

An Adaptive Finite Element Method for the Compressible Navier-Stokes Equations

P. Capon and P.K. Jimack

School of Computer Studies and Centre for CFD, University of Leeds.

1 Introduction

The non-dimensionalized Navier-Stokes equations for the 2-d flow of a compressible fluid may be written in non-conservative form, using primitive variables, as the system

$$A_0 \mathbf{U}_t + A_i \mathbf{U}_{,i} - (K_{ij} \mathbf{U}_{,j})_{,i} = \mathbf{F} \quad (i, j = 1, 2), \quad (1.1)$$

where $\mathbf{U} = (\rho, u, v, T)^T$, and A_0 , A_i and K_{ij} are 4×4 matrices which may depend upon \mathbf{U} (as may the source term \mathbf{F}). For external flow problems we may associate with the spatial domain, Ω , an internal rigid boundary, Γ_B , and a far-field boundary, Γ_∞ , which is divided into an inflow and an outflow boundary according to the direction of the free-stream velocity $\mathbf{u}_\infty = (\cos \alpha, \sin \alpha)^T$. In this paper we use the finite element method to find steady solutions of (1.1), with associated boundary conditions, for a variety of flows (including both transonic and supersonic cases). We then attempt to automatically assess the accuracy of these solutions so as to take advantage of mesh adaptivity. This adaptivity takes two forms, as described in section 3. Firstly we make use of conventional local refinement in those regions of the domain where the solution is not of sufficient accuracy, we then enhance this refinement by moving the nodes of the mesh as well. The paper ends with a number of numerical examples which demonstrate the effectiveness of this new adaptive procedure in terms of both reduction in error and increase in efficiency.

2 The Finite Element Discretization and Solution

The equations and boundary conditions taken here are exactly the same as in the primitive variable formulation used by Bristeau *et al* in [1]. It should be noted however that, following Hauke and Hughes [5], it is possible to express other common formulations of the Navier-Stokes equations (e.g. using conservation or entropy variables) in the form (1.1). We solve these equations using piecewise linear finite elements in space, based upon an unstructured triangulation of the domain Ω . This leads to the following system of equations for the finite element solution \mathbf{U}^h , where \mathbf{V}^h are piecewise linear test functions:

$$\int_{\Omega} (A_0 \mathbf{U}_t^h + A_i \mathbf{U}_{,i}^h) \cdot \mathbf{V}^h + \mathbf{V}_{,i}^h \cdot K_{ij} \mathbf{U}_{,j}^h - \mathbf{F} \cdot \mathbf{V}^h) d\mathbf{x}$$

$$+ \sum_e \int_{\Omega^e} \mathcal{L} \mathbf{V}^h \cdot \tau \mathcal{L} \mathbf{U}^h d\mathbf{x} - \int_{\partial\Omega} \mathbf{V}^h \cdot (K_{ij} \mathbf{U}_{,j}^h) \mathbf{n}_i ds = 0. \quad (2.1)$$

The first and last integrals in (2.1) constitute the usual Galerkin formulation. The second term consists of a sum over each of the triangular elements Ω^e and uses the least-squares operator

$$\mathcal{L} = A_0 \frac{\partial}{\partial t} + A_i \frac{\partial}{\partial x_i} - \frac{\partial}{\partial x_i} (K_{ij} \frac{\partial}{\partial x_j}) - \mathcal{F}, \quad (2.2)$$

as suggested by Shakib *et al* in [9]. This adds stability to the discretization whilst keeping it consistent. Unlike Shakib *et al*, however, we do not use space-time finite elements but instead use finite differences in time as outlined below. Other differences in this work include the lack of a discontinuity capturing term in (2.1) and a simplified definition of τ : here $\tau = \frac{h}{2} I$, where h is a mesh size parameter and I is the 4×4 identity matrix. These variations significantly reduce the cost of our computational solutions without appearing to adversely affect their quality for the flow regimes in which we are interested. In order to obtain a steady solution as efficiently as possible we use implicit time-stepping (e.g. backward Euler) so that the size of the time-steps is not restricted by any stability constraints. Hence at each implicit time-step we must solve a nonlinear system of the form

$$\mathbf{G}(\mathbf{W}) = 0, \quad (2.3)$$

where \mathbf{W} is a vector of the unknown values of (ρ, u, v, T) at the vertices of the mesh at the new time. These systems are solved using Newton's method with an exact Jacobian at each iteration. Since the Jacobians are generally non-symmetric and indefinite we use a GMRES algorithm for each linear update, with an ILU(0) preconditioner to speed up convergence [3]. Further speed-ups are also achieved by making use of local time-stepping to obtain steady solutions.

3 Adaptivity

Clearly the work required to solve the system (2.3) at each time-step will go down as the number of vertices in the grid is decreased. Conversely, the error in the finite element solution is likely to go up. Hence an ideal solution will have an optimal number of vertices in the optimal positions to allow a perfect balance between accuracy and computational expense.

3.1 Local Mesh Refinement

In a typical adaptive algorithm which uses h -refinement to obtain an accurate steady solution of (1.1), the problem is first solved on a relatively coarse unstructured grid. The error in this initial solution is then estimated and in those regions where it is large the mesh is modified by the addition of extra triangles and nodes. Typically this is done by refining elements into four children as shown in Fig.1. Note that there can be numerous levels of refinement to allow further accuracy to be obtained, and that the shape of each triangle is determined by the shape of its parent (where it has one). Further details of this refinement

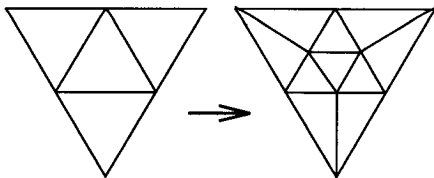


Figure 1. Local refinement of a triangular element.

algorithm can be found in [7] or [8]. Clearly, however, the approach depends critically upon the local estimate of the finite element error. Error indicators that are suitable for this purpose include the use of certain physical flow properties (such as gradients of density or velocity) or more expensive *a posteriori* error estimates. In this work however, we follow the approach of [4] and base our error indicator directly upon the steady residual

$$R_e(\mathbf{U}) = \left(\int_{\Omega^e} (A_i \mathbf{U}_{,i} - (K_{ij} \mathbf{U}_{,j})_{,i} - \mathbf{F})^2 d\mathbf{x} \right)^{\frac{1}{2}}, \quad (3.1)$$

on each element Ω^e . In practice one does not wait for a completely steady solution of (1.1) to be found before refining the mesh: it is far more efficient to allow adaptivity to take place during the local time-stepping. This approach to adaptivity has been used quite widely in recent years and is known to be reliable, provided a suitable error indicator is chosen. Moreover, the user has a degree of control over the accuracy of the final solution through the error tolerance that is used to decide whether or not to refine each element. One problem with this method however is that it is highly dependent upon the suitability of the initial mesh that is used and so, on occasions, it can be rather less efficient than one might ideally like.

3.2 Local Mesh Refinement With Node Movement

To allow greater efficiency in our finite element solution we have therefore considered an additional adaptive procedure. This relocates existing node points in a further attempt to ensure that they are positioned in those regions where the error is largest. This has the effect of uncoupling the geometry of our final solution mesh from that of the coarse initial mesh yet, unlike many other node movement strategies, we do not sacrifice the direct error control of the h -refinement approach. The exact strategy that we use to reposition the nodes is very similar to that in [6]. This involves considering the patch of elements surrounding each node in turn and then altering its position within this patch according to a formula of the following form:

$$\underline{\mathbf{x}}_{new} = \frac{1}{2} \left(\underline{\mathbf{x}}_{old} + \sum_{i=1}^m w_i \underline{\mathbf{x}}_i \right). \quad (3.2)$$

Here $\underline{\mathbf{x}}_{new}$ and $\underline{\mathbf{x}}_{old}$ are the updated and previous position vectors of the node, $\underline{\mathbf{x}}_1, \dots, \underline{\mathbf{x}}_m$ are the position vectors of the m neighbouring element centroids, and

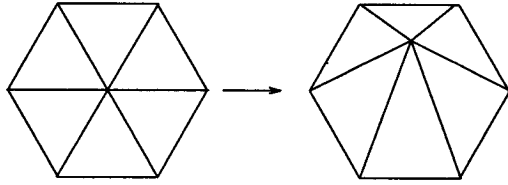


Figure 2. Local repositioning of a node.

the weights w_i are given by

$$w_i = \frac{R_i(\mathbf{U})}{\sum_{j=1}^m R_j(\mathbf{U})}. \quad (3.3)$$

Note that these weights are based upon equidistributing the same residual, (3.1), that is used as the error indicator for h -refinement. In practice the mesh is only actually updated after all of the new positions have been calculated (hence the update is independent of the nodal ordering), and for simplicity nodes on internal boundaries remain fixed. This procedure has a smoothing effect on the mesh as well as moving nodes towards regions where the error indicator is large, as shown in Fig.2. Hence elements tend to become aligned with flow properties such as shocks and boundary layers. Implementation of this method is straightforward and works in conjunction with the h -refinement by performing one iteration of mesh movement prior to refining the mesh.

4 Results and Discussion

Before solving the Navier-Stokes equations, we test the above approach on a simpler, but still nonlinear, convection-dominated problem which has a known analytic solution.

4.1 System of Burgers Equations

The 2-d system of equations,

$$\mathbf{U} \cdot \nabla \mathbf{U} - \epsilon \nabla^2 \mathbf{U} = \mathbf{F}, \quad (4.1)$$

where ϵ is a constant, has an exact known solution for a suitable choice of boundary conditions and source term \mathbf{F} :

$$\mathbf{U} = \left(\begin{array}{c} \frac{3}{4} - \frac{1}{4}(1 + \exp((-4x + 4y)/(32\epsilon))^{-1}) \\ \frac{3}{4} + \frac{1}{4}(1 + \exp((-4x + 4y)/(32\epsilon))^{-1}) \end{array} \right) \quad (4.2)$$

We solve (4.1) in a convection dominated case ($\epsilon = 0.001$) on the unit square, and so the solution consists of a sharp front along the line $y = x$. Starting with a coarse unstructured mesh consisting of 244 elements, we reach a converged solution on the final mesh by iterating the three steps: obtain solution on current mesh, calculate new node positions and refine the mesh according to element

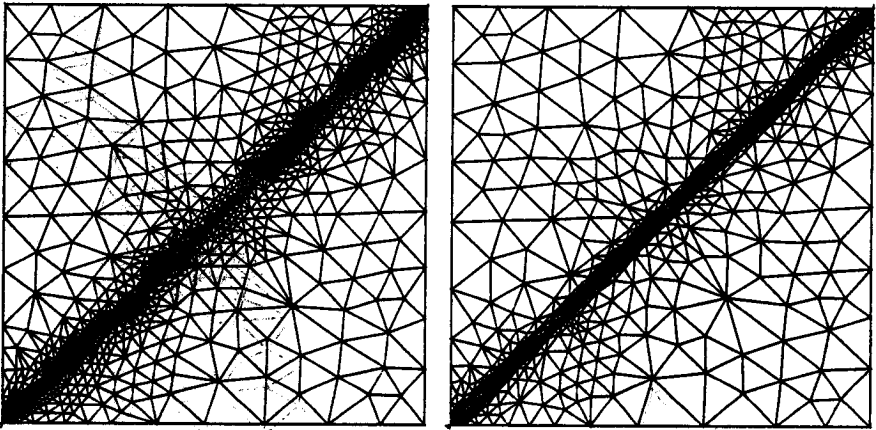


Figure 3. Final meshes for Burgers system: a) without and b) with node relocation

Table 1. Elements in final mesh and error norm for Burgers system

With node relocation?	Elements	Exact error norm
No	2792	4.30E-4
Yes	1916	3.42E-4

residuals. For comparison a solution is obtained using conventional h -refinement (without the relocation step). Fig.3 shows the final meshes that are obtained, with and without mesh relocation and Table 1 gives the number of elements in the meshes, along with the exact error norm in both cases. It is clear that not only are significantly fewer nodes needed when node relocation is being used, but the solution is more accurate (the error is reduced by more than 20%).

4.2 Supersonic Flow

As a test case for the compressible Navier-Stokes equations, we use an example considered in the GAMM workshop on the Navier-Stokes equations [2]: supersonic flow around a NACA0012 aerofoil, with a freestream Mach number, M_∞ , of 2 and Reynolds number of 500. The initial mesh contains only 166 elements, and a steady state solution is reached via local timestepping, with a node relocation/mesh refinement stage every five timesteps. Fig.4 shows a section of the final mesh, with and without the node relocation. When nodes are not repositioned, the structure of the initial mesh can still be seen to be influencing the final mesh which, on inspection, appears to be less than ideal. Allowing the nodes to move however seems to improve the quality of the mesh, with the bow shock and the boundary layer at the aerofoil both being resolved. In Table 2, the number of elements and residual norm of the final mesh are shown for the

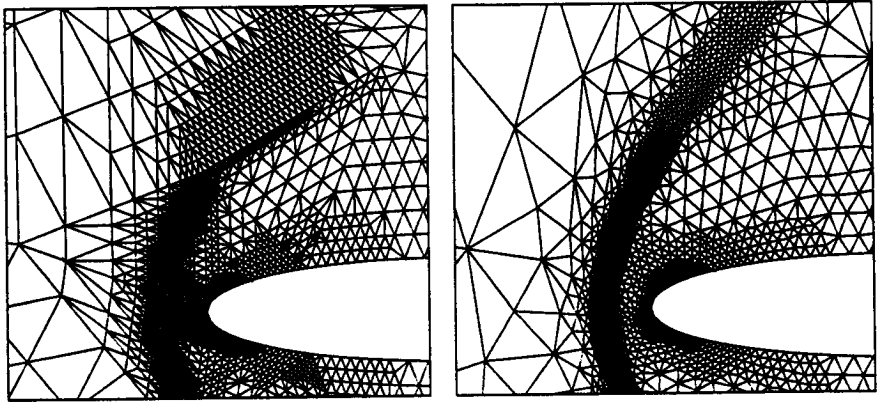


Figure 4. Final meshes for Supersonic flow: a) without and b) with node relocation

Table 2. Elements and residual norm for flows around NACA0012

With node relocation?	Elements	Final residual norm
$Re=500, M_\infty=2.0, \text{Angle of attack}, \alpha = 0^\circ$		
No	10007	6.61E-7
Yes	10456	5.52E-7
$Re=2000, M_\infty=0.85, \text{Angle of attack}, \alpha = 0^\circ$		
No	6434	6.66E-7
Yes	5871	6.07E-7
$Re=500, M_\infty=0.8, \text{Angle of attack}, \alpha = 10^\circ$		
No	6472	4.89E-7
Yes	5367	3.23E-7

above case, along with two transonic cases which also appear in [2]. In each case the approximate error has been reduced significantly and in all but one of those cases the number of elements required has also been decreased.

4.3 Further Work

Possible areas for future research include developing a mechanism for allowing nodes on the internal boundary, Γ_B , to move, the incorporation of a simple turbulence model and the use of these adaptive techniques in time-dependent problems (by generalizing the work in [7] and [8]).

Acknowledgements

We would like to thank Phil Woods, Shaun Forth, Mike Baines and Martin Berzins for their contributions to this work through a number of very useful

discussions. The first author would also like to thank British Aerospace plc and the EPSRC for their financial support in the form of a CASE studentship.

References

- [1] Bristeau, M. O., Glowinski, R., Dutto, L., Periaux, J. and Roge, G. (1990). Compressible Viscous Flow Calculations Using Compatible Finite Element Approximations. *Int. J. for Num. Methods in Fluids*, **11**, 719-749.
- [2] Bristeau, M. O., Glowinski, R., Periaux, J. and Viviand, H. (Eds.) (1987). Numerical Simulation of Numerical Navier-Stokes Flows. *A GAMM Workshop*, Freidr. Vieweg and Sohn.
- [3] Brown, P. N. and Saad, Y. (1987). Hybrid Krylov Methods for Nonlinear Systems of Equations. *Laurence Livermore National Laboratory Research Report UCLR-97645*.
- [4] Hansbo, P. and Johnson, C. (1991). Adaptive Streamline Diffusion Methods for Compressible Flow using Conservation Variables. *Comp. Meth. Appl. Mech. Eng.*, **87**, 267-280.
- [5] Hauke, G. and Hughes, T. J. R. (1994). A Unified Approach to Compressible and Incompressible Flows. *Comp. Meth. Appl. Mech. Eng.*, **113**, 389-395.
- [6] Hubbard, M. E. (1994). Grid Adaption and Multidimensional Upwinding. *Numerical Analysis Report 8/94*, University of Reading, unpublished.
- [7] Jimack, P. K. (1992). Adaptive Error Control in the Finite Element Method for Time-Dependant Problems. *School of Computer Studies Research Report 92.11*, University of Leeds, unpublished.
- [8] Jimack, P. K. (1993). A New Approach to Finite Element Error Control for Time-Dependent Problems. In *Numerical Methods for Fluid Dynamics 4* (ed. M.J. Baines and K.W. Morton), OUP, 567-573.
- [9] Shakib, F., Hughes, T. J. R. and Johan, Z. (1991). A New Finite Element Formulation for Computational Fluid Dynamics: X. The Compressible Euler and Navier-Stokes Equations. *Comp. Meth. Appl. Mech. Eng.*, **89**, 141-219.