ELSEVIER

Journal of Computational and Applied Mathematics 67 (1996) 73–93

JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS

# A multigrid method with unstructured adaptive grids for steady Euler equations

Kris Riemslagh, Erik Dick*

*Department of Mechanical and Thermal Engineering, Universiteit Gent, Sint-Pietersnieuwstraat 41, B-9000 Gent, Belgium*

Received 2 March 1994; revised 14 September 1994

**Abstract**

The multigrid method based on multi-stage Jacobi relaxation, earlier developed by the authors for structured grid calculations with Euler equations, is extended to unstructured grid applications. The meshes are generated with Delaunay triangulation algorithms and are adapted to the flow solution.

*Keywords:* Multigrid methods; Unstructured grids; Adaptivity; Steady Euler equations

## 1. Introduction

In two previous papers [6, 7], the authors analysed the use of multi-stage Jacobi relaxation in multigrid methods for two-dimensional steady Euler equations on structured grids. It was shown that by using multi-staging the classical defect correction procedure typically employed in relaxation multigrid methods [5, 10, 11] to obtain the solution of a high order accurate discretization, can be avoided. Full high order methods and mixed high order methods were demonstrated. In the full high order formulation, the same high order accurate operator is used on all grids. In the mixed formulation, a high order operator is used on the finest grid while on the coarser grids a different high order operator or a first order operator is used. In particular, the mixed discretization with high accuracy nonlinear TVD-operator on the finest grid and the linear high order Van Leer $\kappa = \frac{1}{3}$-operator on the coarser grids was shown to be efficient.

Defect correction procedures are found to be inefficient in cases where the results of the first order operator and the higher order operator differ significantly. The reason is that the inner (multigrid) iteration on the first order operator cannot reduce sufficiently the residue of the higher order operator. As a consequence, the global convergence rate is largely dominated by the outer

* Corresponding author. E-mail: erik.dick@rug.ac.be.


*SSDI* 0377-0427(94)00117-0

defect correction (nonmultigrid) iteration. In the mixed discretization method a better performance is obtained by two means. First, the higher order operator for which the solution is sought, i.e. the nonlinear TVD-operator, is used in all iterations on the finest grid. Second, a linear high order operator which is much closer to the nonlinear TVD-operator than the first order operator, is used on the coarse grids. Due to its linearity, this last operator allows the optimization of the multigrid procedure.

In this paper, we demonstrate how the mixed method can be employed on unstructured adaptive grids. The use of unstructured grids in computational fluid dynamics is nowadays well established. Unstructured grids offer great flexibility when dealing with complex geometries and allow mesh adaptation techniques. Adaptive mesh procedures can lead to an improvement in the quality of the computed solution by computing the specific flow features. Considerable success has already been achieved in this. We cite examples by Mavriplis [14], Weatherill and Soni [21], Löhner [13], and Barth [2].

The unstructured grids used here are of the Delaunay type. To generate a grid, a description of the domain boundary has to be given. Then points on the boundaries are introduced, based on local curvature and local grid spacing information. Using these points an initial grid is constructed, conformable to the boundary. Then, a criterion based on the edge length is applied to refine this initial mesh. On the first mesh a flow solution is calculated by means of the multigrid method. As soon as the residual drops below a threshold value, an adaption cycle is started. Using the same procedure as for the first grid, criteria are applied to refine the mesh. Instead of being based on mesh properties they are now based on flow properties like pressure and entropy differences. In this way, shock regions, stagnation regions and slip regions are captured accurately. Each newly generated mesh forms the finest mesh of a multigrid procedure.

Results are given for transonic flow over a NACA-0012 airfoil.

## 2. The flux-difference splitting method

In this section, we explain how the flux-difference splitting method can be employed on unstructured grids. Fig. 1 shows part of an unstructured triangular grid. In the vertex-centred finite volume method nodes are located at the vertices of the grid. Around every node a control volume is constructed by connecting the centres of gravity of the cells surrounding the node. At the boundaries, the control volumes are formed with the midpoints of the edges of the cells.

The flux-difference over the surface $S_{ij}$ is written as

$$F_j - F_i = s_{ij} A_{ij} (U_j - U_i), \tag{1}$$

where $U$ stands for the vector of conserved variables, $s_{ij}$ is the length of the surface and $A_{ij}$ is the discrete flux Jacobian. The first order upwind flux is defined by

$$F_{ij} = \tfrac{1}{2}(F_i + F_j) - \tfrac{1}{2} s_{ij} (A_{ij}^{+} - A_{ij}^{-})(U_j - U_i), \tag{2}$$

where $A_{ij}^{+}$ and $A_{ij}^{-}$ are the positive and the negative parts of $A_{ij}$. The upwind flux can also be written as

$$F_{ij} = F_i + s_{ij} A_{ij}^{-} (U_j - U_i). \tag{3}$$

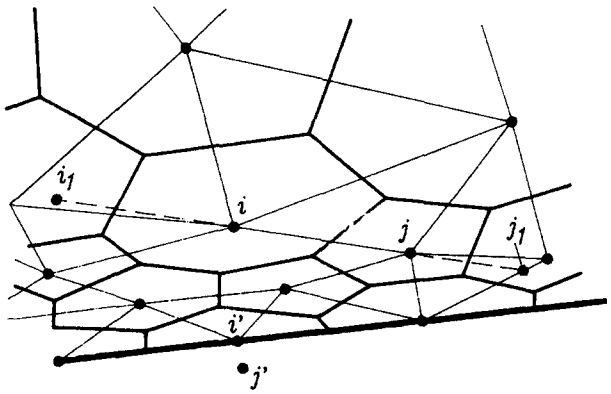


Fig. 1. Control volumes in the interior and on a solid boundary.

This flux expression shows the wave components incoming to the control volume around the node $i$. To determine the flux Jacobians, we use polynomial flux-difference splitting here. Details of this method are given in [4, 5]. The technical form of the splitting is not relevant, however, for the method described here. The second order accurate flux is defined with the flux extrapolation technique. The second order correction is calculated from a limited combination of differences between nodes $j$ and $i$ and shifted differences obtained by extrapolation of the line segment $ij$. The shifted differences are calculated by interpolation of the data in the points $i_1$ and $j_1$ as indicated in Fig. 1. The line segment $ij$ is extended with a length equal to $ij$ when this results in a point inside the adjacent cell (point $i_1$). Otherwise, the extrapolated point is taken to be on the opposing edge (point $j_1$). This latter restriction is necessary to preserve TVD-properties. The minmod limiter is always applied.

The resulting flux expression is

$$F_{ij} = F_i + s_{ij} A_{ij}^{-} (U_j - U_i) + F.C., \tag{4}$$

where $F.C.$ denotes the flux correction for high order accuracy.

## 3. Boundary conditions

The example to follow is the flow around an airfoil. This external type flow has a solid boundary, and a farfield boundary. At the solid boundary, impermeability is imposed by setting the convective part of the flux equal to zero. This requires a modification of the flux expression (4). For a point $i'$ on the boundary, the point $j'$ does not exist (see Fig. 1). This can be introduced in (4) by setting $F.C.$ to zero and by setting the values of the variables in the ficticious node $j'$ equal to the values of the variables in the node $i'$. The matrix $A_{ij}^{-}$ in (4) is then calculated with the values of the variables in the node $i'$. Of course, since the difference of the variables is zero, the first order difference part in (4) is also zero. The impermeability is introduced by replacing $F_i$ by $F_i - F_i^*$, where $F_i^*$ is the convective part of the flux. The term $-F_i^*$ can be seen as a new flux correction term $F.C.$ As will be

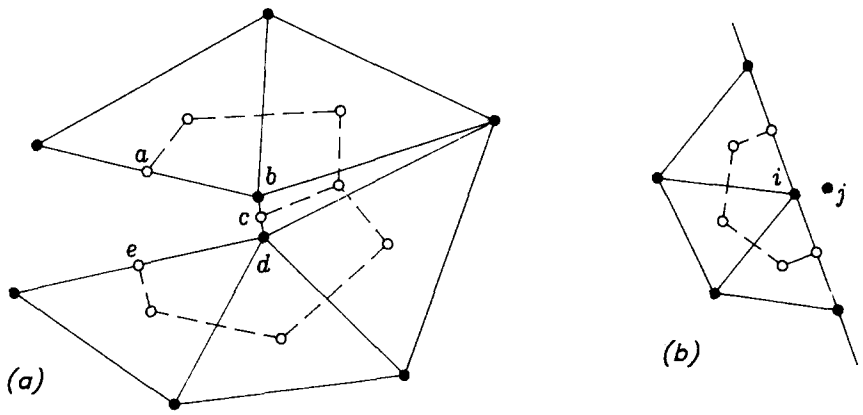


Fig. 2. Treatment of the trailing edge and the farfield boundary.

discussed in the next section, the matrix $A_{ij}^{-}$ at a solid boundary plays an important role in the relaxation method, although it is multiplied by a zero term.

Special attention is to be given to the trailing edge of the airfoil. In order to impose a correct Kutta-condition, the trailing edge is represented by two nodes very close to each other ($10^{-8}$ of the chord in the actual calculation). The local grid around the trailing edge is schematically shown in Fig. 2. For clarity, the two nodes at the trailing edge are separated in the figure. The control volumes are indicated. For node $b$, imposing impermeability is approximately equivalent to imposing a velocity direction along $ac$. Similarly for node $d$, the direction is approximately $ec$. If the trailing edge were represented by the single node $c$, impermeability would result in imposing approximately the direction $ae$ for the velocity, which is physically erroneous.

The farfield boundary around the airfoil is formed by a hexagon at a distance of 100 chord lengths from the airfoil (Fig. 9). We can use a very large distance in an unstructured grid application since the mesh size near the farfield boundary can be very large. For node $i$ on the boundary (Fig. 2(b)), the outside node $j$ necessary to define the flux through the boundary face is given the free stream state values. By means of flux-difference splitting, the actual node on the boundary then takes only the part of the state of the farfield that corresponds to ingoing characteristics.

## 4. The multi-stage Jacobi relaxation

We bring multi-staging into the Jacobi method in the same way as multi-staging is used for time stepping methods and we use the optimization results known for time stepping schemes.

For the time-dependent Euler equations, the discrete set of equations associated with the node $i$ reads

$$Vol_i \frac{dU_i}{dt} + \sum_j A_{ij}^{-}(U_j - U_i)s_{ij} + \sum_j F.C. = 0, \tag{5}$$

where the index $j$ loops over the faces of the control volume and the surrounding nodes. A single stage time stepping method on (5) gives

$$\left(\frac{Vol_i}{\Delta t}\right)\delta U_i + \sum_j \bar{A}_{ij}\,(U_j^n - U_i^n)s_{ij} + \sum_j F.C.^n = 0. \tag{6}$$

The Jacobi relaxation applied to the steady part of (5) reads

$$\sum_j \bar{A}_{ij}\,(U_j^n - U_i^{n+1})s_{ij} + \sum_j F.C.^n = 0. \tag{7}$$

Using the increments $\delta U_i = U_i^{n+1} - U_i^n$, this gives

$$\left(-\sum_j \bar{A}_{ij}\,s_{ij}\right)\delta U_i + \sum_j \bar{A}_{ij}\,(U_j^n - U_i^n)s_{ij} + \sum_j F.C.^n = 0. \tag{8}$$

The $4\times 4$ matrix coefficient of $\delta U_i$ in (8) is nonsingular. The coefficient matrix employed in the relaxation method only can become singular for zero velocity, i.e., convective eigenvalues on all control volume faces equal to zero. In a practical calculation this does not happen. Even at a stagnation point, the velocity is only exactly zero on one face of a control volume. The other faces then still contribute in a way that singularity is avoided. The matrix can be near to singular. If wanted, the condition number of the matrix can be improved by a so-called fix. This means that the contribution of the velocity components to the coefficient matrix are not allowed to be smaller than a prescribed minimum value (usually of the order of 0.01 to 0.001 times the free stream velocity). We did not use such a fix.

In the expressions (6) to (8), the matrices $\bar{A}_{ij}$ are on the time or relaxation level $n$. The difference between (single stage) Jacobi relaxation (8) and single stage time stepping (6) is seen in the matrix coefficient of the vector of increments $\delta U_i$. In the time stepping method, the coefficient is a diagonal matrix. In the Jacobi method, the matrix is composed of parts of the flux Jacobians associated with the different faces of the control volume. The collected parts correspond to waves incoming to the control volume. In time stepping, the incoming waves contribute to the increment of the flow variables all with the same weight factor. In the Jacobi relaxation the corresponding weight factors are proportional to the wave speeds. As a consequence, Jacobi relaxation can be seen as time stepping in which all incoming wave components are scaled to have the same effective speed, i.e., all have a $CFL$-number equai to unity.

This last observation is very important. The coefficients of a multi-stage scheme can only be optimized for a single wave equation [6, 7]. The $CFL$-number plays a dominant role in this optimization. By the introduction of multi-staging into the Jacobi relaxation, there is a guarantee that the optimization applies equally well to all wave components in the field. In a time stepping scheme, the optimization can only be reached for one wave component.

There is similarity between the multi-stage Jacobi method and the time stepping method with preconditioning introduced by Turkel [17] and the characteristic time stepping method introduced by Van Leer et al. [19]. The difference is that the scaling of the wave components in the multi-stage Jacobi method is realized on the discrete equations of the original steady Euler system while in the other methods it is done analytically before discretization through multiplication of the vector of time derivative terms by a matrix.

For a node on a solid boundary, an expression similar to (8) is obtained provided that for a face on the boundary the flux expression (4) is used and that the difference in the first order flux-difference part is introduced as $U_i^n - U_i^{n+1}$, similar to the term $U_j^n - U_i^{n+1}$ which is used for a flux on an interior face. So, in order to avoid a singular matrix coefficient of the vector of increments in (8), this treatment at the boundaries is necessary. A boundary node can then be updated in the same way as a node in the interior.

A multi-stage scheme ($k$ stages) is defined by

$$\begin{aligned} U^0 &= U_i^n, \\ U^1 &= U^0 + \alpha_1 v \delta U^0, \\ U^2 &= U^0 + \alpha_2 v \delta U^1, \\ &\vdots \\ U^k &= U^0 + \alpha_k v \delta U^{k-1}, \\ U_i^{n+1} &= U^k, \end{aligned} \tag{9}$$

where $\delta U$ is the increment obtained from single-stage Jacobi relaxation. The coefficient $v$ indicates the $CFL$-number. The last coefficient in the stepping series ($\alpha_k$) is equal to 1.

In the sequel, we use a five stage scheme on the finest grid adapted for the TVD-operator with coefficients $\alpha_1 = 0.066$; $\alpha_2 = 0.16$; $\alpha_3 = 0.307$; $\alpha_4 = 0.576$; $\alpha_5 = 1.0$; $v = 1.55$. We denote this set as TVD5 [7]. On the coarse grids we use a three stage scheme with coefficients $\alpha_1 = 0.2884$; $\alpha_2 = 0.501$; $\alpha_3 = 1.0$; $v = 1.3254$. We denote this set as VL 1.33 [7]. These sets were shown to be optimum for a mixed discretization with the TVD-operator on the finest grid and the Van Leer $\kappa = \frac{1}{3}$-operator (K3) on the coarser grids.

## 5. The mesh generation

Unstructured triangular grids are very attractive for two-dimensional flow calculations. One reason is that geometries of arbitrary complexity can be meshed. Another is that mesh adaption by the addition of points is very simple. The grid generation technique used here leads to Delaunay triangular meshes. Many algorithms exist and are used for the generation of such meshes, we cite the methods of Baker [1], Mavriplis [15] and George [8], among many others. These methods use the basic algorithms of Lawson [12], Bowyer [3] and Tanemura et al. [16]. A Delaunay triangulation of a planar set of points can be characterized as the unique triangulation such that the circumcircle of every triangle contains no point other than the forming points of the triangle. This circle criterion can be shown to be equivalent to the equiangular property, which selects the triangulation that maximizes the minimum of the six angles in any pair of two triangles making up a convex quadrilateral [8]. The meshes are generated by the following algorithms.

### *5.1. Triangulation of a polygon*

The first algorithm is based on the advancing front method, developed by Tanemura et al. [16], but simplified to work on a set of nodes which are already connected. The nodes have to lie on the

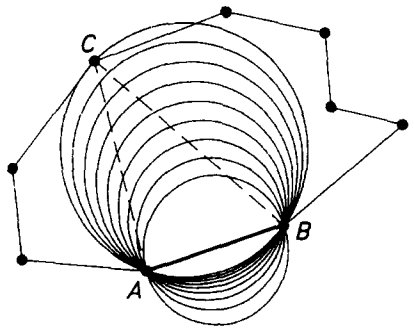


Fig. 3. Generation of the Delaunay triangulation of a nonconvex polygon.

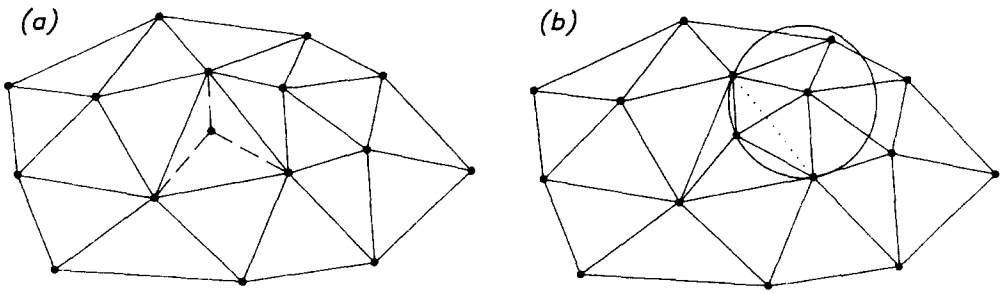


Fig. 4. Adding a node to an existing Delaunay triangulation.

boundary of a polygon. With this algorithm a given, not necessarily convex, polygon can be triangulated. No extra nodes are added. Fig. 3 explains the procedure.

For an arbitrary edge of the polygon, the node that has to be connected to the edge is determined. For a Delaunay triangulation, this node results in the smallest circumcircle. It can be found by a walk along the boundary of the polygon. Connecting the edge to the node splits up the polygon in to two smaller polygons, or reduces the size of the original polygon. Applying the algorithm recursively results in a constrained Delaunay triangulation of the given polygonal boundary. If the number of boundary nodes is limited, the triangulation can be generated efficiently.

### *5.2. Node incremental algorithm*

The second basic algorithm used in the grid generation makes the addition of a node to an existing triangulation possible. The new node is connected to three or four existing nodes: three nodes if the new node lies inside a triangle, four nodes if the new node lies on an edge. The triangulation is made Delaunay by a diagonal swapping algorithm [12]. For every node connected to the new node, the circle criterion is checked. If the circumcircle contains the fourth node of the quadrilateral, as shown in Fig. 4, the diagonal is swapped. This results in two new triangles that

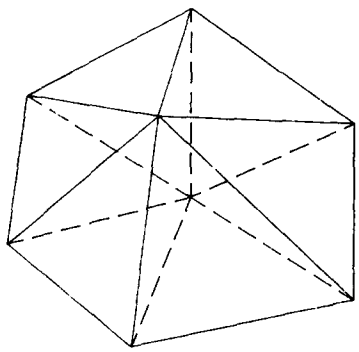

Fig. 5. Action of the smoother.

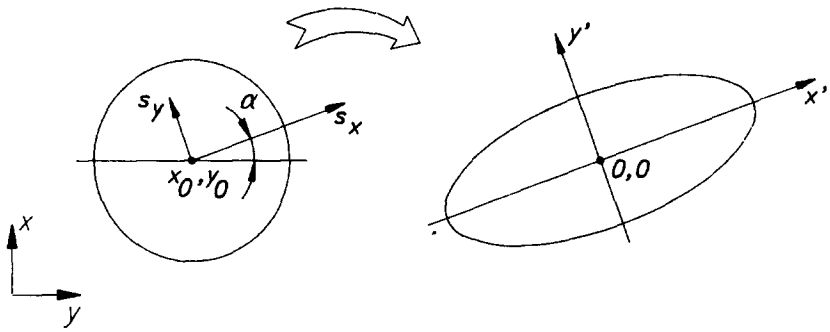


Fig. 6. Local scaling transforming a circle into an ellipse.

have to be checked by the circle criterion. When all triangles connected to the new node satisfy the circle criterion, the Delaunay property of the triangulation is restored.

### 5.3. *Node smoother*

The third algorithm is a smoother, which moves the nodes but does not change the topology of the mesh. Every node is moved towards the midpoint of the polygon it is connected to, as is shown in Fig. 5, as if the node were connected to its neighbours by springs. After a few loops over all the nodes, the mesh quality is much improved. After the smoothing, the Delaunay property is restored by swapping.

### 5.4. *Locally scaled space*

To enable anisotropy in the construction of the grid, the above-described algorithms are applied in a transformed, locally scaled space. Every node has its own transformation parameters, $(\alpha, s_x, s_y)$. The transformation is a rotation over the angle $\alpha$, followed by a rescaling of the new $x$ and $y$ axes by $s_x$ and $s_y$. The criteria by which the Delaunay triangulation is built, are based on transformed properties. The initial transformation parameters are calculated from the required boundary mesh spacing and boundary edge direction. Transformation parameters of new nodes are interpolated. Fig. 6 illustrates the effect of the scaling.

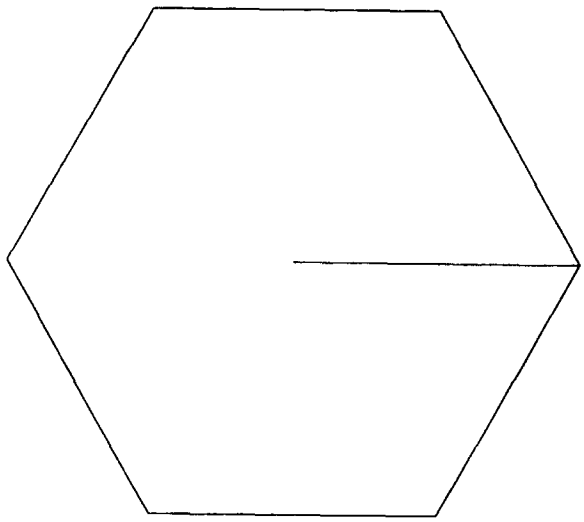

Fig. 7. Starting mesh consisting of one simply connected cell around the NACA-0012 profile. The profile is located at the centre of the hexagon.

## 6. Application

The mesh generation procedure can be used for any two-dimensional mesh, and is illustrated here for an airfoil NACA-0012 with a hexagonal farfield boundary 100 chords away from the profile as shown in Fig. 9. The Mach number of the incoming flow is 0.8 and the profile has an angle of attack of 1.25°.

### *6.1. The first mesh*

Given the analytical description of the NACA-0012 profile, a set of nodes located on the profile is determined. These nodes are used as beginning and end points of the Bezier curves by which the geometry is described. Additional central points of the Bezier curves are calculated by requiring that the profile is continuous up to the second derivative. The larger the chosen set of nodes, the higher the accuracy by which the profile is represented, but also the larger the minimum number of nodes on the profile. The mesh generator presently in use is not allowed to remove the beginning and end points of the Bezier curves. Because the curvature is largest on the leading edge, the nodes are chosen with a stretching of 1.05 with the highest density at the leading edge. The total number of points on the profile is 61.

Using the description of the boundary, a polygon is constructed so that all boundary nodes can be reached walking along the boundary edges. For this purpose the NACA-0012 profile is connected to the farfield. The starting mesh, given in Fig. 7 consists of one cell. The cell was generated by substituting every Bezier curve by one boundary edge.

The triangulation of the starting mesh is carried out by applying the first mesh generation algorithm, i.e., the triangulation of a polygon. The result is a triangulation of only boundary nodes. In every boundary node, local transformation parameters are determined. These parameters are

used to define the locally required mesh spacing in two dimensions. On the farfield, the maximum edge length is taken to be 50 chord lengths in both directions. On the profile, the maximum length is chosen to be the actual length of the boundary edges for the tangential direction and $\frac{1}{3}$ of the actual length in the normal direction.

Now that the local transformation parameters are determined, a triangulation is generated of which all the edge lengths scaled by their local parameters are smaller than one. This is accomplished by looping over the edges and evaluating the length of every edge. When the length is larger than one, a node is added to the triangulation in the centre of the edge by applying the second mesh generation algorithm. When a node is added, the Delaunay triangulation is regenerated, and local transformation parameters are interpolated. After all nodes are added, the mesh generation is finished with a smoothing phase. Also during the smoothing, the local transformation parameters are used to preserve mesh stretching and anisotropy. The result is shown in Fig. 9. On this first mesh the flow is calculated.

### 6.2. *Unstructured multigrid*

Before the actual calculations start, the coarse meshes that are to be used in the multigrid cycle are generated. These meshes are built with telescoping nodes, i.e., all the nodes of a coarse mesh appear in all the finer meshes. Since the finest mesh is already given, the multigrid meshes are generated from fine to coarse. The generation of a coarse mesh consists of two phases. In the first phase a set of nodes from the fine mesh is selected to appear in the coarse mesh. Then the coarse mesh is generated.

During the selection phase, special attention must be paid to the boundary nodes. Typically, half of the boundary nodes are selected. Some of the boundary nodes are crucial for the description of the geometry and have to be selected every time during the coarsening, e.g. the leading and trailing edges of an airfoil. In [9] a coarsening algorithm is given which is based on the idea that all the neighbours of a coarse node can be removed from the list of selected nodes. First, the nodes of the fine mesh are stored in a list of nodes. In this list, the special nodes appear first, then the boundary nodes, and then the other nodes. Initially, all nodes are selected. The set of nodes is determined by means of the following algorithm.

```
Do for every node of the list:
  if node is selected
     then unselect all neighbours of the node;
end do.
```

The algorithm reduces the number of nodes roughly by a factor of 4. As a result of the algorithm, at least one neighbour of every fine grid node appears in the coarse grid, unless the node itself appears in the coarse grid.

After the selection of the set of nodes that will form the coarse mesh, these nodes have to be connected. Here we do this by removing the nonselected nodes from the fine mesh. When a node is removed, all the edges connected to the node are also removed, leaving a cavity in the mesh. This cavity is retriangulated by the first mesh generation algorithm. Because the average number of connected neighbours is six, and usually below ten, the retriangulation of a cavity is very fast, and independent of the total number of nodes in the mesh. When all nodes are removed, the coarse mesh is obtained.

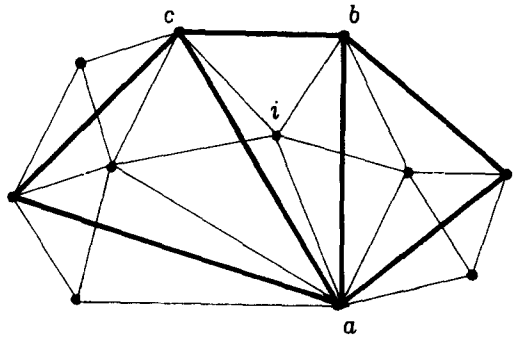


Fig. 8. Integrid operations on mesh with telescoping nodes.

The property of telescoping nodes reduces the complexity of the intergrid operations. The injection of flow variables is straightforward, as is the prolongation of corrections in the telescoping nodes. For the fine grid nodes that do not appear in the coarse grid, corrections can be interpolated on the coarse grid, as for totally unrelated meshes. Here, the correction in node $i$ is taken to be the average of the corrections in nodes $a, b, c$, that appear on the coarse mesh (see Fig. 8).

For the restriction an algorithm is used that keeps the required intergrid connectivity information to a minimum. First, the residuals $R_j$ are averaged or weighted on the fine grid. This averaging, which is done implicitly or explicitly has the function of a low-pass filter. The resulting quantity is then divided by the control volume area on the fine grid, injected to the coarse grid, and multiplied with the coarse grid volume area. With $\tilde{R}_o$ the quantity in node $o$, calculated with the residual $R_o$ in node $o$ and the residuals $R_j$ in the surrounding nodes, the explicit residual weighting is given by

$$\frac{\tilde{R}_o}{V_o} = \frac{R_o + \varepsilon \sum_{j=1}^{n} R_j}{V_o + \varepsilon \sum_{j=1}^{n} V_j}, \tag{10}$$

while the implicit residual weighting is given by

$$\frac{\tilde{R}_o}{V_o} = \frac{R_o + \varepsilon \sum_{j=1}^{n} \tilde{R}_j}{V_o + \varepsilon \sum_{j=1}^{n} V_j}, \tag{11}$$

where $V_o$ and $V_j$ denote control volume areas in the node and in the surrounding nodes. Since the function of the weighting is to filter out the high frequencies, a few Jacobi sweeps (e.g., 3) are enough to calculate $\tilde{R}_o$ in the implicit weighting. In the explicit formula $\varepsilon = 0.5$ and in the implicit formula $\varepsilon = 2.0$ [7]. Here we use the implicit formula.

For the first mesh of Fig. 9 two coarse meshes were generated, given in Fig. 10. The multigrid procedure uses the three grids.

### 6.3. *The flow calculation*

After all multigrid meshes are determined, an initialization phase follows. For every mesh the following is done.

- Every boundary has a type of boundary condition associated with it.

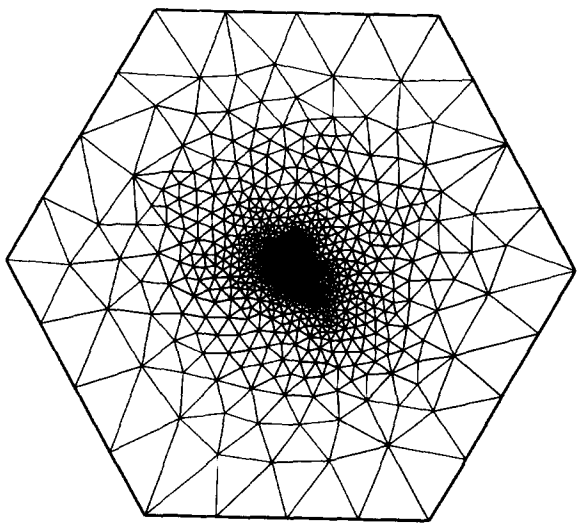

Fig. 9. The first mesh with 1944 nodes.

- The boundary edges are duplicated. In a vertex-centred formulation every boundary edge is associated with one internal control volume face and one boundary control volume face.
- The parameters that determine the initial flow field are given.
- The length of every control volume edge is calculated.
- The interpolation parameters for the shifted differences for the second order correction are calculated.
- For every control volume the area is calculated.

The time needed for these preparations, including the generation of the coarse meshes, is only a few work units. A work unit is defined as the work needed for the calculation of one basic relaxation on the current finest mesh (one stage of a multi-stage Jacobi).

The calculations start in full multigrid mode (FMG). On the coarsest level the initial flow field is determined. After a few multi-stage relaxations with the Van Leer $\kappa = \frac{1}{3}$-operator (K3), the solution is transferred to the finer mesh. Now 2 FAS multigrid cycli with one pre- and one postrelaxation are carried out before the solution is transferred to the finest mesh. Also on the finest mesh two multigrid cycli are done with the K3-operator before the TVD-operator is turned on only on the finest level. The multigrid is of the W-type with one pre- and one postrelaxation at every level. After 50 W-cycli the residual has dropped 10 orders of magnitude. The solution is given by means of iso-Mach lines in Fig. 11.

### *6.4. Adaptive refinement*

One of the most important advantages of unstructured grids is the possibility to refine the mesh locally during the computation. Successive mesh concentrations in critical zones may be performed without knowing them at mesh creation time. The optimal way is to start the computation on a coarse and smooth grid which is not at all specific to a hypothetical solution, and refine during

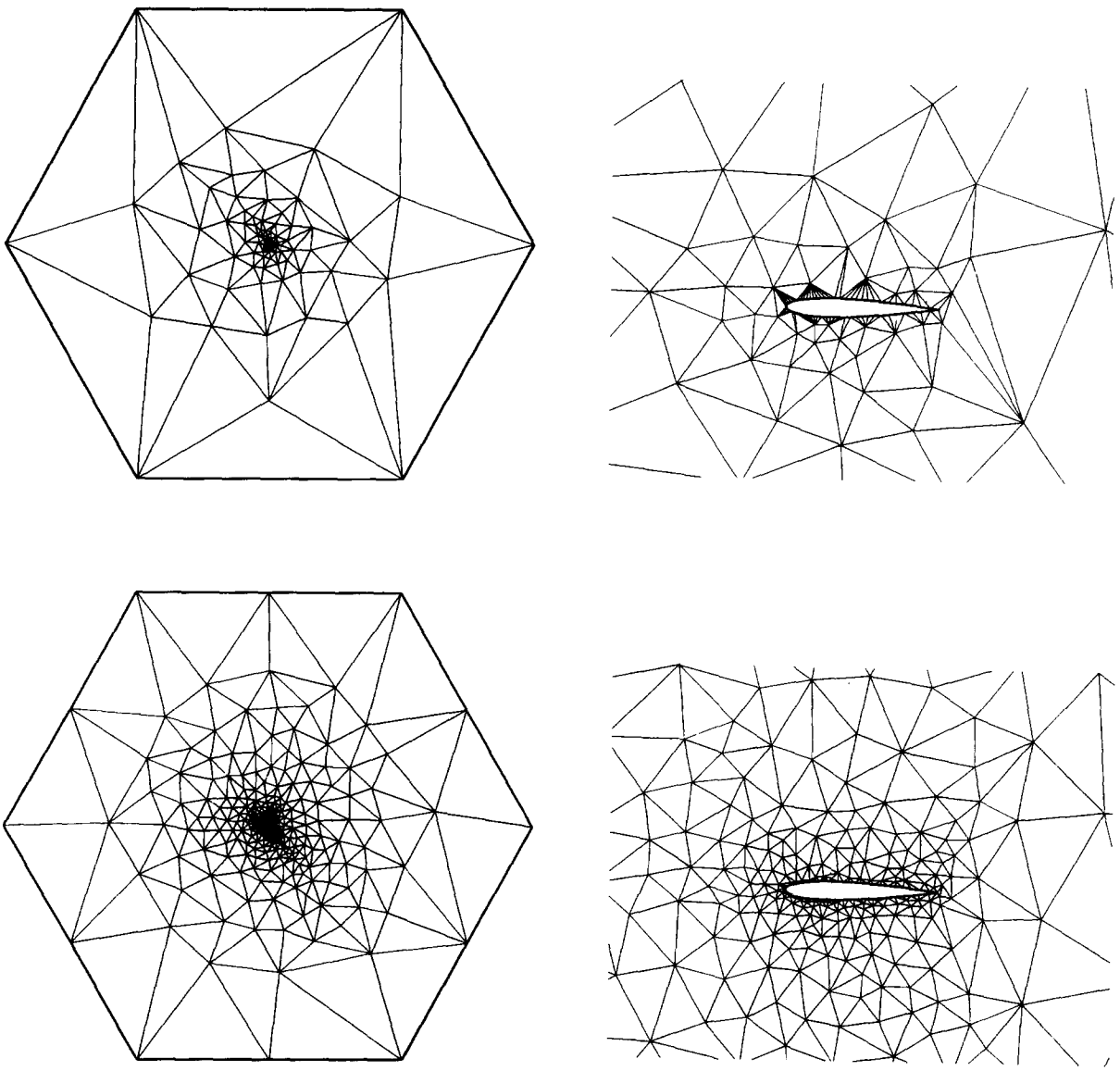

Fig. 10. Coarser meshes that belong to the first mesh. Top: coarsest mesh with 187 nodes. Bottom: coarse mesh with 548 nodes. Right: enlargement around profile.

the calculation, with criteria characteristic for the particular physical solution. In the physical problem treated here, transonic flow over an airfoil, interesting regions are the shockwaves, which have to be precisely located and also well quantified; the stagnation regions; and the trailing edge zone where a contact discontinuity may exist. Ad hoc criteria can be defined characterizing such flow features. We used criteria based on gradients of local flow parameters. Although criteria based on error estimators might be possible, we did not consider these.

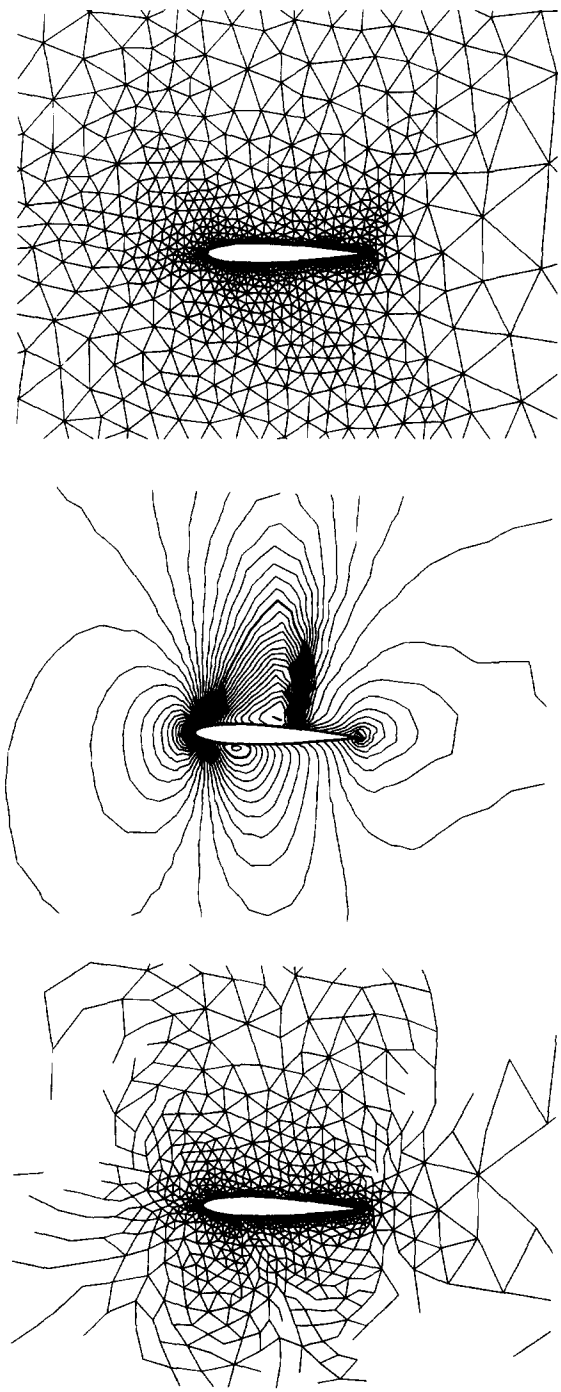

Fig. 11. Top: enlargement of the first mesh (1944 nodes). Middle: Mach contour lines per 0.02; sonic line thick. Bottom: edges that are selected for refinement.

Table 1
Number of nodes of multigrid meshes during different adaption phases

| | | | | |
|---|---|---|---|---|
| Finest mesh | 1944 | 4008 | 7080 | 8566 |
| First coarse mesh | 548 | 1019 | 1738 | 2031 |
| Second coarse mesh | 187 | 302 | 476 | 557 |
| Third coarse mesh | | | 158 | 181 |

The first criterion is based on the pressure difference over an edge. If

$$|p_i - p_j| L_{ij} \geqslant \frac{|p_{\max} - p_{\min}| L_{\text{ref}}}{S_p}, \tag{12}$$

then the edge $ij$ will be refined by placing a new point into the centre of the edge. In this formulation $p_i$ is the pressure at node $i$, $L_{ij}$ is the length of the edge. The variables in the right-hand side are minimum or maximum values taken over all the nodes. $L_{\text{ref}}$ is a reference length of the problem (the chord length), and $S_p$ is the sensitivity constant for the pressure criterion. This criterion triggers shockwaves and stagnation regions to be refined. A second criterion is based on the entropy difference over an edge, similar to the first criterion:

$$|s_i - s_j| L_{ij} \geqslant \frac{|s_{\max} - s_{\min}| L_{\text{ref}}}{S_s}, \tag{13}$$

where $s$ is the entropy ($\ln p/\gamma - \ln \rho$, with $\gamma$ the adiabatic constant). This criterion triggers shock waves and tangential discontinuities. To include the new points in the mesh, the earlier described procedure is used. In the refinement criteria, the right-hand side of the inequalities are constructed in such a way that the same sensitivity constant can be used on all meshes.

For the NACA-0012 profile the sensitivity constants were chosen as $S_p = 250$ and $S_s = 60$. The edges of the first mesh that are selected by the criteria are shown in Fig. 11.

A large part of the edges is selected. In the stagnation regions the values on the left-hand sides of the equations (12) and (13) diminish as a result of subsequent adaption solution cycles. This is caused by the reduction of the pressure difference over an edge, and by the reduction of the length of the edge. In shock regions the left-hand side diminishes only because the length becomes smaller.

### 6.5. *Results of the calculation*

On the adapted mesh another 50 W-cycli are performed. For this purpose new multigrid meshes are generated. The initial solution on the adapted mesh is interpolated from the previous solution. After the flow calculation the adaption is repeated. Three adaption phases are carried out. The number of nodes in the different meshes is given in Table 1. The evolution of the meshes and associated solutions is shown in Figs. 12, 13 and 14. On the last mesh the contour lines of the entropy parameter $s = p^{1/k}/p$ are given.

In Fig. 15 the pressure distribution over the profile is given by

$$C_p = \frac{p_\infty - p}{\frac{1}{2}\rho_\infty u_\infty^2} \tag{14}$$

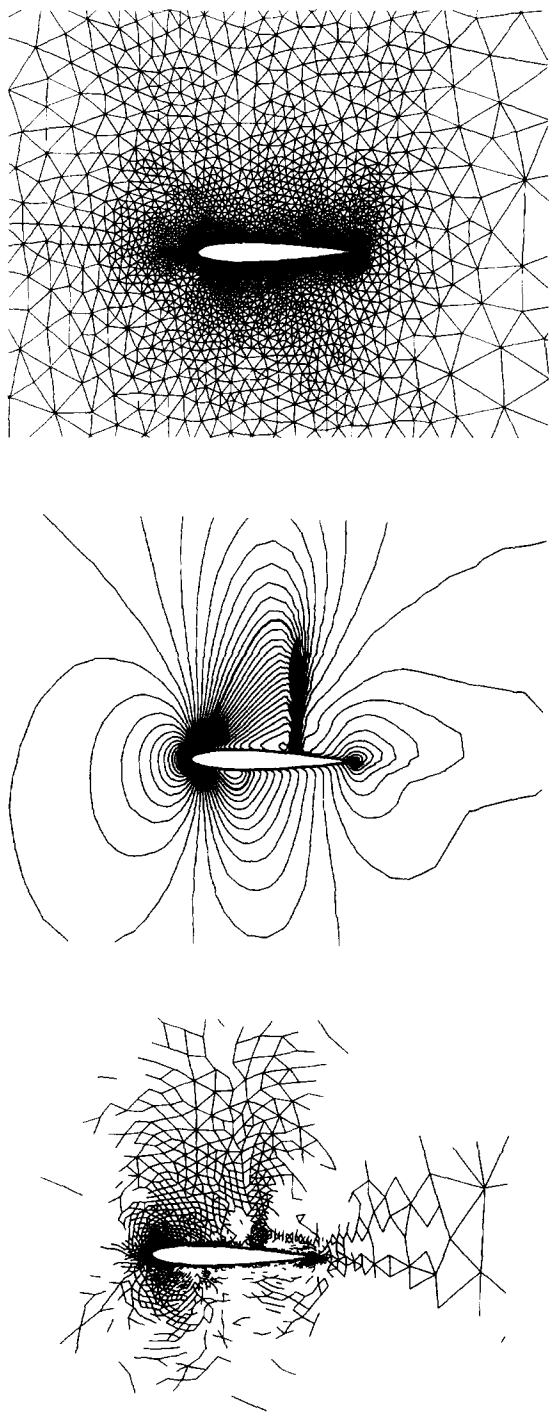

Fig. 12. Top: enlargement of the second mesh (4008 nodes). Middle: Mach contour lines per 0.02; sonic line thick. Bottom: edges that are selected for refinement.

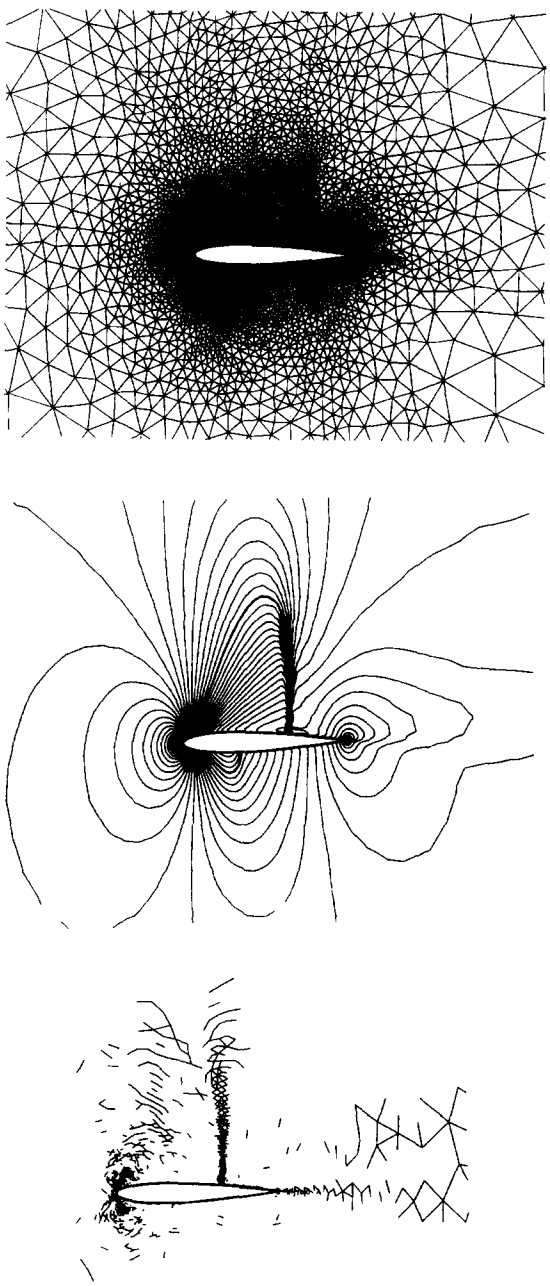

Fig. 13. Top: enlargement of the third mesh (7080 nodes). Middle: Mach contour lines per 0.02; sonic line thick. Bottom: edges that are selected for refinement.

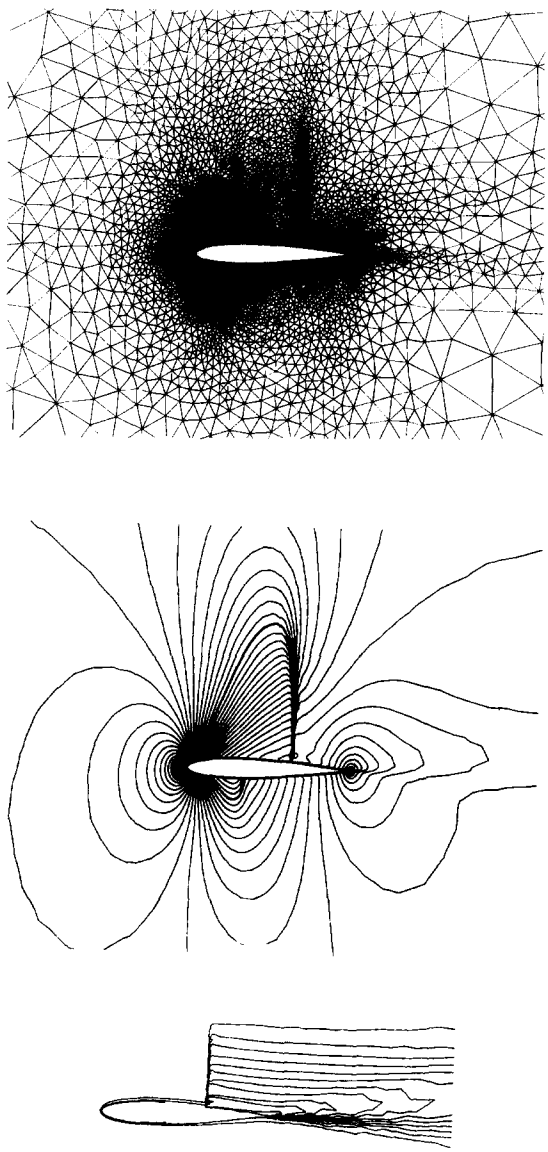

Fig. 14. Top: enlargement of the fourth mesh (8566 nodes). Middle: Mach contour lines per 0.02; sonic line thick. Bottom: contour lines per 0.001 of the entropy parameter.

with $p_\infty$, $\rho_\infty$ and $u_\infty$ as the pressure, the density and the velocity at the farfield, respectively. The irregularities in the pressure distribution are the consequence of the abrupt change in cell size from one cell to another.

Table 2 shows the obtained lift and drag coefficients on the different meshes. The lift coefficient does not converge monotonically with the grid refinement. This behaviour was also observed in the grid refinement study on the same test case by van der Burg et al. [18]. A lift coefficient around 0.35,

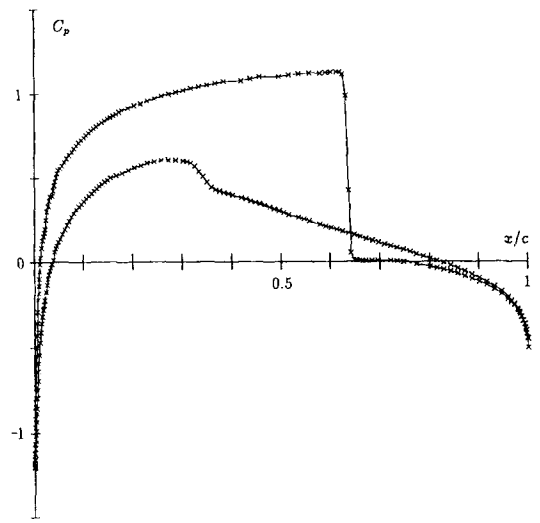


Fig. 15. Pressure distribution along the profile.

Table 2
Lift and drag coefficients on the different meshes

| # nodes | 1944 | 4008 | 7080 | 8566 |
|---|---|---|---|---|
| $C_L$ | 0.3353 | 0.3443 | 0.3529 | 0.3433 |
| $C_D$ | 0.0412 | 0.0298 | 0.0267 | 0.0255 |

as found here, corresponds well with the result obtained in [18] on a structured $257 \times 129$ grid (0.3455) and the results given as reference results obtained on different grids in the AGARD report on test cases for inviscid flow field methods [20] (0.3463 to 0.3736). The drag coefficient shows a monotone convergence with grid refinement. The value obtained on the finest grid (0.0255) is still somewhat larger than the value obtained in [18] (0.0221) and the values given in [20] (0.0221 to 0.0230).

Fig. 16 shows the convergence behaviour of the $L_\infty$-norm of the residual as a function of the number of work units. The $L_\infty$-norm is taken over all equations and over all nodes. The three adaptions are marked by a jump in the residual value. The figure must be interpreted with care since the time needed to calculate one work unit depends on the number of nodes of the finest mesh of the multigrid. Therefore one work unit in the first phase is much cheaper than one in the following phases. This presentation was chosen to show that the residual drop during the different phases is almost the same, indicating the mesh independency of the multigrid convergence. The mean convergence rate over the four phases is 1 magnitude per 6 cycles or 115 work units.

In Fig. 17 an enlargement of the leading edge region and the shock region on the suction surface are shown.

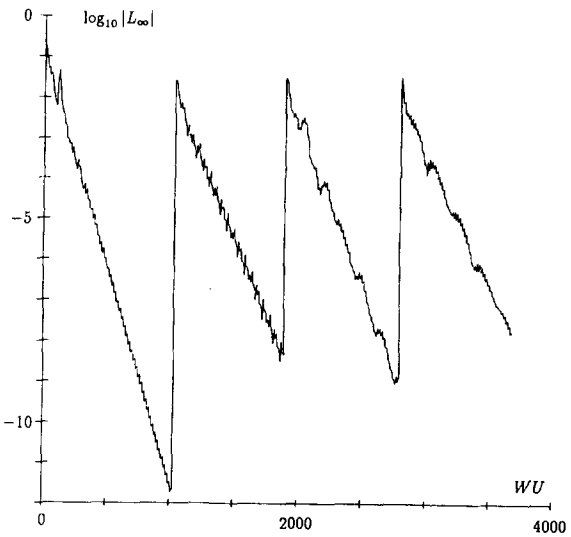


Fig. 16. Convergence behaviour for the second order TVD-scheme (TVD5 on finest and VL 1.33 on coarser meshes). Logarithm of residual as a function of work units.

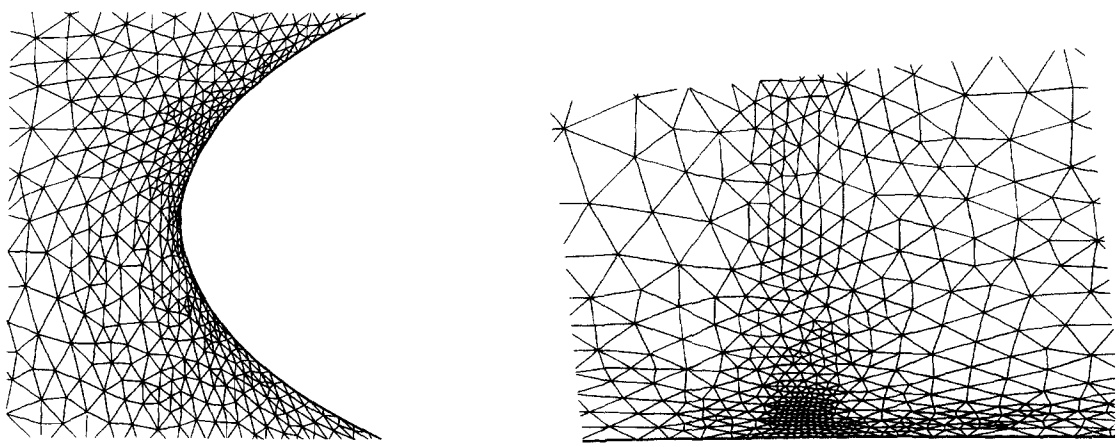

Fig. 17. Enlarged view of the final mesh, the stagnation region and the shock region.

## 7. Conclusion

By the combination of an adaptive unstructured grid generation technique, suitable refinement criteria and a high resolution flow solver, high quality solutions for Euler equations are generated. By using the multigrid method, this solution is obtained very efficiently.

## Acknowledgements

The research reported here was carried out under contract IT/SC/13, which forms part of the Belgian National Programme for Large-Scale Scientific Computing and under contract IUAP/17, hich forms part of the Belgian National Programme on Interuniversity Poles of Attraction, both initiated by the Belgian State, Prime Minister's Office, Science Policy Programming.

## References

[1] T.J. Baker, The generation of tetrahedral meshes around complete aircraft, in: *Proc. Second Internat. Conf. on Numerical Grid Generation in CFD, Miami* (1988) 675–685.
[2] T.J. Barth, Aspects on unstructured grids and finite volume solvers for the Euler and Navier–Stokes equations, AGARD report 787 (1992) 6.1–6.61. VKI Special Course on Unstructured Grid Methods for Advection Dominated Flows.
[3] A. Bowyer, Computing Dirichlet tessellations, *Comput. J.* **24** (1981) 162–166.
[4] E. Dick, Multigrid methods for steady Euler and Navier–Stokes equations based on polynomial flux-difference splitting, in: *Proc. Third European Conf. on Multigrid Methods, Bonn*, Int. Series on Numerical Mathematics **98** (Birkhäuser, Basel, 1991) 1–20.
[5] E. Dick, Second order formulation of a multigrid method for steady Euler equations trough defect-correction, *J. Comput. Appl. Math.* **35** (1991) 159–168.
[6] E. Dick and K. Riemslagh, Multi-staging of Jacobi relaxation to improve smoothing properties of multigrid methods for steady Euler equations, *J. Comput. Appl. Math.* **50** (1994) 241–254.
[7] E. Dick and K. Riemslagh, Multi-staging of Jacobi relaxation in multigrid methods for steady Euler equations, II, *J. Comput. Appl. Math.* **59** (1995) 339–348.
[8] P.L. George, Génération automatique de maillages, *Rech. Math. Appl.* **16** (Masson, Paris, 1990).
[9] H. Guillard, Node-nested multigrid with Delaunay coarsening, Rapport de recherche 1898, INRIA, 1993.
[10] P.W. Hemker, Defect correction and higher order schemes for the multigrid solution of the steady Euler equations, in: *Proc. Second European Conf. on Multigrid Methods, Bonn*, Lecture Notes in Math. **1228** (Springer, New York, 1986) 149–165.
[11] B. Koren, Defect correction and multigrid for an efficient and accurate computation of airfoil flows, *J. Comput. Phys.* **77** (1988) 183–206.
[12] C.L. Lawson, Generation of a triangular grid with application to contour plotting, Technical Report 299, CalTech, 1972.
[13] R. Löhner, An adaptive finite element scheme for transient problems in CFD, *Comput. Methods Appl. Mech. Eng.* **61** (1987) 323–338.
[14] D.J. Mavriplis, Multigrid solution of the 2-D Euler equations on unstructured triangular meshes, *AIAA J.* **26** (1988) 824–831.
[15] D.J. Mavriplis, Unstructured mesh algorithms for aerodynamic calculations, in: *Proc. 13th International Conf. on Numerical Methods in Fluid Dynamics, Roma*, Lecture Notes in Physics **414** (Springer, Berlin, 1993), 57–77.
[16] M. Tanemura, T. Ogawa and N. Ogita, A new algorithm for three-dimensional Voronoi tesselation, *J. Comput. Phys.* **51** (1983) 191–207.
[17] E. Turkel, Preconditioned methods for solving the incompressible and low speed compressible equations, *J. Comput. Phys.* **72** (1987) 277–298.
[18] J.W. van der Burg, J.G.M. Kuerten and P.J. Zandbergen, Improved shock-capturing of Jameson's scheme for the Euler equations, *Internat. J. Numer. Methods Fluids* **15** (1992) 649–671.
[19] B. Van Leer, W.T. Lee and P.L. Roe, Characteristic time-stepping or local preconditioning of the Euler equations, *AIAA* (1991) 91-1552.
[20] H. Viviand, Numerical solutions of two-dimensional reference test cases, in: Technical report, AGARD (1985) Advisory Report 211, Test Cases for Inviscid Flow Field Methods.
[21] N.P. Weatherill and B.K. Soni, Grid adaption and refinement in structured and unstructured algorithms, in: *Proc. Third Internat. Conf. on Numerical Grid Generation in CFD and related Fields, Barcelona* (1991) 143–157.