University of Glasgow
DEPARTMENT OF
AEROSPACE ENGINEERING

An investigation into the convergence of the AF-CGS method

I.C. Glover

Glasgow University Aero report 9327

An investigation into the convergence of the AF-CGS method

I.C. Glover

Glasgow University Aero report 9327

An investigation into the convergence of the AF-CGS method

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

November 30, 1993

Abstract

The relationship between the relative residual produced by the AF-CGS method and the lift, drag and pressure coefficients is investigated for steady turbulent flow over a NACA64A010 aerofoil, and over an RAE2822 aerofoil. This enables the most suitable convergence criterion to be established.

*supported by MoD and British Aerospace

Contents

1 Introduction

The AF-CGS method was implemented for inviscid flows in [1], and for unsteady turbulent flows in [2]. It was modified for use with steady turbulent flows in [3]. The convergence criterion which was used in [3] was the reduction of the relative residual by two orders from freestream. This was observed to give solutions which showed a good agreement with experiment. A more thorough investigation of the relationship between the relative residual and various flow properties, namely the lift, drag and pressure coefficients on the aerofoil, was required to determine how good the computed solution was when different convergence criteria were reached.

Such an investigation is carried out in this report for turbulent aerofoil flows. The report is set out as follows. An outline of the AF-CGS method is given. The convergence of the AF-CGS method is then investigated by considering the behaviour of the lift, drag and pressure coefficients as the relative residual changes. This investigation is carried out for turbulent flow over both a NACA64A010 aerofoil and an RAE2822 aerofoil and introduces an error indicator which depends upon the difference between computed and experimental pressure coefficient values. Finally, conclusions are drawn as to the best convergence criterion to choose.

2 The AF-CGS method

This method is described in detail in [4] and [3], but we shall present an outline here for completeness. The flows of interest are described by the thin-layer Navier-Stokes equations, which are outlined in much of the literature, for example [4]. 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 [5] and Roe [6] 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 Osher's scheme is used 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.

One implicit step may be written

$$(\frac{\partial c}{\partial p} + \Delta t \frac{\partial R_x^\mu}{\partial p} + \Delta t \frac{\partial R_y^\mu}{\partial p})\delta p = -\Delta t(R_x + R_y) \qquad (1)$$

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 (1). 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 [4] 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.

We adopt an approach which involves the solution of the unfactored linear system (1), which is solved 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 (1) 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. A summary of these approaches is given in [3]. The latest of these involves the use of an analytic evaluation of the flux derivatives.

The problem with this method is the complexity of the derivatives of the Osher approximations to the fluxes. These derivatives were still evaluated numerically in [7] 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 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.

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 [7] 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 [8] and is restated in [1].

Successful conjugate gradient methods need good preconditioning. Incomplete LU decomposition (ILU) has been successfully applied for steady fluid flow problems [7] [9]. 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 matrix on the left hand side of (1) can be factored into three block diagonal matrices

$$(\frac{\partial c}{\partial p} + \Delta t\frac{\partial R_x}{\partial p} + \Delta t\frac{\partial R_y}{\partial p}) \approx (\frac{\partial c}{\partial p} + \Delta t\frac{\partial R_x}{\partial p})(\frac{\partial c}{\partial p})^{-1}(\frac{\partial c}{\partial p} + \Delta t\frac{\partial R_y}{\partial p}). \quad (2)$$

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

$$A\mathbf{x} = \mathbf{b} \quad (3)$$

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

$$C^{-1}A\mathbf{x} = C^{-1}\mathbf{b} \quad (4)$$

more amenable to conjugate gradient methods. The ADI method gives a fast method of calculating an approximate solution to (3) or, restating this, of forming the matrix vector product

$$C^{-1}\mathbf{b} = \mathbf{x}. \quad (5)$$

Hence, if we use the inverse of the ADI factorisation, given by the right hand side of (2) 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 by 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 (3) has been solved to required tolerance

3 Turbulent flow over a NACA64A010 aerofoil

The AF-CGS method is implicit, with each iteration requiring a considerable amount of CPU time, so we do not want to run the code for any more iterations than is necessary to achieve a good solution. In [3] the relative residual was used as a measure of convergence. This is given by

$$\text{Relative residual} \ = \frac{||R_n||_2}{||R_0||_2}$$

where R_0 represents the residual of the freestream data, and R_n represents the residual after n iterations. We would like to investigate the relationship between the value of the relative residual, i.e. the convergence, and various flow properties, so that we can achieve a balance between the quality of the solution and the computational cost.

For turbulent flow over a NACA64A010 aerofoil the flow conditions are given by

$$M_\infty = 0.796, \ \alpha = 0, \ \text{Re} = 12.56 \times 10^6, \ 71 \times 33 \text{ mesh}.$$

In all cases the explicit method is run for 150 iterations from freestream, before switching to AF-CGS with a local CFL number of 30. The real variables are stored using either single precision (*real* version of the code), or double precision (*real*8* version of the code). A plot of the logarithm of the relative residual against the number of iterations is shown in Figure 1, for both the *real* and the *real*8* versions of the code. The *real* version of the code is observed to stop converging when the relative residual has been reduced by between 2.5 and 3 orders from freestream. The convergence rate of the *real*8* version is observed to slow considerably after this point is reached. Similar results were noted in [10] for inviscid flows, and were attributed to oscillations in the far-field cell residuals. The results shown in Figures 2 and 3 indicate that a similar situation is occurring here. However, the reason for this occurrence is not known at the present time. Figure 2 shows the frequency with which the largest residual occurs in a particular row of cells for each of 850 AF-CGS iterations which follow

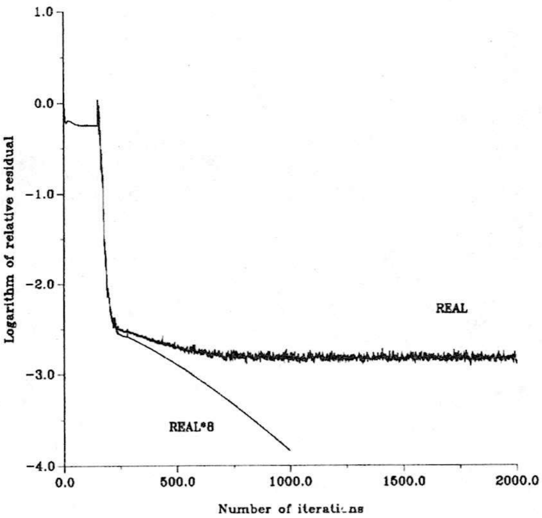

Figure 1: *Convergence histories for the real and real*8 versions of the code.*

the explicit starting procedure. The cells in the y-direction are indexed from $j = 1$ near the aerofoil to $j = 32$ near the far-field boundary. The largest residual is shown to occur in a cell whose j index is 31 or 32 for more than half of the iterations. In Figure 3 the ratio of the largest cell residual in the near-field to that over the whole mesh is plotted against the number of AF-CGS iterations. The near-field is defined by nodes in the x-direction whose index lies between 6 and 66, and nodes in the y-direction whose index lies between 1 and 23. For about the first 100 iterations the largest cell residual often occurs in the near-field, but for the next 250 iterations the largest cell residual occurs mainly in the far-field. Now, the 100 iterations point in Figure 3 corresponds to the 250 iterations point in Figure 1 because of the explicit starting procedure. Hence the point at which the convergence of the AF-CGS scheme either stops or slows down corresponds to the point at which the largest cell residual switches from occurring mainly in the near-field to occurring mainly in the far-field. In [10] this difficulty was overcome by adding some artificial dissipation, but such a technique is far from desirable for the present Navier-Stokes calculations.

Using the *real*8* version of the code, Figure 1 shows that we can drive the relative residual down considerably lower than with the *real* version. However the *real*8* version requires twice as much CPU time per iteration and twice as much memory as the *real* version, so we would prefer to use

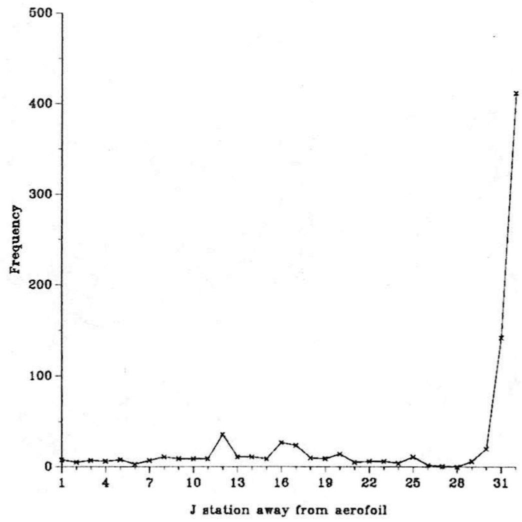

Figure 2: *The frequency with which the largest residual occurs in a cell which has a particular j index. A cell in which $j = 1$ is next to the aerofoil (or the wake cut), whilst a cell in which $j = 32$ is next to the far-field boundary.*

the *real* version and investigate more closely the relationship between the size of the relative residual and how good the computed solution is.

3.1 Convergence of flow properties

For the *real* version of the code, a plot of the relative lift coefficient against convergence is shown in Figure 4. The value of the lift coefficient is taken relative to that obtained when the *real*8* version of the code has been used to reduce the relative residual to 1.5×10^{-4} after 850 AF-CGS iterations. The convergence criterion used in [3] was to reduce the relative residual by two orders from freestream. From Figure 4, this is observed to correspond to a plateau, which gives a value for the lift coefficient which is 42 percent higher than the final value. However, when the relative residual is reduced by two and a half orders the lift coefficient is within 8 percent of the final value. Similar results are shown in Figure 5 for the drag coefficient relative to that obtained after 1000 total iterations of the *real*8* version of the code. When the relative residual has been reduced by two orders, there is a 58 percent difference in the drag coefficient from the final value. This difference is lowered to 14 percent by a further half order reduction in the relative residual.

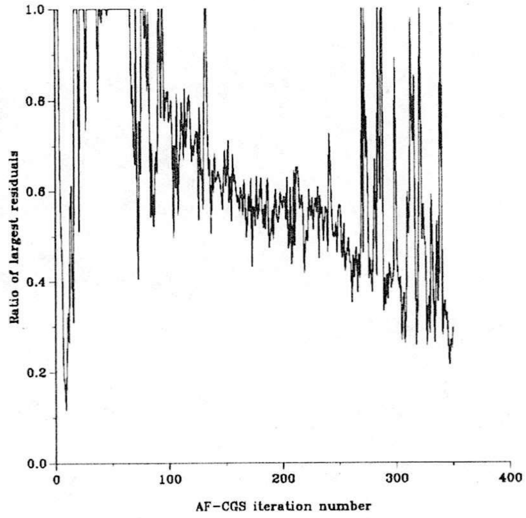

Figure 3: *Ratio of the largest cell residual in the near-field to that over the whole mesh.*

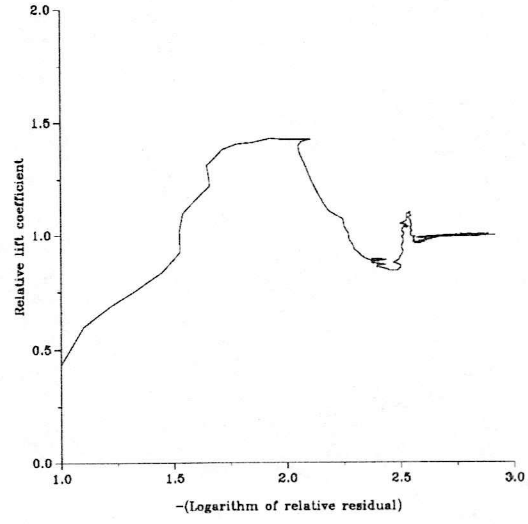

Figure 4: *The relative lift coefficient against minus the logarithm of the relative residual.*

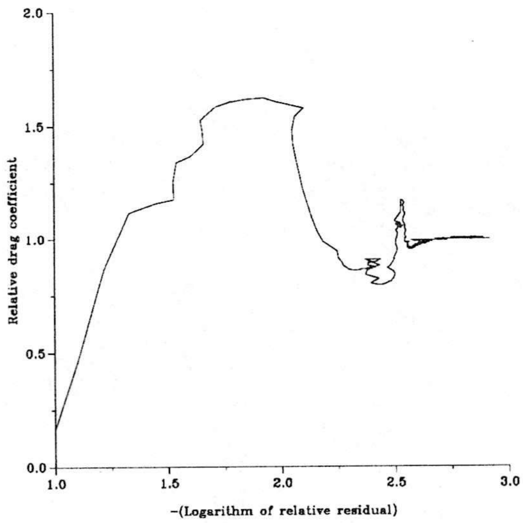

Figure 5: *The relative drag coefficient against minus the logarithm of the relative residual.*

Since experimental data exists for this problem, we can introduce a new convergence criterion. If we calculate the discrete L_2 norm of the difference between the experimental and computed values for the pressure coefficient on the aerofoil, and divide by the discrete L_2 norm of the experimental data, we can establish a relationship between this difference and the convergence, which is shown in Figure 6. Once more we observe a plateau around a two orders reduction from freestream corresponding to a difference of 34 percent from the final value. When the relative residual has been reduced by two and a half orders from freestream the difference from the final value is lowered to 11 percent.

Taking into account the relationships between the relative residual and the various flow properties that have been considered, and also the fact that the *real* version of the code cannot reduce the relative residual by three orders, we shall modify our convergence criterion to be the reduction of the relative residual by two and a half orders from freestream.

This requires 59 AF-CGS iterations on top of the 150 explicit iterations, and uses 390 seconds of CPU time on a SPARC10 Model 30. Figure 6 shows that a two and a half orders reduction gives a better approximation to the pressure distribution than a two orders reduction, which requires 300 seconds on the same machine. In Figure 7, the pressure distribution obtained after 209 total iterations of the *real* version of the code is com-

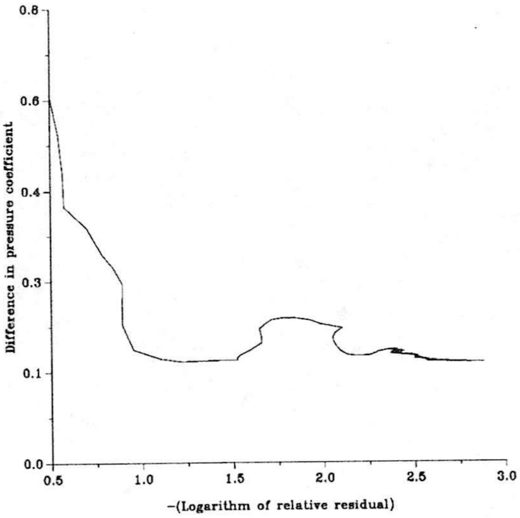

Figure 6: *The norm of the difference between the computational and experimental pressure coefficients against minus the logarithm of the relative residual.*

pared with that obtained after 1000 total iterations of the *real*8* version of the code, and also with experiment. Reducing the relative residual by two and a half orders using the *real* version of the code is observed to produce virtually the same pressure distribution as reducing the relative residual by almost four orders using the *real*8* version of the code.

A two orders reduction from freestream is observed to give a good pressure distribution and general flow solution, while a two and a half orders reduction also gives good integrated values.

4 Flow over an RAE2822 aerofoil

The flow conditions are given by

$$M_\infty = 0.73, \quad \alpha = 2.79, \quad \mathrm{Re} = 6.5 \times 10^6, \quad 257 \times 65 \text{ mesh.}$$

In all cases the explicit method is run for 400 iterations from freestream before switching to AF-CGS with a local CFL number of 30. A plot of the logarithm of the relative residual against the number of iterations for the *real* version of the code is shown in Figure 8. Once again we see the convergence stop when the relative residual has been reduced by about two and a half orders from freestream. The *real*8* version cannot be run on this mesh on the available machines because of the memory requirement.

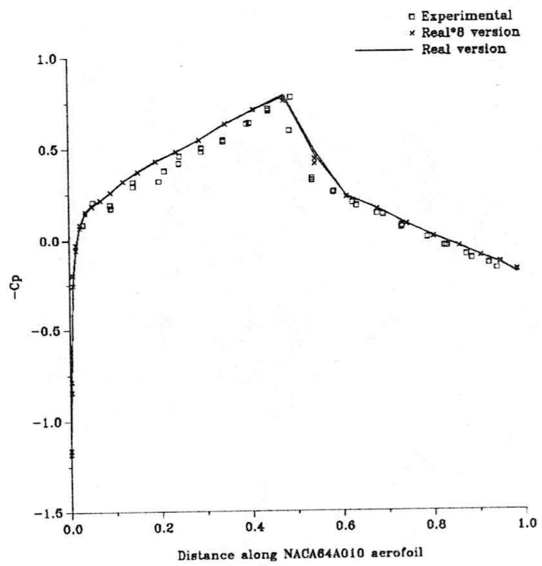

Figure 7: *A plot of various pressure coefficient profiles.*

Figure 9 shows the frequency with which the largest residual occurs in a cell with a particular j index for each of 1500 AF-CGS iterations which follow the explicit starting procedure. Again a high proportion of the largest residuals occur near the far-field boundary. A consideration of the near-field residuals leads to a similar profile to that shown in Figure 3.

4.1 Convergence of flow properties

Plots of the relative lift coefficient and the relative drag coefficient against minus the logarithm of the relative residual are shown in Figures 10 and 11 respectively. The difference from the final value after a two orders reduction is 3 percent for the lift coefficient and 9 percent for the drag coefficient. For a two and a half orders reduction these values are 2 percent and 3 percent respectively. These values are much lower than those obtained for the NACA64A010 aerofoil, which is to be expected due to the quality and size of the present mesh. Once again there is experimental data for this problem, and we can measure the difference between the computed and experimental values for the pressure coefficient on the aerofoil. This is shown in Figure 12, and shows that a two orders reduction corresponds to a difference of 10 percent from the final value, while a 2.5 orders reduction corresponds to a 7 percent difference. To take the residual down 2.5 orders requires 400 explicit iterations plus 180 AF-CGS iterations, and

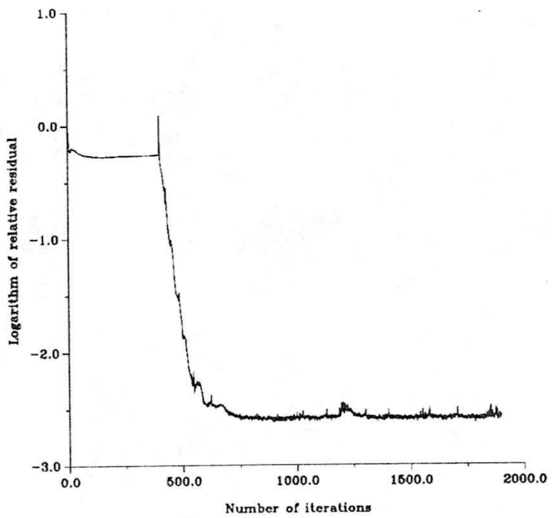

Figure 8: *Convergence history for flow over an RAE2822 aerofoil.*

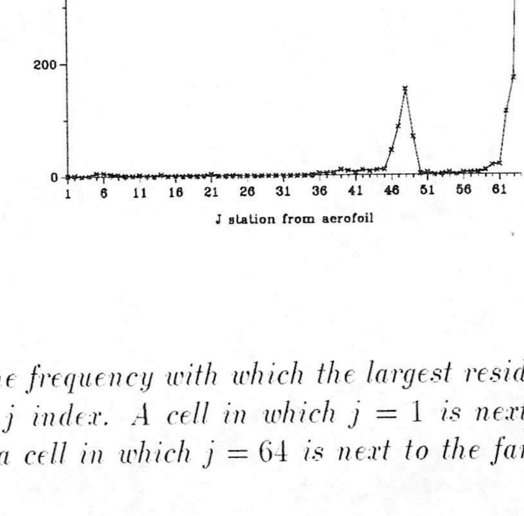

Figure 9: *The frequency with which the largest residual occurs in a cell which has a particular j index. A cell in which $j = 1$ is next to the aerofoil (or the wake cut), whilst a cell in which $j = 64$ is next to the far-field boundary.*

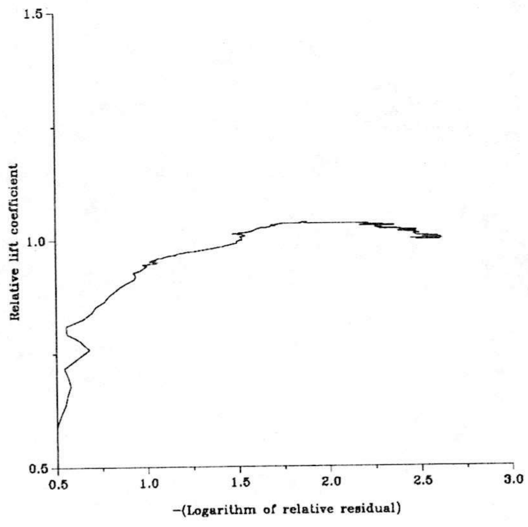

Figure 10: *Relative lift coefficient against minus the logarithm of the relative residual.*

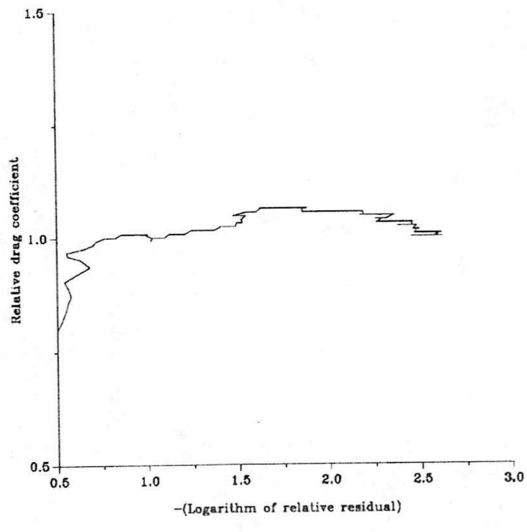

Figure 11: *Relative drag coefficient against minus the logarithm of the relative residual.*

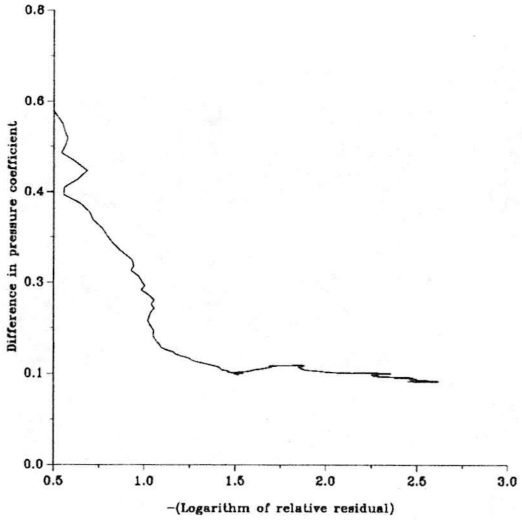

Figure 12: *Difference between computed and experimental pressure coefficient against minus the logarithm of the relative residual.*

takes 7000 seconds of CPU time on an IBM RS/6000 320H, while a 2 orders reduction requires 5100 seconds on the same machine. When the convergence has stopped the lift, drag and pressure coefficients remain at a particular value while the relative residual oscillates. A comparison of the pressure coefficient profiles after 180 and 1500 AF-CGS iterations with the experimental results is shown in Figure 13. The profile after 180 AF-CGS iterations is close to the profile after 1500 iterations.

5 Conclusions

In [3] a two orders reduction of the relative residual from freestream was shown to give a good solution with relatively small expense. By investigating the relationship between various flow properties and the relative residual it has been shown that an increase in the CPU time of around 30 percent leads to a better solution, with a two and a half orders reduction from freestream. For the NACA64A010 aerofoil the improvement in the computed solution is considerable, but for the RAE2822 aerofoil there is a much smaller improvement in the solution.

Taking into account both the increased cost and the improvement in the solution, the best option is to use the *real* version of the code to reduce the relative residual by 2.5 orders from freestream.

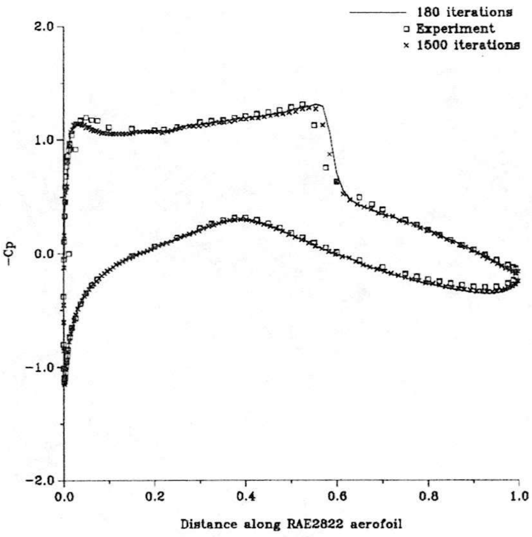

Figure 13: *Various pressure coefficient profiles for the RAE2822 case.*

The logarithm of the relative residual cannot be reduced to −3 with this version, which appears to be caused by relatively large cell residuals in the far-field. An investigation into the cause of these larger residuals is required.

References

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

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

[3] 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.

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

[5] 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.

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

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

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

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

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