University of Glasgow

DEPARTMENT OF

AEROSPACE ENGINEERING

Fast and Accurate Unfactored Two-Dimensional Turbulent Flow Simulation

K.J. Badcock, I.C. Glover and B.E. Richards

Glasgow University Aero report 9326

Fast and Accurate
Unfactored Two-Dimensional
Turbulent Flow Simulation

K.J. Badcock, I.C. Glover and B.E. Richards

Glasgow University Aero report 9326

Fast and Accurate Unfactored Two-Dimensional Turbulent Flow Simulation

K.J. Badcock*, I.C. Glover† and B.E. Richards
Aerospace Engineering Department
University of Glasgow,
Glasgow, G12 8QQ, U.K.

October 28, 1993

Abstract

The approximate factorisation-conjugate gradient squared (AF-CGS) method has been successfully demonstrated for unsteady turbulent flows. In the present paper the method is adapted to obtain rapid convergence for steady flows. Modifications to the original method are described and test results are given for a laminar subsonic flow and two turbulent supersonic flows including the well studied AGARD RAE2822 case 9.

*supported by SERC contract
†supported by MoD and British Aerospace

Contents

1 Introduction

The Navier-Stokes equations are being used with increasing frequency to model flow problems in the aerospace industry. This is well illustrated by the growth during the last ten years in aeroelastic studies which take the Navier-Stokes equations as the aerodynamic model. Due to the complicated nonlinear nature of the equations and to the variety of length scales in the problem the computational effort required in this type of simulation is still formidable. The development of efficient algorithms still has an important role to play as computational fluid dynamics (CFD) is required to tackle more complex flows.

A large number of codes which solve the Navier-Stokes equations are based on implicit time differencing and approximate factorisation. This method has met with considerable success and it is unnecessary to document the achievements of this approach here. However, we note that with the development of fast iterative methods such as the conjugate gradient methods an alternative to approximate factorisation is becoming available. In the present paper we continue the development of an unfactored fully implicit method and a brief review of this approach is given below.

A variety of techniques relating to the calculation of the Jacobian and the solution of the linear system have been developed in [1] and subsequent publications. For generality, finite-difference approximations were developed for the Jacobian calculation and the refinement of this fully discrete calculation has eventually led to a fast analytic method which makes use of the chain rule and a symbolic algebra package. These developments are discussed in the next section.

The problem of solving the linear system was tackled in [2] where GMRES was used together with a preconditioning strategy which was designed to be easily parallelisable. A block diagonal preconditioner was employed together with a diagonal damping factor to obtain fast convergence of the GMRES solution. An outer iterative loop was used to regain the solution to the original system. These techniques were all applied to a hypersonic cone test problem and fast convergence was noted when compared to an explicit local time-stepping method. More recently a block incomplete LU decomposition (ILU) preconditioner was found to give substantial improvements in performance [3].

Newton's method was applied to the Navier-Stokes equations for aerofoil flows in [4] [5]. The scheme used an implicit time-stepping procedure equivalent to Newton's method with a damping factor inversely proportional to the time step and the analytical Jacobian was calculated. A sparse, direct solver was applied and mesh sequencing was found to speed up the rate of convergence with coarse grid solutions being used to provide initial solutions on finer grids. Tests were performed on flows over NACA 0012 aerofoils and the method was found to have the robustness of explicit methods whilst retaining fast convergence once a sufficiently good solution had been reached.

In [6] the use of the iterative solver GMRES along with ILU preconditioning was investigated with the schemes of [4] [5] . A Baldwin-Lomax turbulence model was used although its contribution to the Jacobian was neglected. The ILU preconditioner was found to work well for test cases consisting of inviscid transonic flow over an aerofoil, laminar viscous subsonic flow over a NACA 0012 aerofoil at a Reynold's number of 5000 and transonic turbulent flow over an RAE2822 aerofoil at a Reynolds number of 6.5×10^6.

The calculation of the Jacobian represents an obstacle to the successful application of implicit methods. An efficient finite difference approach was used in [7] and results were obtained for transonic compressible flow over an Onera M6 wing and for incompressible flow around a sphere, over a flat plate and around a ship's hull. A direct way of tackling the complexity of the Jacobian calculation was used in [8] where the symbolic manipulation package MACSYMA was used to calculate analytical expressions for the derivatives which were then output directly to FORTRAN code. The method was tested on flat plate and wedge flows.

An interesting application of Newton's method has been to the study of non-uniqueness for the solutions of the nonlinear algebraic discrete equations arising from CFD. In [9] a number of solutions were obtained for the Euler and Navier-Stokes equations and their stability was examined. The problems studied were inviscid and viscous flow in a nozzle, flows over a NACA 0012 aerofoil and flows around a cylinder. The linear systems obtained were all small enough to allow direct solution although the comment is made that sparse matrix solvers should be more widely utilised in CFD. Similar non-uniqueness effects were noted in [10] for solutions of the characteristic form of the Euler equations for nozzle flows.

The undamped form of Newton's method is inappropriate for turbulent flows modelled by the Baldwin-Lomax model due to the large stencil which this results in. However, implicit time stepping can be interpreted as a damped form of Newton's method and so can still give fast convergence even if quadratic convergence is lost. The turbulent viscosity terms can be treated explicitly in this modified approach without significantly compromising the stability properties of the overall scheme. This also opens up new possibilities for preconditioners which were examined for unsteady aerofoil flows in [11] [12] [13]. This work uses an approximate factorisation (AF) to provide a preconditioner for the conjugate gradient squared (CGS) iterative solver, combining the advantages of the two approaches. The present paper adapts the AF-CGS method, which was previously developed in the above references for unsteady rigid and pitching aerofoil flows, to obtain fast convergence to steady flow problems. Test results for turbulent and laminar aerofoil flows are presented to demonstrate the utility of the method.

The rest of the paper is organised as follows. First the method details are described. Next a brief discussion of convergence criteria is given and finally results are presented for three test flows and conclusions are drawn.

2 AF-CGS Method

The flows of interest are described by the thin-layer Navier-Stokes equations 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

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

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

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 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 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 [14] and Roe [15] have proved to be successful for the computation of viscous transonic flows. This is due to the properties of the numerical dissipation of these methods. 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. In the present work we use Osher's scheme for the spatial discretisation. High order accuracy is provided by a MUSCL interpolation limited by Von Albada's limiter. Characteristic far field conditions are used and the temperature is imposed along with no-slip conditions on the aerofoil.

The main focus of this paper is on the solution method for the nonlinear discrete system of equations obtained for the discretisation of the steady form of equation 1. Solving the unsteady equation to steady state by time-stepping is generally considered a reliable way of obtaining this solution. In this section we develop an unfactored implicit method which is a variant of an algorithm for the unsteady equations discussed in [11].

To illustrate the basic concepts write one implicit step as

$$(\frac{\partial c}{\partial p} + \Delta t\frac{\partial R_x^\mu}{\partial p} + \Delta t\frac{\partial R_y^\mu}{\partial p})\delta p = -\Delta t(R_x + R_y) \tag{2}$$

where $c = (\rho, \rho u, \rho v, e)^T$ is the vector of conservative variables and $p = (\rho, u, v, p)^T$ is the vector of primitive variables. Here the term Δt denotes a diagonal matrix of local time steps and the matrices $\partial R_x^\mu/\partial p$ and $\partial R_y^\mu/\partial p$ 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 in contrast to the formulation in [11] because the accurate resolution of moving shockwaves is not required for steady solutions and because the calculation of the linearisation matrix of R_x and R_y proves more efficient with respect to p than c.

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

$$(\frac{\partial c}{\partial p} + \Delta t\frac{\partial R_x}{\partial p} + \Delta t\frac{\partial R_y}{\partial p}) \approx (\frac{\partial c}{\partial p} + \Delta t\frac{\partial R_x}{\partial p})(\frac{\partial c}{\partial p})^{-1}(\frac{\partial c}{\partial p} + \Delta t\frac{\partial R_y}{\partial p}). \tag{3}$$

The factored system of equations can be efficiently solved at the cost of incurring an error in the solution of 2. This error 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. The method details are described below. First, the matrix generation details are considered.

The matrix on the left hand side of 2 involves derivatives of complicated functions and considerable computational effort is expended in computing them. Various approaches have been adopted to overcome the complexity of the expressions involved. The first method used was a full finite difference approximation to the derivatives [1]. This required 100 evaluations of the explicit residual to calculate the matrix but this disadvantage was offset by the generality of the approach and the ease of programming. An improvement in efficiency was obtained in [16] where the number of evaluations was reduced leading to a calculation time equivalent to twenty explicit residual computations.

The problem with the fully analytic method is the complexity of the derivatives of the Osher approximations to the fluxes. These derivatives were still evaluated numerically in [17] but analytic expressions were used for the derivatives of the MUSCL interpolation and the chain rule was used to provide the required terms in the matrix. Analytic evaluation of the viscous terms in the matrix led to a considerable speed up due to the expense of calculating the power functions involved in the expressions for temperature and viscosity. The overall calculation took around 10 explicit evaluations.

Symbolic manipulation codes can be used to overcome algebraic problems with evaluating analytic expressions for derivatives of complicated functions. The package REDUCE was used in [18] to calculate the derivative of the 3-D Osher's flux function to incorporate in a scheme similar to that given in [17]. This fully analytic calculation required equivalent CPU time of 2 explicit evaluations.

The present work uses a fully analytic calculation which takes around 3 explicit evaluations. The package REDUCE is used to calculate the derivatives but it is also used along with the optimisation package SCOPE to produce optimised FORTRAN code. Problems were encountered with the optimiser which occasionally did not produce correct code on optimisation but careful comparison with unoptimised code allowed the identification of rogue terms. The calculation of the linearisation takes up around sixty-eight per-cent of the CPU time at each step. The improvement in efficiency of this part of the code from the fully discrete method of 1988 has lead to an improvement in the overall operating speed by a factor of 10.7. Without this improvement the code would not be competitive. The comparison of the various approaches is shown in table 1.

method	variables	reference	dimensions	time for Jacobian
full FD	primitive	[1]	2	100
full FD version 2	primitive	[16]	3	20
mixed analytic/FD	conservative	[17]	2	10
full analytic	primitive	[18]	3	2
full analytic	primitive	present	2	3.25

Table 1: *Times for linearisation of discrete fluxes scaled by the time for one explicit step using the same flux calculation.*

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 [17] 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 [19] and is restated in [20].

Successful conjugate gradient methods need good preconditioning. Incomplete LU decomposition (ILU) has been successfully applied for steady fluid flow problems [17] [6]. However, the ILU decomposition 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 [20] 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 to $A^{-1} \approx C^{-1}$ 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 method 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 diagonalise ADI factors
- 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 Convergence criteria

There are a variety of convergence criteria that may be used to determine when the steady state is reached. The reduction of the residual to machine zero is one possibility. This is going to be somewhat inefficient as it is likely to require far more time than is necessary to obtain an accurate solution. A straightforward modification is to terminate the solution procedure when the logarithm of the cell or nodal residual has been reduced below a given tolerance. For example, the reduction of the logarithm of the nodal residual below -5 was used as the convergence criterion in [21] for transonic flow in a bumpy channel. This type of convergence measure tends to correspond to a reduction in the residual by 2 or 3 orders from the residual of the initial guess. It is possible to relax the convergence criterion further still. Several convergence criteria were considered in [22], the most stringent being a similar measure to that in [21]. The other convergence criteria considered in [22] were the number of supersonic points becoming fixed, and the lift coefficient being within either 1 per cent, or 0.1 per cent of its final value. The last two of these are really dependent on running the code to a much higher level of convergence, and then going back to decide where this cut off point can occur. For transonic problems a count of the number of supersonic points is a useful criterion, but not for subsonic or hypersonic problems.

We would like to use a convergence criterion that is valid for a wide variety of flow types, gives an accurate solution, and allows the solution procedure to be terminated without the need to go on for a larger number of iterations before checking back to see when it could have been terminated. The convergence criterion that we shall use in all of the test cases is the reduction of the relative residual by two orders from freestream.

This is observed to give an accurate result for the cases considered, and corresponds to the actual residual being reduced to the order of 10^{-4}. It is noted that more work is required to provide stopping criteria which are not based on the experience and intuition of the operator.

4 Numerical Results

We now present three examples to demonstrate the fast convergence to an accurate steady state solution that can be achieved using the present method. These three examples are for laminar flow over a symmetric aerofoil, turbulent flow over a symmetric aerofoil, and a more complicated turbulent flow over an unsymmetric aerofoil. Convergence comparisons are made with a local time-stepping explicit method.

4.1 Laminar flow over a NACA0012 aerofoil

The flow conditions are given by

$$M_\infty = 0.5, \ \alpha = 0, \ \mathrm{Re} = 5.0 \times 10^3, \ 128 \times 32 \text{ mesh}.$$

The explicit method alone reduces the relative residual by only one order from freestream after 5000 iterations, requiring 3700 seconds of CPU time on a SPARC10. The AF-CGS method is run from freestream, and the optimal convergence time is 540 seconds on a SPARC10, with a local CFL number of 10. The AF-CGS scheme does not converge if the local CFL number is larger than 10. A plot of the computed pressure coefficient is shown in Figure 1 and is consistent with the results of [5] [23]. A comparison between the convergence rates of the explicit method and the present method is shown in Figure 2, which illustrates how much faster the AF-CGS method achieves convergence. The AF-CGS method has converged in one seventh of the time required by the explicit method to reduce the relative residual by just one order from freestream.

4.2 Turbulent flow over a NACA64A010 aerofoil

The flow conditions are given by

$$M_\infty = 0.796, \ \alpha = 0, \ \mathrm{Re} = 12.56 \times 10^6, \ 70 \times 32 \text{ mesh}.$$

The explicit method alone requires 7642 iterations, and 3400 seconds of CPU time on a SPARC10, to converge. The AF-CGS code cannot be run directly from freestream in this case, unless the local CFL number is taken

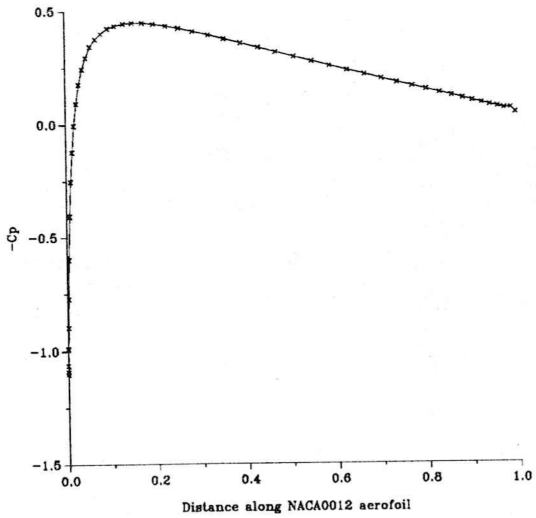

Figure 1: *The pressure coefficient for laminar flow over a NACA0012 aerofoil.*

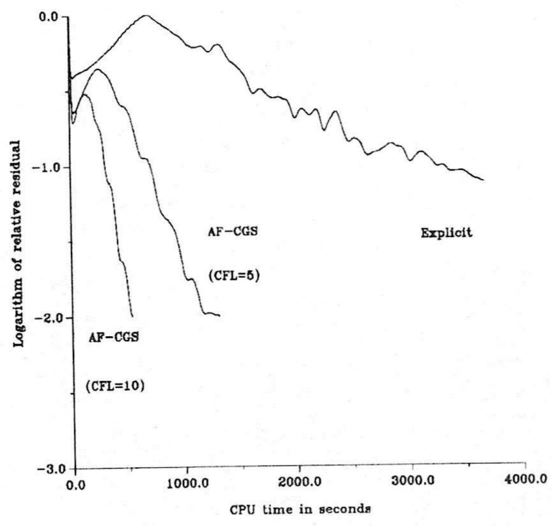

Figure 2: *Comparison of convergence rates for laminar flow over a NACA0012 aerofoil.*

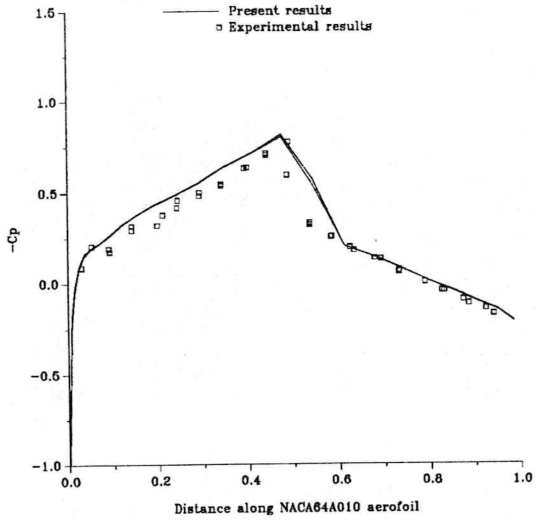

Figure 3: *Comparison of numerical results, and experimental results of [24], for turbulent flow over a NACA64A010 aerofoil.*

to be less than 2. A switch to a higher value of the local CFL number can still be made after 15 – 20 iterations. However, there are difficulties with the CGS solver taking several iterations to converge to the solution of the linear system at each of these starting steps. The explicit method provides a quick and simple way of overcoming the starting problems. For this test case, 150 iterations of the explicit method from freestream smooth the solution sufficiently for AF-CGS to be used thereafter. The optimal convergence time is obtained with a local CFL number of 30 after the switch to AF-CGS. This time, including the 150 explicit starting steps, is 300 seconds on a SPARC10. A plot of the pressure coefficient on the aerofoil surface for the present results, and also for experiment, is shown in Figure 3. A comparison of the convergence rates for the explicit method, and for AF-CGS with a local CFL number of 30, is shown in Figure 4, showing the much improved convergence rate of the present method. A comparison of the time for AF-CGS to convergence for varying values of the local CFL number is shown in Figure 5. The convergence time decreases up to a local CFL number of 30, before increasing again. When the local CFL number is reduced below 30, the convergence time increases because the number of AF-CGS steps required to achieve convergence increases. When the local CFL number is increased to 35, the number of AF-CGS steps required for convergence decreases, but the time increases

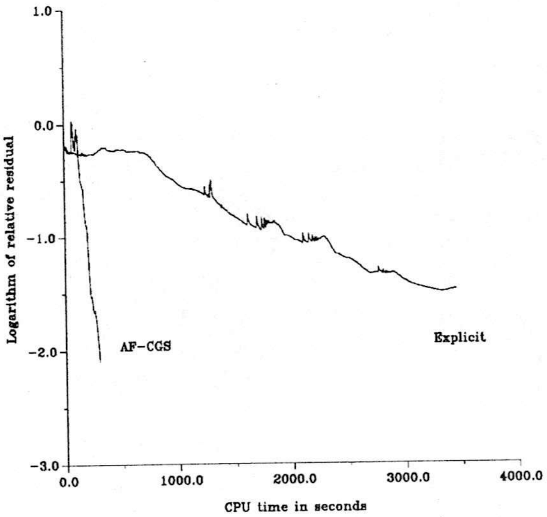

Figure 4: *Comparison of convergence rates for the explicit method alone, and AF-CGS with a local CFL number of* 30, *for turbulent flow over a NACA64A010 aerofoil.*

because the CGS method takes more iterations to converge to the solution of the linear system at each of these steps. This balance between the number of AF-CGS steps and the number of CGS iterations for each step, is considered in detail for unsteady flows in [12].

4.3 Turbulent flow over an RAE2822 aerofoil

$$M_\infty = 0.73, \quad \alpha = 2.79, \quad \mathrm{Re} = 6.5 \times 10^6, \quad 256 \times 64 \text{ mesh}.$$

After 5000 iterations, and 17,700 seconds of CPU time on an IBM RS/6000 320H, the explicit method fails to reduce the relative residual by one order. The explicit method proves impractical for this problem. However, if we use AF-CGS with an explicit starting procedure, the optimal time to convergence of 5100 seconds is obtained on the RS/6000. The explicit scheme was run for 400 iterations from freestream before switching to AF-CGS with a local CFL number of 30. A plot of the pressure coefficient on the aerofoil surface for the present results, and also for experiment, is shown in Figure 6. The present results are observed to show a good agreement with experiment. A comparison of the convergence rates for the explicit method, and for AF-CGS is shown in Figure 7. This shows the large im-

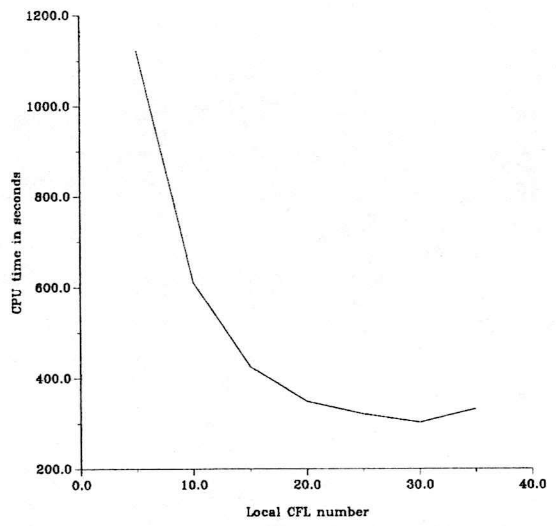

Figure 5: *Comparison of convergence rates for AF-CGS for varying values of the local CFL number, for turbulent flow over a NACA64A010 aerofoil.*

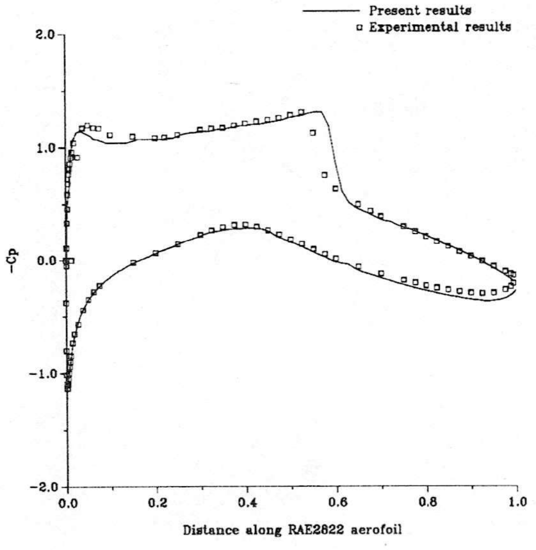

Figure 6: *Comparison of numerical results, and experimental results of [25] for turbulent flow over an RAE2822 aerofoil.*

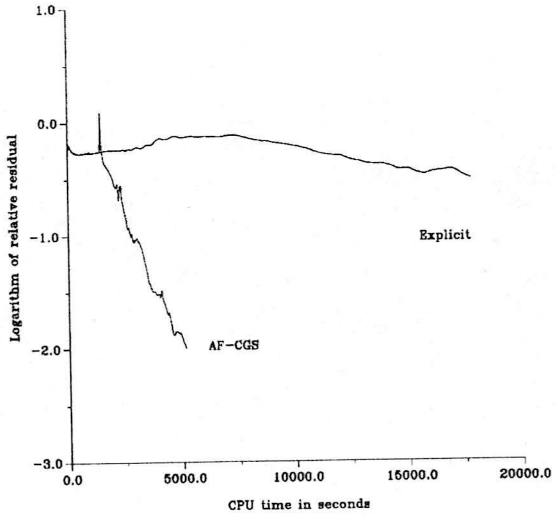

Figure 7: *Comparison of convergence rates for the explicit method alone, and AF-CGS with a local CFL number of* 30, *for turbulent flow over an RAE2822 aerofoil.*

provement that has been achieved in the convergence rate using AF-CGS.

5 Conclusions

The AF-CGS method has been shown to give fast convergence to steady state for the three test cases considered and the steady state solutions have been observed to be accurate. The AF-CGS method shows a great improvement over the explicit method with the same discretisation. A comparison of the AF-CGS method with other schemes will be considered in a future report.

The further development of the scheme could involve the investigation of acceleration techniques, and the extension to three dimensions. The algorithm speed could be increased by the freezing of the Jacobian matrices when the relative residual has been reduced by one order or at every few steps. Investigation of the appropriate grid density for the high order scheme is required. The use of a Block ILU Factorisation [3] as an alternative to the ADI preconditioner is worth investigating as it is likely to be beneficial when the scheme is extended to three dimensions.

References

[1] N.Qin and B.E.Richards. Sparse quasi-Newton method for high resolution schemes. *Notes in Numerical Fluid Dynamics*, 20:310–317, 1988.

[2] N.Qin X.Xu and B.E.Richards. α-GMRES:a new parallelizable iterative solver for large sparse non-symmetric linear systems arising from CFD. *Int. J. Num. Mths. Fluids*, 15:613–623, 1992.

[3] X. Xu. BILUF:a preconditioner of linear solver in Newton's method for solving steady state laminar locally conical Navier-Stokes equations. Technical report, G.U. Aero report 9321, 1993.

[4] V.Venkatakrishnan. Newton solution of inviscid and viscous problems. *A.I.A.A. Journal*, 27(7):885–891, 1989.

[5] V.Venkatakrishnan. Viscous computations using a direct solver. *Computers and Fluids*, 18(2):191–204, 1990.

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

[7] D.L.Whitfield and L.K.Taylor. Discretized Newton-relaxation solution of high resolution flux-differnce split schemes. Technical report, A.I.A.A. 91-1539-CP, 1991.

[8] P.D.Orkwis and D.S.McRae. Newton's method solver for high-speed viscous separated flowfields. *A.I.A.A. J.*, 30:78–85, 1991.

[9] M.Hafez C.P.Van Dam and J.Ahmad. Calculation of viscous flows with separation using Newton's method and direct solver. Technical report, A.I.A.A. 88-0412-CP, 1988.

[10] A.B.Stephens G.R.Shubins and H.M.Glaz. Steady shock tracking and Newton's method applied to one-dimensional duct flow. *J. Comp. Phys.*, 39:364–374, 1980.

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

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

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

[14] 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.

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

[16] X. Xu and B.E. Richards. Simplified procedure for numerically approximate Jacobian matrix generation in Newton's method for solving the Navier-Stokes equations. Technical report, G.U. Aero report 9320, 1993.

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

[18] X. Jin. Derivation of analytical Jacobian matrix for Newton's method to 3D N-S equations. Technical report, G.U. Aero report 9318, 1993.

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

[20] M. Vitaletti. Solver for unfactored schemes. *AIAA J.*, 29:1003–1005, 1988.

[21] A.Lerat and J.Sidès. Efficient solution of the steady Euler equations with a centered implicit method. In K.W.Morton and M.J.Baines, editors, *Numerical Methods for Fluid Dynamics 3*, pages 65–86. Clarendon Press, 1988.

[22] R.Radespiel, C.Rossow, and R.C.Swanson. Efficient cell-vertex multigrid scheme for the three-dimensional Navier-Stokes equations. *A.I.A.A. J.*, 28:1464–1472, 1990.

[23] J.A.McKenzie P.I.Crumpton and K.W.Morton. Cell vertex algorithms for the compressible Navier-Stokes equations. Numerical Analysis Report 12, Oxford University Computing Laboratory, 1991.

[24] J.J. Olsen. Compendium of unsteady aerodynamic measurements. Technical Report 702, AGARD, 1982.

[25] P.H.Cook, M.A.McDonald, and M.C.P.Firmin. Aerofoil RAE2822 - pressure distributions, and boundary layer and wake measurements. Technical Report AR-138, AGARD, 1979.