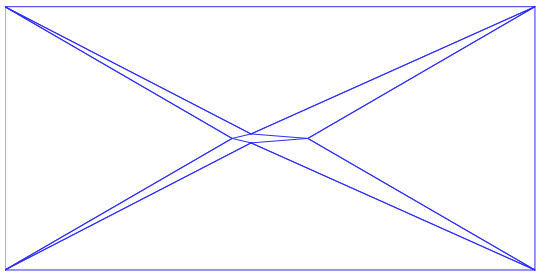

Fig 20: Initial coarse grid for NACA0012

This initial grid is refined and the geometry is adapted to an airfoil with NACA0012 cross section. The final grid output as a result of the refinement procedures is shown in Figure 21.

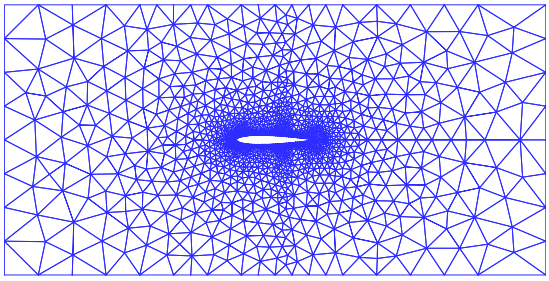

Fig 21: Final computational grid output around a NACA0012 within a rectangular domain

To further examine the grid cells near the solid boundary, close views of the final computational grid at the airfoil's leading and trailing edges are shown in Figure 22.

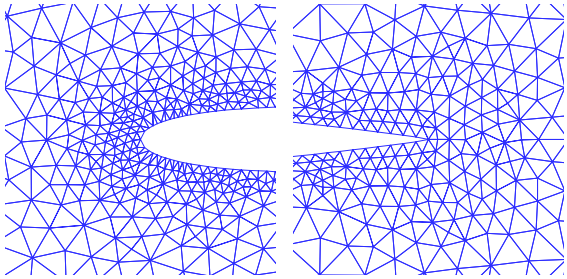

Fig 22: Close views of the final computational grid at the airfoil's leading and trailing edges

For consistency with previous cases, Euler flow solutions over the airfoil were obtained for 6000 iterations. The convergence history for this solution is shown in Figure 23.

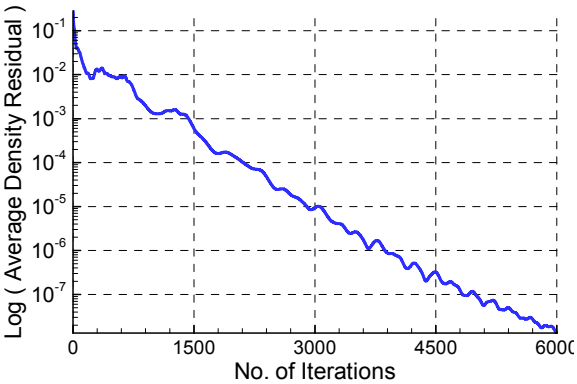

Fig 23: Convergence history for the solution of flow around a NACA0012 airfoil at $M_\infty = 0.8$ and $\alpha=1.25$

The Mach number contours for this transonic solution are shown in Figure 24. A strong shock wave can be seen on the upper surface of the airfoil, while a weaker shock is present on the lower surface. The Mack number contours in this figure are plotted over the final grid to show the grid size distribution at the shock propagation.

Similarly, Figure 25 shows the pressure contours for the solution of flow around NACA0012 airfoil at $M_\infty = 0.8$ and $\alpha=1.25$. Strong sharp contours can be seen near regions of shock and stagnation points, while smooth pressure contours are seen elsewhere.

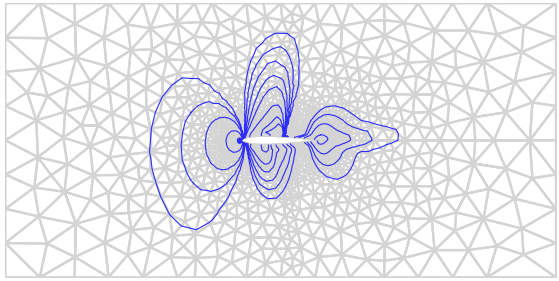

Fig 24: Mach number contours for the solution of flow around NACA0012 airfoil at $M_\infty = 0.8$ and $\alpha=1.25$

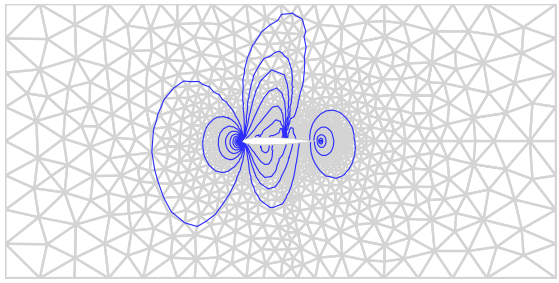

Fig 25: Pressure contours for the solution of flow around NACA0012 airfoil at $M_\infty = 0.8$ and $\alpha=1.25$

Finally, the coefficients of pressures on the upper and lower surfaces of NACA0012 airfoil are compared with solutions of Jameson [7] and are plotted in Figure 26. The shock at 60% of the chord is clearly captured and can be seen in this figure.

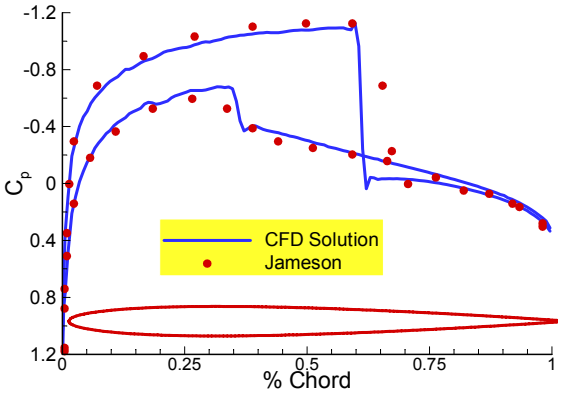

Fig 26: pressure coefficients on the upper and lower surfaces of NACA0012 airfoil at M∞ = 0.8 and α=1.25

4 Conclusion

An algorithm and procedure is developed and used to generate triangular unstructured grids. Grid refinement is based on a combination of point insertion and cell-subdivision methods. Edge-based and cell-based connectivity matrices were employed simultaneously to prevent long searches for identification of vertices, edges, or cells. The algorithm starts with a very coarse starting grid and adapts the solid surface geometry to a prescribed geometry. In this case, geometries were adapted to generate unstructured grids around a circular cylinder and a NACA0012 airfoil.

The finite volume approach for the governing equations and a multi-step Runge-Kutta scheme along with initial and boundary conditions have been formulated and programmed to develop an Euler flow solver.

The discretized Euler flow equations are essentially non-dissipative. Hence, convective terms of the flow equations can generate spurious oscillations that can affect the quality of the solution and can result in a deterioration of the convergence. A small amount of numerical dissipation is added to damp the oscillations enabling the flow solution to reach its steady state conditions.

The first computational example considers a subsonic flow solution with a free-stream Mach number of 0.25 over a circular cylinder. When the coefficients of pressure are compared with an analytical solution, apart from regions of high pressure gradients, the results are identical elsewhere on the cylinder (See Figure 12). The small deviations from the analytical solution at regions of high pressure gradients are due to the addition of numerical dissipation for solution stability.

The second example discussed the solution at a higher free stream Mach number on the cylinder. It was shown that small shock waves are present on the cylinder and their locations are identical to those given by published data.

The last case involved the generation of computational cells around an airfoil. An Euler flow solution for this airfoil was obtained and the shock capturing capability of the flow solver was demonstrated.

The integrity of the unstructured grid generation method and procedures has been demonstrated by obtaining inviscid flow solutions at various flow conditions.

Nomenclature

A_i	Area of cell *i*
$\vec{D}$	Dissipative term
E	Total energy
M_∞	Free-stream Mach number
p	Static pressure
$\vec{Q}$	Net flux out of the cell
u, v	Cartesian velocity components
Δx	Spatial increment in the x directions
Δy	Spatial increment in the y directions
Δt	Time step
$\varepsilon^{(2)}$	1st -order pressure switch
$\varepsilon^{(4)}$	3rd -order pressure switch
k_2, k_4	Adjustable dissipation coefficients
ν_i	Pressure sensor for cell *i*
γ	Ratio of specific heats
α	Angle of attack
ρ	Density

References:

[1] Thompson, J. F., Warsi, Z. U. A., and Mastin, C W, *Numerical Grid Generation: Foundations and Applications*, North Holand, 1985.

[2] Lohner, R., Progress in Grid Generation via the Advancing Front Technique, *Engineering with Computers*, Vol. 12, 1996, pp. 186-210.

[3] Marcum, D. L., *Unstructured grid generation using automatic point insertion and local reconnection*, Handbook of Grid Generation,

Chapter 18, Thompson, J. F., Soni, B. K., and Weatherill, N. P. Editors, CRC Press, 1998.

[4] Bahrainian, S. S., Construction of Hexahedral Block Topology and its Decomposition to Generate Initial Tetrahedral Grids for Aerodynamic Applications, *The 7th Iranian Aerospace Society Conference*, Sharif University of Technology, AE0457, Vol. 1, 2008, pp. 11-12.

[5] Jameson, A., and Mavriplis, D., Finite Volume Solution of the Two-Dimensional Euler Equations on a Regular Triangular Mesh, *AIAA Journal*, Vol. 24, No. 4, 1986.

[6] Jameson, A., Solution of the Euler Equations for Two Dimensional Transonic Flow by a Multigrid Method, *Reprinted from Applied Mathematics and Computation*, Vol. 13, 1983, pp. 327-355.

[7] Jameson, A., Baker, T. J., and Weatherill, N. P., Calculation of Inviscid Transonic Flow Over a Complete Aircraft, *AIAA Paper*, 1986.