ABSTRACT

This paper presents a finite-volume method for solving the compressible, two-dimensional Euler equations using unstructured triangular meshes. The integration in time, to a steady-state solution, is performed using an explicit, multi-stage Runge-Kutta algorithm. A special treatment of the artificial viscosity along the boundaries reduces the production of numerical losses. Convergence acceleration is achieved by employing local time-stepping, implicit residual smoothing and a multigrid technique.

The use of unstructured meshes, based on Delaunay triangulation, automatically adapted to the solution, allows arbitrary geometries and complex flow features to be treated easily. The employed refinement criterion does not only detect strong shocks, but also weak flow features.

Solutions are presented for several subsonic and transonic standard test cases and cascade flows that illustrate the capability of the algorithm.

NOMENCLATURE

- \( c \) [m/s]: speed of sound
- \( D \): discretized dissipative flux
- \( e \) [J/(kg K)]: total energy
- \( \tilde{e} \): local error
- \( F, G \): flux vectors
- \( h \) [J/(kg K)]: total enthalpy
- \( I_{xx}, I_{yy}, I_{xy} \) [m²]: second order momentum
- \( k^{(2)}, k^{(4)} \): coefficients of dissipative fluxes
- \( L \): pseudo-Laplace operator
- \( l_{i,j}, l \) [m]: characteristic lengths
- \( N \): number of edges sharing a node
- \( n_x, n_y \): components of outward normal
- \( P \): forcing function
- \( p \) [Pa]: static pressure
- \( Q \): discretized flux integral
- \( R_x, R_y \) [m]: first order momentum
- \( R \) [s]: residual
- \( t \) [s]: time
- \( x, y \) [m]: cartesian coordinates
- \( U \): vector of conserved variables
- \( u, v \) [m/s]: cartesian velocity components
- \( w_n \) [m/s]: normal velocity vector
- \( w \) [-]: geometrical weight
- \( \alpha \) [m²/s]: scaling factor
- \( \Gamma \) [m²]: boundary of \( \Omega \)
- \( \gamma \) [-]: ratio of specific heats
- \( \Delta s \) [m]: length of control-volume edge
- \( \Delta t \) [s]: time-step
- \( \varepsilon \) [-]: coefficient for implicit residual smoothing
- \( \epsilon^{(2)}, \epsilon^{(4)} \) [-]: coefficients for dissipative terms
- \( \lambda_x, \lambda_y \) [1/m]: cartesian coefficients
- \( \lambda \) [-]: Courant number
- \( \nu \) [-]: pressure switch
- \( \rho \) [kg/m³]: density
- \( \Omega \) [m³]: volume of domain

Subscripts
- \( i, j \): nodes

Superscripts
- \( j \): Runge-Kutta stage
- \( m \): Jacobi iteration
- \( n \): time level

INTRODUCTION

Today there are two basic approaches to compute steady-state solutions of the Euler equations. The first, and per-
happens more traditional, involves the use of structured meshes of quadrilateral cells. The solution methods based on these meshes are highly developed and offer high accuracy at low computational costs. As long as the geometry is simple there are no difficulties in using structured meshes. To mesh solution domains with complex topologies in a structured way it is often necessary to resort to some tricks like blacking out cells or putting slits in the mesh. Another possibility is to split the solution domain into topologically more simple subdomains which can be meshed separately leading to a zonal mesh consisting of several structured local meshes linked by their boundaries. As a consequence these kind of meshes require more complex solution algorithms, which must be capable of transferring information between the various local meshes. However, even with simple topologies the structured mesh may be severely distorted, eventually leading to bad solutions.

Over the last decade an alternative approach employing completely unstructured meshes, that consist of triangular or quadrilateral cells have become widely used in research and industry. Due to their requirement for indirect addressing of data, which does not allow the best use of the architecture of vector computers, they require higher cpu-times. Nevertheless they have two desirable properties which make them superior to their structured counterparts:

- Arbitrary complex geometries can be meshed without severely distorted mesh cells.
- Solution-depending mesh-adaption can be performed in a very efficient way.

Both properties are a direct consequence of the lack of any mesh structure. Using a structured mesh, each mesh node has a fixed number of neighbours depending on its location, boundary or interior. In contrast an unstructured mesh has no such limitations thus making it easy to mesh complex geometries. On structured meshes adaption is achieved by inserting complete mesh lines or moving mesh nodes whilst preserving their connectivity. The first means inserting mesh nodes where they are not needed, the second may lead to highly distorted mesh cells. In contrast on unstructured meshes adaption is easy and efficient to perform by adding new nodes only where they are needed.

In this study a time-marching solution method for the two-dimensional Euler equations on purely triangular unstructured meshes is presented. The flow algorithm solves the time dependent integral form of the equations by means of a mesh-vertex finite volume spatial discretization. Special care is taken in the boundary treatment of artificial viscosity, needed to stabilize the scheme and to reduce oscillations near shocks. The combination of two artificial viscosity models found in the literature has let to an improved model providing lower numerical losses. Integration in time, to a steady-state solution is performed using an explicit multi-stage procedure. Convergence acceleration is achieved by employing local time-stepping, implicit residual smoothing and a multigrid technique that operates on a sequence of unrelated triangular meshes. Further acceleration can be achieved by using an edge sorting technique, which allows the scheme to run vectorized on a supercomputer like a Cray 2. Mesh adaption by means of mesh enrichment is performed iteratively, using a Delaunay triangulation algorithm in conjunction with the flow solver. The combination of mesh generator and flow solver represents a powerful tool to resolve flow features like shocks in an efficient way as some examples will show.

DISCRETIZATION OF THE GOVERNING EQUATIONS

The two-dimensional Euler equations governing time-dependent inviscid and compressible fluid flows are a nonlinear system of hyperbolic equations which can be written in conservative form as

$$\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} + \frac{\partial G}{\partial y} = 0.$$  \hspace{1cm} (1)

The conservative variable vector $U$ and the fluxes $F$ and $G$ are given by

$$U = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho e \end{pmatrix}, \quad F = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho u h \end{pmatrix}, \quad G = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho v h \end{pmatrix},$$  \hspace{1cm} (2)

where $\rho$ is the fluid's density, $u$, $v$ are the velocity components, $p$ is the pressure and $e$ is the total internal energy per unit mass. The total enthalpy per unit mass $h$ is defined as

$$h = e + \frac{p}{\rho}.$$  

By assuming that the fluid obeys the perfect gas state equation, $p$ can be calculated as

$$p = (\gamma - 1)\rho \left[ e - \frac{1}{2} (u^2 + v^2) \right],$$

$\gamma$ being the ratio between the specific heats of the fluid.

For a finite-volume spatial discretization, the computational domain is divided in a finite number of non-overlapping control volumes formed by joining the centroids of neighbouring triangles as shown in fig. 1. By integrating equation (1) over the control volume $\Omega_i$ and taking advantage of the divergence theorem, we get for each mesh node $i$

$$\int_{\Gamma_i} \frac{\partial U_i}{\partial t} d\Omega + \int_{\Gamma_i} (F_{n.x} + G_{n.y}) d\Gamma = 0,$$  \hspace{1cm} (3)

where $\Gamma_i$ is the boundary of $\Omega_i$, and $n_x$, $n_y$ are the components of the outward normal to $\Gamma_i$. The first term in (3) is discretized by assuming a constant value $U_i$ over the control volume, thus yielding

$$\int_{\Omega_i} \frac{\partial U_i}{\partial t} d\Omega = \Omega_i \frac{dU_i}{dt},$$  \hspace{1cm} (4)
The boundary integral of the second term is discretized by taking the average of the nodal flux values at the nodes $i$ and $j$ connected by a mesh edge, while it is assumed that this average is constant over the corresponding control volume edge. If we denote $\Gamma_{i,j}$ the portion of $\Gamma_i$ corresponding to a mesh edge sharing node $i$, the second term of equation (3) can be recasted as

$$\sum_{j=1}^{N} \int_{\Gamma_{i,j}} (F_{i,j} n_x + G_{i,j} n_y) d\Gamma = Q_{i} \ ,$$  

(5)

where $N$ is the number of mesh edges sharing node $i$. Equation (5) is computed in practice by means of a loop over all mesh edges in which the above mentioned flux contributions are accumulated to the appropriate nodes.

**ARTIFICIAL VISCOSITY**

In order to stabilize the above described scheme and to avoid oscillations near shocks some artificial viscosity has to be augmented by means of the standard adaptive blend of second and fourth order smoothing (Jameson et al., 1981). As proposed by Jameson et al. (1986) the artificial viscosity is constructed as two loops over mesh edges. In the first loop a pseudo-Laplacian is computed at each node providing the second order dissipation. In the second loop this pseudo-Laplacian is differenced again to produce the fourth order dissipation.

As shown by Lindquist (1988) and by Holmes and Connell (1989) numerical losses can be minimized by using a pseudo-Laplacian as close as possible to a true Laplacian. While computing a true Laplacian is expensive Holmes and Connell proposed an alternative approach using geometrical weights, writing the pseudo-Laplacian of $\phi$ for a node $i$ as

$$L(\phi)_i = \sum_{j=1}^{N} w_{j,i} (\phi_j - \phi_i) \ ,$$  

(6)

with the weights

$$w_{j,i} = 1 + \Delta w_{j,i} \ .$$  

(7)

For each edge there are two weights, one for the node at each end of the edge. Following Holmes and Connell, the $\Delta w_{j,i}$ are of the form

$$\Delta w_{j,i} = \lambda_{x,i} (x_j - x_i) + \lambda_{y,i} (y_j - y_i) \ ,$$  

(8)

where

$$\lambda_{x,i} = \frac{(I_{x,i} R_x - I_{y,i} R_y)}{(I_{x,i} I_{y,i} - I_{x,y,i}^2)} \ ,$$  

(9)

$$\lambda_{y,i} = \frac{(I_{y,i} R_y - I_{x,i} R_x)}{(I_{x,i} I_{y,i} - I_{x,y,i}^2)} \ .$$  

(10)

and

$$R_{x,i} = \sum_{j=1}^{N} (x_j - x_i) , \quad R_{y,i} = \sum_{j=1}^{N} (y_j - y_i) \ ,$$  

(11)

$$I_{x,i} = \sum_{j=1}^{N} (x_j - x_i)^2 , \quad I_{y,i} = \sum_{j=1}^{N} (y_j - y_i)^2 \ ,$$  

(12)

$$I_{x,y,i} = \sum_{j=1}^{N} (x_j - x_i)(y_j - y_i) \ .$$  

(13)

These weights are cheap to compute and guarantee that the pseudo-Laplacian of a linear function will be zero as would be the case for the true Laplacian. In the following these weights are used wherever a pseudo-Laplacian has to be computed.

The final form of the dissipative fluxes for the node $i$ in consideration of the geometrical weights is thus,

$$D_i = \sum_{j=1}^{N} \alpha_i^{(2)} w_{j,i} (U_j - U_i) - \varepsilon_i^{(4)} w_{j,i} \left( \nabla^2 U_j - \nabla^2 U_i \right) \ ,$$  

(14)

with

$$\nabla^2 U_i = \sum_{j=1}^{N} w_{j,i} (U_j - U_i) \ .$$  

(15)

As proposed by Mavripilis (1988a) the scaling is

$$\alpha_i^{(2)} = \frac{1}{2} \left( \frac{\Omega_i}{\Delta t_i} + \frac{\Omega_{\parallel}}{\Delta t_{\parallel}} \right) \ .$$  

(16)

The coefficients $\varepsilon_i^{(2)}$ and $\varepsilon_i^{(4)}$ are adapted to local pressure gradients and are defined as

$$\varepsilon_i^{(2)} = k^{(2)} \max [\nu_i, \nu_j] \ ,$$  

(17)

$$\varepsilon_i^{(4)} = \max \left[ 0, k^{(4)} - \varepsilon_i^{(2)} \right] \ .$$  

(18)
where \( k(2) \) and \( k(4) \) are empirical coefficients and the pressure switch \( \nu \) is calculated as a loop over all edges

\[
\nu_i = \frac{\sum_{j=1}^{N} w_{j,i}(p_j - p_i)}{\sum_{j=1}^{N} (p_j + p_i)}.
\] (19)

To compute the second- and fourth-order smoothing operator at a node \( i \) the first and second level neighbouring nodes are needed. Computation of eq. (14) at an inner mesh node is straightforward. Boundary nodes require a special treatment. As proposed by Ronzheimer (1988) two layers of imaginary outer mesh nodes are constructed as shown in fig. 2. The values at the outer mesh points \( i^\prime, i^\prime\prime \) are obtained by linear extrapolation along a normal through a boundary node \( i \) using the auxiliary node \( f \).

In a series of computational tests the improved artificial viscosity model obtained by the combination of Ronzheimer's boundary treatment with the geometrical weights proposed by Holmes and Connell (1989) has given the best results by means of low numerical losses.

**TIME INTEGRATION TO STEADY STATE**

In consideration of the artificial viscosity the space-discretized form of eq. (3) may be written as

\[
\frac{dU_i}{dt} = -\frac{1}{\Omega_i} [Q_i - D_i] = -R_i,
\] (20)

representing a set of coupled ordinary-differential equations where \( R_i \) is the residual error at each node. These equations are integrated in time using a fully explicit hybrid five-stage Runge-Kutta scheme, which is well suited to drive a multi-grid scheme because of its strong damping of high-frequency error modes (Mavripilis, 1988a). As shown by Kroll and Jain (1987) a five-stage Runge-Kutta scheme where the dissipative terms are calculated only in the first two steps is computationally cheaper than schemes with fewer steps. To advance one step \( \Delta t \) it is set for the node \( i \)

\[
\begin{align*}
U_i^{(0)} & = U_i^0 \\
U_i^{(1)} & = U_i^{(0)} - \lambda \alpha_1 (\Delta t / \Omega_i) [Q(U_i^{(0)}) - D(U_i^{(0)})] \\
U_i^{(2)} & = U_i^{(1)} - \lambda \alpha_2 (\Delta t / \Omega_i) [Q(U_i^{(1)}) - D(U_i^{(1)})] \\
& \vdots \\
U_i^{(5)} & = U_i^{(4)} - \lambda \alpha_5 (\Delta t / \Omega_i) [Q(U_i^{(4)}) - D(U_i^{(1)})] \\
U_i^{n+1} & = U_i^{(5)},
\end{align*}
\] (21)

where \( U_i^0 \) and \( U_i^{n+1} \) are the values at the beginning and at the end of the \( n \)th time step and \( \lambda \) represents the Courant-number. The standard values for the coefficients \( \alpha_1 \ldots \alpha_5 \) are

\[
\alpha_1 = \frac{1}{4}, \alpha_2 = \frac{1}{6}, \alpha_3 = \frac{3}{8}, \alpha_4 = \frac{1}{2}, \alpha_5 = 1.
\]

Using an explicit time-stepping scheme the magnitude of the maximum permitted time step is restricted according to the limited stability region. A decent time step limit can be written for each control volume belonging to a node \( i \) as

\[
\Delta t_i = \frac{\Omega_i}{\sum_j (|w_{nj}| + c_j) \Delta s_j},
\] (22)

where \( w_n \) and \( c \) are the normal velocity and the speed of sound on the \( j \)th edge of the control volume and \( \Delta s \) is the length of this edge.

**BOUNDARY CONDITIONS**

The boundary conditions are similar to those described by Ronzheimer (1988) and by Holmes (1989). For solid walls a weak formulation was chosen, where the fluxes normal to the wall are set to zero, except for the pressure terms in the momentum equations.

At inflow boundaries total pressure, total temperature and flow angle are prescribed. Using Riemann invariants, corresponding to an outgoing characteristic, the solution vectors at the inflow points are corrected to match the boundary conditions. For subsonic flows the calculated static pressure at the outflow boundary is merely replaced by the specified exit value.

Periodic boundaries are handled by treating the nodes on these boundaries like interior nodes. To that purpose the areas of both half control volumes, belonging to corresponding nodes on either side of the periodic boundary, are added.
and stored at both nodes. The time-step is then calculated for the full control volume. The fluxes at these boundary nodes are calculated by simply adding the contribution to both nodes.

CONVERGENCE ACCELERATION

Several acceleration techniques have been implemented. One is the use of a local time step calculated from local flow properties, enabling the scheme to operate everywhere close at its stability limit. The scheme is accelerated further by the introduction of implicit residual smoothing as proposed by Jameson et al. (1986): The residual $\tilde{R}_i$ at the node $i$ is replaced by an average of the residuals of the neighbouring nodes

$$\tilde{R}_i = R_i + \epsilon \nabla^2 \tilde{R}_i,$$  \hspace{1cm} (23)

where $\epsilon$ is a smoothing coefficient. Computing $\nabla^2 \tilde{R}_i$ in a loop over edges using the pseudo-Laplacian formulation, eq. (23) may be written as

$$\tilde{R}_i = \frac{R_i + \epsilon \sum_{j=1}^{N} \tilde{R}_j}{1 + \epsilon N}.$$  \hspace{1cm} (24)

Since the resulting matrix of residuals is diagonally-dominant for the $\epsilon$ values of interest (i.e., $\epsilon = 0.5$), this implicit system can be solved by performing several Jacobi iterations

$$\tilde{R}_i^{(m)} = \frac{R_i^{(0)} + \epsilon \sum_{j=1}^{N} \tilde{R}_j^{(m-1)}}{1 + \epsilon N}.$$  \hspace{1cm} (25)

where $R_i^{(0)}$ represents the unsmoothed residual. In practice only two iterations were used to get an adequate approximation of the smoothed residual. This smoothing technique allows the Courant number $\lambda$ in eq. (21) to be increased up to values of 7 or even higher.

In some cases a slightly quicker convergence could be achieved by using the geometrical weights proposed by Holmes and Connell (1989) leading to

$$\tilde{R}_i^{(m)} = \frac{R_i^{(0)} + \epsilon \sum_{j=1}^{N} w_{ij} \tilde{R}_j^{(m-1)}}{1 + \epsilon \sum_{j=1}^{N} w_{ij}}.$$  \hspace{1cm} (26)

An important feature of the Runge-Kutta time-stepping scheme is its capability of eliminating high-frequency errors rapidly, whereas it is slow at reducing lower-frequency errors. Since the highest-frequency errors on a particular mesh are related with its mesh width a remedy is to use a number of successively coarser meshes to eliminate a distinct bandwidth of errors on each mesh. This idea leads to the well-known multigrid techniques. In contrast to a structured approach where progressively coarser meshes can easily be obtained by amalgamating groups of fine mesh cells, the proceeding in an unstructured approach is more complicated.

One way is to generate fine meshes by repeatedly subdividing cells of a coarse mesh in some manner (Jameson and Mavriplis, 1986). A major drawback of this method is the poor topological control of the resulting fine meshes. A more promising approach is to use a sequence of completely unrelated meshes as proposed by Mavriplis (1988b). This approach, adopted in this study, enables the user to construct a number of individual, in some sense optimal meshes.

After the generation of the meshes is completed the first step in this multigrid scheme is to calculate and store the transfer coefficients for interpolation and distribution of flow variables or corrections and residuals from one mesh to the next. (For more details see Mavriplis, 1988b). The time-stepping starts at the finest mesh. After one time step flow variables and residuals are transferred to the next coarser mesh where the next time step is performed. When the coarsest mesh in the sequence has been reached, the corrections are transferred back to the finest mesh without any intermediate time stepping. An important detail in Mavriplis' multigrid scheme is the introduction of a so called forcing function.

$$P = R' - R(U'),$$  \hspace{1cm} (27)

where $R'$ and $U'$ represents the residuals and the flow variables transferred from one mesh to the next coarser one. Time-stepping on the coarser meshes is performed using a modified form of eq. (21), taking

$$U_i^{(k)} = U_i^{(k-1)} - \lambda \Delta t \left[ R_i(U_i^{(k-1)}) + P_i \right]$$  \hspace{1cm} (28)

for the $k$th stage. In the first stage on a coarse mesh, when $U_i^{(k-1)} = U_i$, the forcing function in eq. (28) replaces the calculated residuals $R(U_i^{(k-1)})$ by the transferred residuals $R'_i$. The effect of this procedure is to decouple the solutions on the various meshes as soon as the fine mesh residuals are getting small, indicating to be close to a converged solution. Since the steady-state solution on each individual mesh in the sequence would differ slightly from each other, depending on their coarseness, a scheme without decoupling would not lead to a converged solution.

All operations in the above described scheme can be coded as loops over edges, accumulating the contributions of each edge to the quantity of interest (i.e. time-step, residual) at the nodes at the ends. Since every node belongs to an undetermined number of edges vectorization of these loops would lead to recurrences in the accumulation-step. The way around this problem is to sort the edges into groups so that none of the two end nodes associated with the edge can be found more than once in a group. Here, a sorting algorithm similar to that proposed by Morgan et al. (1991) was adopted sorting the edges into groups of $n \times 64$ edges, with $n$ as high as possible. As a consequence, each loop
over edges in the above described solution scheme has to be replaced by an outer loop over groups and a vectorized inner loop over the edges in one group.

This fully vectorized treatment of the loops over edges leads to reduction of cpu time up to a factor of 2.5 on a supercomputer such as a Cray 2 and requires up to 5 times less cpu time than a fully scalar scheme, depending on the number of groups.

ADAPTATION

One of the attractive properties of the unstructured approach is that a solution-dependent mesh-adaption is easy to implement and can be performed in a very efficient way. For that purpose a flow solver and an unstructured mesh generator have to be coupled in an iterative manner. After a given number of time-steps, the solution is examined and the mesh is adapted in a suitable way before the flow solver is started again.

Today one can find many successful adaption strategies in the literature differing mainly in the adaption criterion employed and in the mesh generation algorithm. Mesh refinement, sometimes in combination with mesh de-refinement, is often found in conjunction with a Delaunay triangulation algorithm (i.e. Holmes and Connell, 1989, Basel et al., 1991). In this method the mesh is only locally adapted by adding or removing mesh points. A global mesh adaption, known as 'remeshing', can be performed by use of the advancing front method (i.e. Rick et al., 1993), generating a completely new mesh in each adaption cycle. Adaption criteria based on the examination of first or second order derivatives of one or more key-variables try to indicate the distribution of solution errors in the domain and give information where a mesh has to be refined or de-refined.

In this study a mesh enrichment technique in conjunction with a Delaunay triangulation algorithm (Irmisch and Schwolow, 1994) has been chosen. As proposed by Dannenhoffer and Baron (1985) density is used as a key-variable for mesh refinement, which allows to detect and resolve shocks as well as contact-discontinuities. After a prescribed number of time-steps the undivided difference of density along an edge is examined. The edge is flagged if the deviation of this value from the average of all density differences across the mesh is larger than the standard deviation. In the next step the centers of gravity of both triangles belonging to a flagged edge are inserted in the existing mesh using the Delaunay triangulation algorithm. Each time a boundary triangle is refined the associated boundary edge is refined as well. In order to keep the true boundary shape the new boundary point is calculated using parametric cubic splines.

As shown by Warren et al. (1991) for transonic flows over airfoils, this procedure may lead to wrong shock locations and strengths. This is caused by the dominance of strong features in the adaption criterion, preventing weak features from being detected and resolved. An easy way to avoid this problem is to multiply the undivided difference by a length scale associated with the control volume. The modified local error is written as

\[ |\tilde{e}_{i,j}| = \left( \frac{l_{i,j}}{I} \right)^{(1/r)} |\Delta \rho| \]  

(29)

where \( l_{i,j} \) represents the characteristic length scale of an edge

\[ l_{i,j} = \max \left( \sqrt{h_i}, \sqrt{h_j} \right) \]  

(30)

and \( I = \max (l_{i,j}) \) is a reference length which is the same for all the edges. For the results presented in the next chapter the parameter \( r \) ranges between 1 and 2.5, depending on the flow case. Small values of \( r \) increase the resolution of weak features, large values suppress them.

RESULTS

In the first test case the transonic flow in a symmetric turbine cascade, known as HOBSON-2 cascade (Fottner, 1990), for which an exact solution exists was analyzed. The exact, shock-free solution was obtained from a hodograph solution, where inlet and outlet Mach number is 0.575 and inlet and outlet angles are 46.123° and -46.123°. Figs. 3 and 4 show a part of the mesh consisting of 1438 nodes and 2616 triangles and the computed Mach isolines, indicating a nearly exact symmetric solution. The dotted isoline in fig. 4 represents the sonic line. Very good agreement between exact and computed pressure distribution along suction- and pressure surface is shown in fig. 5.

To give an example of the benefits of the various acceleration techniques used in the flow solver the low Mach-number flow in the test section of a wind-tunnel is considered. The complete sequence of fine and coarse meshes used in the multigrid algorithm are shown in fig. 6 together with the density isolines calculated on the finest mesh for an inlet Mach-number of 0.23. Following a suggestion of Mavriplis (1988b) the three meshes, containing 120, 509 and 2227 mesh nodes, were designed such that each coarser mesh has roughly one-quarter the number of mesh nodes as the previous mesh. Four calculations were performed using various combinations of the acceleration techniques as listed in tab. 1. The relative improvements in convergence can be assessed from fig. 7, where convergence history for the four calculations are shown. The \( L_2 \) norm of the density residuals are plotted vs the actual cpu time, normalized by the time required for the second calculation. Using implicit residual averaging allows the Courant number to be increased to 7, compared to 1 without averaging, leading to a clearly increased convergence rate as shown in fig. 7. Nearly the same speed-up was achieved by using the multigrid technique without residual averaging. Clearly, the fastest convergence rate was achieved by combination of all four acceleration techniques, being approximately five times faster than residual averaging or multigrid alone.

Another standard test case for transonic flows with shocks is the well known GAMM-test case (Rizzi and Viviani, 1981). It consists of a parallel-walled channel with a bump along
the lower wall with an inlet Mach number of 0.85. In a first attempt an unstructured mesh with the same distribution of nodes as the structured mesh proposed by Rizzi and Viviand (1981) was generated (fig. 8). The computed Mach isolines in fig. 9 show a local supersonic region with a well captured shock at its downstream end. The shock location at 85% bump length lies well within the range of shock locations obtained by several structured mesh Euler solvers (Rizzi and Viviand, 1981) ranging from 84% to 86% bump length.

The same test case was used to check the mesh-adaption technique. In a first case the parameter \( r \) in eq. (29) was given a very large value to suppress the influence of the length scale. In a second case \( r \) was set equal to 1. Fig. 10a shows the coarse starting mesh, consisting of 120 nodes, which was used in both cases. The computed flow field by means of Mach isolines is shown in fig. 11a. The meshes obtained after 7 adaption cycles are shown in figs. 10b and 10c, consisting of 1262 and 2944 nodes respectively. In the first case (figs. 10b and 11b) the mesh was only refined in the vicinity of the shock, while in the second case (figs. 10c and 11c) even weak solution gradients (i.e. the lateral density gradient behind the shock) were detected and refined. As a consequence the shock in the first case is located at 80% bump length in contrast to 86% in the second case, where a larger supersonic region was computed similar to that in fig. 9.

In the next test case the transonic flow through a highly loaded compressor cascade is analysed. For that purpose the so called "ONERA-V2 double circular arc blade" (Hoheisel, H. and Seyb, N.J., 1986), representing a near hub section of a compressor blade, was chosen. It was designed to achieve 50\(^\circ\) of deflection at an inlet Mach number of 0.85. Figs. 12 and 13 show the meshes and the calculated pressure isolines at the beginning and the end of 6 adaption cycles. The initial mesh contains only 276 nodes, the last one 1884 nodes. The original smeared shock at the end of the local supersonic region on the suction side has been resolved sharply while the supersonic region itself shows a far increased extension, reaching in front of the profile in the cascade. Again, the dotted isoline in fig. 13 represents the sonic line. The last two test cases demonstrate the ability of the scheme to give reasonable solutions even starting with a nearly arbitrary coarse mesh simply by executing some adaption cycles.

The last test case is the flow through the subsonic MTU T106 turbine cascade (Weiß and Fottner, 1993) with blunt leading and trailing edges. Inlet and outlet angles are 52.3\(^\circ\) and -63.2\(^\circ\) respectively. The theoretical outlet Mach number is 0.59. Figs. 14 and 15 show the mesh obtained after several adaption cycles, consisting of 2422 nodes and 4556 triangles and the computed pressure isolines. The close-ups in figs. 16 and 17 show the properly resolved round leading and trailing edges. Fig. 18 shows the surface distribution of static pressure for both the experimental data and the computed solution. The small discrepancies are to be attributed to the viscous effects, especially to a boundary layer separation zone found by Weiß and Fottner (1993) at the rear part of the suction side.

To demonstrate the quality of the improved artificial viscosity model three calculations for the above turbine cascade flow case were performed using the new model (model III) and the two models proposed by Holmes and Connell (1989) (model I) and by Ronzheimer (1988) (model II). For all three calculations the mesh shown in fig. 14 was used and the coefficients \( k^{(2)} \) and \( k^{(4)} \) were set to 0.6 and 1/64 respectively. As a measure for the numerical loss production, surface distributions of the ratio of computed to isentropic total pressure obtained by the three models are shown in fig. 19. While model I provides a smoother variation of the total pressures ratio the overall losses are slightly higher than that of model II (figs. 19a and 19b). Clearly, the lowest numerical losses are obtained by the new model (fig. 19c).

Having no access to a comparable structured mesh solver, no direct comparisons of required cpu-times for one of the above test cases have been performed. Results from other authors (i.e. Mavripilis, 1988b) using similar codes indicate that due to the indirect addressing of data unstructured mesh solvers are approximately three times slower than their structured counterparts.

CONCLUSION

A finite-volume method for the steady state solution of the two-dimensional Euler equations using adaptive triangular meshes has been presented. The steady-state solution is reached by integrating the unsteady form of the equations using an explicit multistage Runge-Kutta algorithm. An improved artificial viscosity model reduces the production of numerical losses. Reasonable acceleration of the scheme was achieved by using local time steps, vectorization, implicit residual smoothing and a multigrid technique. Automatic mesh adaption is performed by iteratively coupling the flow solver with a Delaunay triangulation algorithm. The undivided difference of density along mesh edges multiplied with a length scale is used as an adaption criterion. As was shown in the test cases, the use of the length scale is profitable to the calculation of correct shock-wave positions and strengths. The ability of the present method to treat subsonic and transonic flow through channels, compressor- and turbine-cascades has been demonstrated.

Future work will concentrate on the development of a method to predict compressible viscous flow through complex geometries, solving the Reynolds-averaged Navier-Stokes equations.

ACKNOWLEDGEMENT

The research project in the scope of which the reported study was performed is part of a national research cooperation "AG TURBO Arbeitsgemeinschaft Hochtemperatur Gasturbine". The project has been supported by the German Ministry of Science and Technology and the Motorenbau- und Turbinen Union München. The permission for publication is gratefully acknowledged.
REFERENCES


Dannenhoffer, J.P. and Baron, J.R., 1985, Grid Adaptation for the 2-D Euler Equations, AIAA paper No. 85-0484.


Mavripilis, D.J., 1988a, Accurate Multigrid Solution of the Euler Equations on Unstructured and Adaptive Grids, AIAA paper No. 88-3706-CP.


FIG. 3: PARTIAL VIEW OF THE MESH THROUGH HOBSON-2 TURBINE CASCADE

FIG. 4: FLOWFIELD MACH ISOLINES FOR THE TRANSONIC HOBSON-2 CASCADE
TAB. 1: COMBINATIONS OF ACCELERATION TECHNIQUES

<table>
<thead>
<tr>
<th>No.</th>
<th>vectorized</th>
<th>loc. timestep</th>
<th>residual averaging</th>
<th>multigrid</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>-</td>
</tr>
<tr>
<td>3</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>4</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

FIG. 5: COMPARISON OF COMPUTED SURFACE MACH NUMBER DISTRIBUTION WITH ANALYTICAL SOLUTION FOR THE HOBSON-2 CASCADE

FIG. 6: A) SEQUENCE OF WIND-TUNNEL MESHES USED FOR THE MULTIGRID ALGORITHM  
B) FLOWFIELDS DENSITY ISOLINES FOR THE FINEST MESH

FIG. 7: CONVERGENCE ACCELERATION AS MEASURED BY THE RMS VALUE OF DENSITY RESIDUAL
FIG. 8: GAMM-TEST CASE: ORIGINAL NODE DISTRIBUTION

A)

FIG. 9: GAMM-TEST CASE: MACH NUMBER ISOLINES

A)

FIG. 10: A) INITIAL MESH, B) ADAPTED MESH WITHOUT, C) ADAPTED MESH WITH LENGTH SCALE CONSIDERATION

B)

C)

FIG. 11: MACH ISOLINES FOR A) INITIAL MESH, B) ADAPTED MESH WITHOUT, C) ADAPTED MESH WITH LENGTH SCALE CONSIDERATION
FIG. 16: MTU-T106: CLOSE-UP OF LEADING-EDGE

FIG. 17: MTU-T106: CLOSE-UP OF TRAILING-EDGE

FIG. 18: MTU-T106: COMPARISON OF COMPUTED AND MEASURED SURFACE PRESSURE DISTRIBUTION

FIG. 19: PREDICTED SURFACE DISTRIBUTION OF TOTAL PRESSURE RATIO FOR A) MODEL I, B) MODEL II, C) MODEL III