University of Glasgow

DEPARTMENT OF

AEROSPACE ENGINEERING

An investigation of mesh sequencing and mesh coarsening for the AF-CGS method

K.J. Badcock and I.C. Glover

Glasgow University Aero report 9411

An investigation of mesh sequencing and mesh coarsening for the AF-CGS method

K.J. Badcock and I.C. Glover

Glasgow University Aero report 9411

An investigation of mesh sequencing and mesh coarsening for the AF-CGS method

K.J. Badcock and I.C.Glover
Aerospace Engineering Department
University of Glasgow,
Glasgow, G12 8QQ, U.K.

July 26, 1994

Abstract

The use of mesh sequencing as a way to speed up the convergence of the AF-CGS method is considered for the AGARD test case 9. A mesh refinement study is also carried out to determine a compromise mesh which will produce an accurate solution with fewer grid points.

1 Introduction

The AF-CGS method was developed for steady, inviscid aerofoil flows in [1] [2]. An approximate factorisation (AF) is used to provide a preconditioner for the conjugate gradient squared (CGS) iterative solver. This method has been used for unsteady pitching aerofoil flows in [3] [4] [5], and for steady, turbulent aerofoil flows in [6] [7]. The motivation behind the use of implicit methods such as AF-CGS and the method details are fully described in each of these references, but a brief outline of the method is presented here for completeness.

Using the AF-CGS code a good general flow solution for the AGARD test case 9 can be obtained in less than 2 hours on an IBM RS/6000 320H workstation, with a computational mesh of 257×65 nodes. This report considers several possible ways of reducing this time.

The first is the use of mesh sequencing, in which the starting solution on the finest mesh is obtained by iterating on a sequence of coarser meshes, and then interpolating the solution onto the next mesh. This technique has been successfully implemented in [8].

Because the AF-CGS method uses a third order approximation to the inviscid fluxes, we should be able to reduce the number of computational points without a significant change to the solution. This would lead to a reduction in the computational time and the memory requirement. Mesh coarsening is therefore investigated.

In the final section of the report conclusions are drawn.

2 AF-CGS Method

The thin-layer Navier-Stokes equations are used to model the flows in this report, and are given in Cartesian co-ordinates by

$$\frac{\partial \mathbf{w}}{\partial t} + \frac{\partial \mathbf{f}}{\partial x} + \frac{\partial \mathbf{g}}{\partial y} = \frac{\partial \mathbf{s}}{\partial y} \tag{1}$$

where

$$\mathbf{w} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ e \end{bmatrix}, \mathbf{f} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(e+p) \end{bmatrix}, \mathbf{g} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(e+p) \end{bmatrix}$$

$$\mathbf{s} = \begin{bmatrix} 0 \\ \sigma_{xy} \\ \sigma_{yy} \\ u\sigma_{xy} + v\sigma_{yy} - q_y \end{bmatrix}.$$

Here,

$$\sigma_{yy} = 2\mu v_y - \frac{2}{3}\mu(u_x + v_y),\ \sigma_{xy} = \sigma_{yx} = \mu(u_y + v_x),$$

$$q_y = -\kappa \frac{\partial T}{\partial y},\ p = (\gamma - 1)(e - \frac{1}{2}\rho(u^2 + v^2)),$$

$$T = c_v(\frac{e}{\rho} - \frac{1}{2}(u^2 + v^2)).$$

The method is developed in Cartesian coordinates for ease of presentation, but it is noted that generalisation to a curvilinear coordinate system is straightforward. The symbols ρ, u, v, e, p, μ, κ, T represent the fluid density, the two components of velocity, energy, pressure, viscosity, heat conductivity and temperature respectively. The constants γ and c_v stand for the ratio of the specific heats and the specific heat at constant volume respectively. The fluid viscosity is assumed to vary with temperature by Sutherland's law. The Baldwin-Lomax model is used to provide a contribution to the viscosity from turbulence.

The approximate Riemann Solvers due to Osher [9] and Roe [10] have proved to be successful for the computation of viscous transonic flows due to properties of their numerical dissipation. High order versions of these schemes are dissipative enough around shocks to damp spurious oscillations but the dissipation present in boundary layers is small allowing for accurate resolution [11] [12]. Osher's flux approximation has the property of being differentiable which is desirable for the implicit formulation discussed below and hence shall be used. High order accuracy is provided by a MUSCL interpolation limited by Von Albada's limiter [13]. Characteristic far field conditions are used and the temperature is imposed along with no-slip conditions on the aerofoil.

Solving the unsteady equation (1) to steady state by time-stepping is generally considered a reliable way of obtaining the solution to the steady form of (1). In this section we develop an unfactored implicit method which is a variant of an algorithm for the unsteady equations discussed in [3].

To illustrate the basic concepts write one implicit step as

$$(\frac{\partial \mathbf{c}}{\partial \mathcal{P}} + \mathcal{D}\frac{\partial \mathbf{R}_x^\mu}{\partial \mathcal{P}} + \mathcal{D}\frac{\partial \mathbf{R}_y^\mu}{\partial \mathcal{P}})\delta\mathcal{P} = -\mathcal{D}(\mathbf{R}_x + \mathbf{R}_y) \qquad (2)$$

where $\mathbf{c} = (\rho, \rho u, \rho v, e)^T$ is the vector of conservative variables and $\mathcal{P} = (\rho, u, v, p)^T$ is the vector of primitive variables. Here the term $\mathcal{D}$ denotes a diagonal matrix of local time steps and the matrices $\partial \mathbf{R}_x^\mu / \partial \mathcal{P}$ and $\partial \mathbf{R}_y^\mu / \partial \mathcal{P}$ account for the time linearisation of the right hand side except that the turbulent viscosity term is not linearised i.e. it is unaccounted for on the left hand side of (2). This doesn't adversely affect the stability properties of the method in practice and in the following we shall drop the superscript μ for simplicity of notation. The updates are written in terms of primitive variables as opposed to conservative variables [3] because the accurate resolution of moving shockwaves is not required for steady solutions and the calculation of the linearisation matrix of $\mathbf{R}_x$ and $\mathbf{R}_y$ proves more efficient with respect to $\mathcal{P}$ than $\mathbf{c}$.

The almost universal way of dealing with (2) is to approximately factor the matrix on the left hand side of (2) into three block diagonal matrices, namely

$$(\frac{\partial \mathbf{c}}{\partial \mathcal{P}} + \mathcal{D}\frac{\partial \mathbf{R}_x}{\partial \mathcal{P}})(\frac{\partial \mathbf{c}}{\partial \mathcal{P}})^{-1}(\frac{\partial \mathbf{c}}{\partial \mathcal{P}} + \mathcal{D}\frac{\partial \mathbf{R}_y}{\partial \mathcal{P}}). \tag{3}$$

The factored system of equations can be efficiently solved at the cost of incurring an error in the solution of (2), which introduces a stability limit on the time step and is detrimental to the convergence rate of the iteration to the flow steady state. We therefore adopt an alternative approach involving the solution of the unfactored linear system (2) to a prescribed tolerance by a preconditioned conjugate gradient method.

Conjugate Gradient methods find an approximation to the solution of a linear system by minimising the error in a finite dimensional space. Several algorithms are available including BiCG, CGSTAB, CGS and GMRES. These methods were tested in [14] and it was concluded that the choice of method is not as crucial as the preconditioning. However, the CGS method was found to be the quickest of the three methods that do not use re-orthogonalisation and shall be used below. It has the additional advantage that the transpose of the matrix on the left hand side of the linear system is not required, reducing implementation difficulties. The CGS algorithm was derived in [15] and is restated in [1].

Successful conjugate gradient solutions need good preconditioning. Incomplete LU decomposition (ILU) has been successfully applied for steady fluid flow problems [16] [14] but is expensive to compute. An alternative for the present time stepping approach is to use an approximate factorisation to provide the preconditioner. The ADI factorisation was used in [1] to speed convergence to the steady state for inviscid aerofoil problems.

Denoting the linear system to be solved at each time step by

$$A\mathbf{x} = \mathbf{b} \tag{4}$$

we seek an approximation, C^{-1}, to A^{-1} which yields a system

$$C^{-1}A\mathbf{x} = C^{-1}\mathbf{b} \tag{5}$$

more amenable to conjugate gradient methods. The ADI method gives a fast way of calculating an approximate solution to (4) or, restating this, of forming the matrix vector product

$$C^{-1}\mathbf{b} = \mathbf{x}. \tag{6}$$

Hence, if we use the inverse of the ADI factorisation as the preconditioner then multiplying a vector by the preconditioner can be achieved simply by solving a linear system with the right-hand side given by the multiplicand and the left hand side given the approximate factorisation. The factors in C can be diagonalised once at each time step with the row operations being stored for use at each multiplication by the preconditioner.

The exact form of the algorithm for one step of the Navier-Stokes solution is

- calculate matrices and put ADI factors in Hessenberg form
- calculate updated solution by ADI
- use this solution as starting solution for AF-CGS
- perform AF-CGS iterations until (4) has been solved to required tolerance

3 Terminology

In the following two sections the fine mesh used has 257×65 nodes in the streamwise and normal directions respectively and is referred to as grid 3. A 129×33 grid (grid 2) is obtained from grid 3 by deleting every second point in both the x and y directions, and a 65×17 grid (grid 1) is obtained from grid 2 by the same technique.

We define level 1 convergence to be the reduction of the relative residual by 2.5 orders from freestream using the single precision version of the code, and level 2 convergence to be the reduction of the relative residual by 4 orders from freestream using the double precision version of the code. After level 2 convergence has been obtained, the integrated coefficients remain unchanged as the code converges further. Level 1 convergence provides a good general flow solution.

Convergence times are given in terms of a work unit, which is defined as the time for 1 explicit step on grid 3.

4 Mesh sequencing

We consider the computation of turbulent, transonic flow over an RAE2822 aerofoil. The flow conditions for this test case, also known as AGARD case 9 are given by

$$M_\infty = 0.73, \quad \alpha = 2.79^o, \quad \mathrm{Re} = 6.5 \times 10^6.$$

The AF-CGS method can be used to obtain level 1 convergence for this test case in around 1800 work units on grid 3. The explicit method is run for 400 iterations to smooth the freestream data before switching to AF-CGS with a global CFL number of 35.

An alternative way of generating a starting solution for the AF-CGS method on this grid is to use mesh sequencing [8]. A solution is obtained by solving on the coarsest grid, and this solution is then interpolated onto the second coarsest grid, and so on until the finest grid is reached. We carry out this mesh sequencing using grid 3, grid 2 and grid 1. In table 1, the number of work units required to obtain level 1 convergence on grid 3 with mesh sequencing is shown for several levels of intermediate convergence on the two coarser meshes in the sequencing. The explicit method is run for 150 iterations from freestream on grid 1 before switching to AF-CGS with a

Number of orders reduction on grid 1	Number of orders reduction on grid 2	Work units
1	1	900
1.5	1.5	740
2	2	360
2	2.5	400

Table 1: *Work units to convergence using mesh sequencing.*

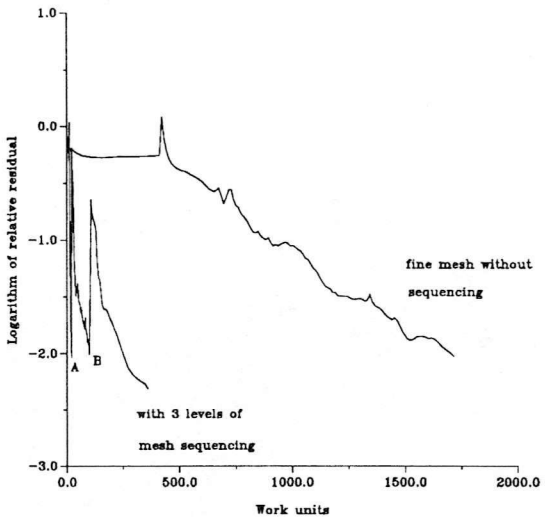

Figure 1: *Comparison of convergence histories with and without mesh sequencing.*

global CFL number of 35. The solution on grid 1 is then interpolated onto grid 2, and is used as the starting solution for the AF-CGS method at a CFL number of 35 on this finer grid. This is repeated from grid 2 onto grid 3 before the final solution is obtained. Table 1 shows that convergence can be obtained in 360 work units, which represents a speed up by a factor of 5 compared with the results that were obtained without mesh sequencing. The convergence histories for the AF-CGS method with and without mesh sequencing are shown in figure 1. At point A, the solution is interpolated to grid 2, and at point B the solution is interpolated to grid 3.

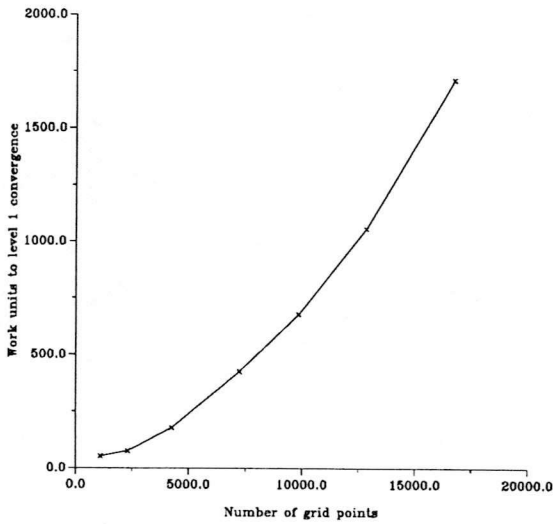

Figure 2: *Number of work units required to reach level 1 convergence for varying grid size. A work unit is defined as the time taken for one explicit iteration on the finest mesh.*

5 Mesh coarsening

The AF-CGS method uses third order accurate approximations to the inviscid fluxes, so it should be possible to obtain an accurate solution using a smaller number of mesh points. A mesh refinement study for the AGARD test case 9 has been carried out in [17]. Grids 2 and 3 were considered, along with a number of grids whose dimensions lie between those of grids 2 and 3, and were obtained by removing every other point from specified regions of grid 3. Unfortunately this is a procedure which is far from ideal, as it can lead to a poor quality mesh when large numbers of points are removed. Figure 2 shows the work units required to reach level 1 convergence on a variety of meshes.

Figure 2 shows that a speed up by a factor of 2.5 can be obtained using a 193×51 mesh. Of course the accuracy of the computed solution is a major issue, and this is examined in figure 3 which shows the value of the lift coefficient on the various grids when level 2 convergence is reached. The horizontal dotted lines represent those values of the lift coefficient which lie within 0.5% of the value obtained on grid 3. The value for the 193×51 mesh lies within this band, and the plots of the computed pressure distributions on these two meshes [17] are virtually identical. Hence we can obtain an almost identical solution to that on grid 3 using just over half of the points and around 40% of the CPU time.

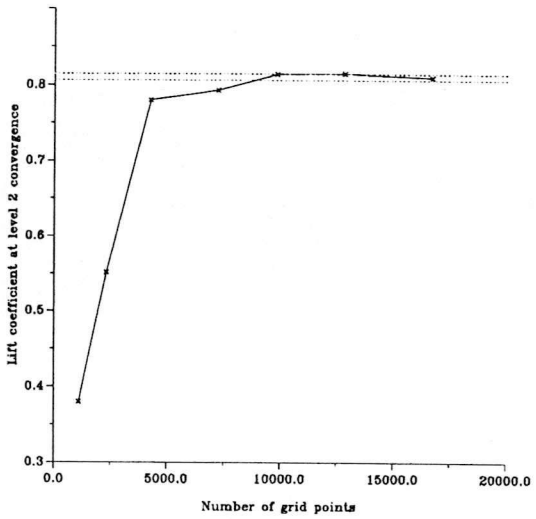

Figure 3: *Comparison of convergence histories with, and without, mesh sequencing.*

We now repeat the mesh sequencing of the previous section by modifying the 193×51 grid slightly so that there are 53 lines in the y direction. We can then define grids of 97×27 nodes and 49×14 nodes. In this way we can obtain level 1 convergence in the equivalent of 250 work units on grid 3. If we could achieve the same speed up as we obtained for grid 3 via mesh sequencing, we would expect this figure to be closer to 140 work units. However, it has already been noted that the algorithm used to define the coarser grids can result in a poor quality grid, which would explain the reduced speed up by mesh sequencing.

Mesh sequencing on the 193×53 mesh gives a solution which is very close to that obtained on grid 3, and in about one seventh of the time.

6 Conclusions

The use of mesh sequencing has produced a speed up by a factor of 5 over the previous results. A mesh refinement study has shown that we can obtain a speed up by a factor of 2.5 without a significant change to the computed solution. Applying both of these techniques would suggest a speed up by at least an order of magnitude. However, the algorithm for producing the coarser meshes can reduce the mesh quality by leaving 'holes' in parts of the mesh. The speed up achieved by implementing both of these techniques was thus found to be by a factor of 7. This corresponds to obtaining a good general flow solution in around 15 minutes on an IBM

RS/6000 320H. However, if we generated a good quality mesh of 193×53 nodes from scratch this time could be further reduced.

References

[1] M. Vitaletti. Solver for unfactored schemes. *AIAA J.*, 29:1003–1005, 1991.

[2] S. Paoletti, M. Vitaletti, and P. Stow. An unfactored implicit scheme for 3d inviscid transonic flows. *A.I.A.A. Paper*, 1992.

[3] K.J. Badcock. An efficient unfactored implicit method for unsteady aerofoil flows. Technical report, G.U. Aero report 9313, 1993.

[4] K.J. Badcock. Computation of turbulent pitching aerofoil flows. Technical report, G.U. Aero report 9322, 1993.

[5] K.J. Badcock. AF-CGS-P:an unfactored method in parallel for turbulent pitching aerofoil flows. Technical report, G.U. Aero report 9323, 1993.

[6] K.J. Badcock, I.C. Glover, and B.E. Richards. Fast and accurate unfactored two-dimensional turbulent flow simulation. Technical report, G.U. Aero report 9326, 1993.

[7] I.C. Glover. An investigation into the convergence of the AF-CGS method. Technical report, G.U. Aero report 9327, 1993.

[8] D. Drikakis and S. Tsangaris. Local solution acceleration method for the Euler and Navier-Stokes equations. *A.I.A.A. Journal*, 30:340–348, 1992.

[9] S. Osher and S.R. Chakravarthy. Upwind schemes and boundary conditions with applications to Euler equations in general coordinates. *J. Comp. Phys.*, 50:447–481, 1983.

[10] P.L. Roe. Approximate Riemann solvers , parameter vectors and difference schemes. *J. Comput. Phys.*, 43:357–372, 1981.

[11] N. Qin. A comparative study of two upwind schemes as applied to Navier-Stokes solutions for resolving boundary layers in hypersonic viscous flows. Technical report, G.U. Aero report 9120, 1991.

[12] P.L. Roe. Finite volume methods for the compressible navier–stokes equations. In *Conf. on Numerical Methods for Laminar and Turbulent flows , Montreal*, pages 2088–2101, 1987.

[13] W.K. Anderson, J.L. Thomas, and B. Van Leer. A comparison of finite volume flux vector splittings for the Euler equations. *A.I.A.A. Journal*, 24:1453–1460, 1986.

[14] K.J .Badcock. Newton's method for laminar aerofoil flows. Technical report, G.U. Aero report 9310, 1993.

[15] P. Sonneveld. CGS: A fast Lanczos-type solver for nonsymmetric linear systems. *SIAM J. Stat. Comp.*, 10:36–52, 1989.

[16] V. Venkatakrishnan. Preconditioned conjugate gradient methods for the compressible Navier-Stokes equations. *A.I.A.A. J.*, 29:1092–1100, 1990.

[17] S.M. Keith. Fourth year project. Technical report, Glasgow University Department of Aerospace Engineering, 1994.