INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 25, 1347–1371 (1997)

GMRES PHYSICS-BASED PRECONDITIONER FOR ALL REYNOLDS AND MACH NUMBERS: NUMERICAL EXAMPLES

N. NIGRO*, M. STORTI AND AND S. IDELSOHN

Grupo de Tecnología Mecánica del INTEC, CONICET, Universidad Nacional del Litoral, Güemes 3450, 3000 Santa Fe, Argentina

SUMMARY

This paper presents several numerical results using a vectorized version of a 3D finite element compressible and nearly incompressible Euler and Navier–Stokes code. The assumptions were set on laminar flows and Newtonian fluids.

The goal of this research is to show the capabilities of the present code to treat a wide range of problems appearing in laminar fluid dynamics towards the unification from incompressible to compressible and from inviscid to viscous flow codes.

Several authors with different approaches have tried to attain this target in CFD with relative success. At the beginning the methods based on operator splitting and perturbation were preferred, but lately, with the wide usage of time-marching algorithms, the preconditioning mass matrix (PMM) has become very popular. With this kind of relaxation scheme it is possible to accelerate the rate of convergence to steady state solutions with the modification of the mass matrix under certain restrictions. The selection of the mass matrix is not an easy task, but we have certain freedom to define it in order to improve the condition number of the system. In this paper we have used a physics-based preconditioner for the GMRES implicit solver developed previously by us and an SUPG formulation for the semidiscretization of the spatial operator.

In sections 2 and 3 we present some theoretical aspects related to the physical problem and the mathematical model, showing the inviscid and viscous flow equations to be solved and the variational formulation involved in the finite element analysis. Section 4 deals with the numerical solution of non-linear systems of equations, with some emphasis on the preconditioned matrix-free GMRES solver. Section 5 shows how boundary conditions were treated for both Euler and Navier–Stokes problems. Section 6 contains some aspects about vectorization on the Cray C90. The performance reached by this implementation is close to 1 Gflop using multitasking. Section 7 presents several numerical examples for both models covering a wide range of interesting problems, such as inviscid low subsonic, transonic and supersonic regimes and viscous problems with interaction between boundary layers and shock waves in either attached or separated flows. © 1997 John Wiley & Sons, Ltd.

Int. J. Numer. Meth. Fluids, **25**: 1347–1371 (1997)

No. of Figures: 31. No. of Tables: 0. No. of Refs: 35.

KEY WORDS: steady state compressible and incompressible flow; Euler and Navier–Stokes equations; 3D finite element method by SUPG; explicit and implicit solver; preconditioning mass matrix; vector code.

* Correspondence to: N. Nigro, Grupo de Tecnología Mecánica del INTEC, CONICET, Universidad Nacional del Litoral, Güemes 3450, 3000 Santa Fe, Argentina. E-mail: nnigro@venus.unl.edu.ar

Contract grant sponsor: CONICET (Argentina)

CCC 0271–2091/97/121347–29 $17.50

Received June 1996
Revised February 1997

1. INTRODUCTION

This paper presents a survey of results using a unified finite element formulation to solve from nearly incompressible to supersonic regimes and from inviscid to viscous flows.[1] During the last few years a lot of effort has been put into the application of the finite element method to treat fluid dynamics problems, trying to bring the inherent advantages of the method into an area where other numerical techniques were previously and successfully applied, such as finite difference and finite volume among others.

Combining a strong mathematical basis with the capability to treat complex geometries, the finite element method is one of the commonly used formulations to solve 3D problems applied to industrial and scientific environments.

The progress achieved in supercomputing makes possible the solution of increasingly more challenging problems with these numerical tools by parallelizing and vectorizing the software developed in this area.

In the CFD context, SUPG,[2–14] or its generalization called Galerkin least square,[15–17] has been one of the most popular methods used during recent years. These methods are based on the Petrov–Galerkin formulation using weighting functions that are different from the approximation ones. While the former is based only on the usage of multilinear polynomial approximation, the latter is generalized to higher-order functions. In this way both methods enforce the stabilization needed to preclude the inherent oscillations that appear when the Galerkin method is applied. In this work we adopt an SUPG method using equal-order polynomials and conservative variables. A shock-capturing operator[2,6,8,10,11] is added to get a better resolution of the shock waves present in the applications. The non-linear system produced by the spatial semidiscretization is solved by a time-marching algorithm using a matrix-free preconditioned GMRES iterative solver[18,19] for each linear system obtained at each time step. Indeed, this preconditioner is defined by two steps: firstly we modify the mass matrix in order to improve the condition number of the system for all Reynolds and Mach numbers[1,20–23] and secondly we use the nodal block diagonal preconditioner inside the GMRES routine.[1,2,11] The goal of this paper is to show the performance of this vector code in solving a very wide range of fluid dynamics problems, covering from inviscid to strongly viscous flows and nearly incompressible to transonic and supersonic flows using the same formulation. Our results were validated with experimental, analytical and other numerical results solved by several authors.

2. PROBLEM STATEMENT

The physics involved in our fluid dynamics problems is mathematically governed by the Euler and Navier–Stokes equations. In the present work we use a version written for a non-inertial frame of reference with the possibility to solve problems involving rotating systems.[24]

The 3D compressible Navier–Stokes equations are classically written in terms of conservative variables in the form

$$\mathbf{F}^i_{\mathrm{a},i} = \mathbf{F}^i_{\mathrm{d},i} + \mathscr{F}. \tag{1}$$

Implicitly we assume $(\,)_{,i} = \partial(\,)/\partial x_i$.

$\mathbf{U} \in \mathbb{R}^5$ is the fluid local state vector, with $\mathbf{U} = [\rho, \rho\mathbf{u}^{\mathrm{T}}, \rho e]^{\mathrm{T}}$, where ρ, $\mathbf{u}$ and e are the density, velocity and total energy of the fluid respectively. $\mathbf{F}_{\mathrm{a}}, \mathbf{F}_{\mathrm{d}} \in \mathbb{R}^{5\times 3}$ are the advective and diffusive

fluxes respectively, which depend on the state vector and its gradient as

$$\mathbf{F}_{\mathrm{a}}(\mathbf{U}) = \begin{bmatrix} \rho\mathbf{u}^{\mathrm{T}} \\ \rho\mathbf{u}\mathbf{u}^{\mathrm{T}} + p\mathbf{I}_{3\times 3} \\ (\rho e + p)\mathbf{u}^{\mathrm{T}} \end{bmatrix}, \qquad \mathbf{F}_{\mathrm{d}}(\mathbf{U}, \nabla\mathbf{U}) = \begin{bmatrix} 0 \\ \boldsymbol{\sigma} \\ (\boldsymbol{\sigma}\cdot\mathbf{u} + \mathbf{q})^{\mathrm{T}} \end{bmatrix}. \tag{2}$$

Here p, $\boldsymbol{\sigma}$ and $\mathbf{q}$ are the thermodynamic pressure, the deviatoric stress tensor and the heat flux vector respectively, with

$$\sigma_{jk} = \mu(u_{j,k} + u_{k,j}) + \lambda u_{l,l}\delta_{jk}, \tag{3}$$

$$q_j = \kappa\theta_{,j}, \tag{4}$$

where μ is the dynamic viscosity, λ is the second viscosity coefficient, κ is the thermal conductivity and θ is the temperature.

We finish the description of the mathematical model by introducing the state equation of the fluid and the relation between the energy and two of the thermodynamic variables of the fluid:

$$\theta = \theta(p, \rho), \qquad e = e(p, \rho). \tag{5}$$

Specifically we use the ideal gas law

$$p/\rho = R\theta = C_{\mathrm{v}}(\gamma - 1)\theta = (\gamma - 1)i, \tag{6}$$

$$p = (\gamma - 1)\rho i, \tag{7}$$

where R and C_{v} represent the gas universal constant and the specific heat at constant volume respectively and i is the internal energy

$$i = e - \tfrac{1}{2}\|\mathbf{u}\|^2. \tag{8}$$

The body force $\mathscr{F}$ is given by

$$\mathscr{F} = \rho \begin{pmatrix} 0 \\ \mathbf{f}_{\mathrm{Cor}} + \mathbf{f}_{\mathrm{Cent}} \\ \mathbf{f}_{\mathrm{Cent}}\cdot\mathbf{u} \end{pmatrix}, \tag{9}$$

where $\mathbf{f}_{\mathrm{Cor}} = -2\boldsymbol{\Omega}\times\mathbf{u}$ and $\mathbf{f}_{\mathrm{Cent}} = -\boldsymbol{\Omega}\times\boldsymbol{\Omega}\times\mathbf{x}$ are the Coriolis and centrifugal forces respectively.

This kind of advective–diffusive system is incompletely parabolic because the continuity equation does not have diffusive term.

In the Euler equation case we can use the above system but dropping the diffusive flux term $\mathbf{F}_{\mathrm{d}}^i$, resulting in a purely advective hypberbolic system.

It is useful to rewrite equation (1) in the quasi-linear form

$$\mathbf{A}_j\mathbf{U}_{,j} = (\mathbf{K}_{ij}\mathbf{U}_{,j})_{,i} + \mathscr{F}, \tag{10}$$

where $\mathbf{A}_j = \partial\mathbf{F}_{\mathrm{a}}^j/\partial\mathbf{U}$ is the jth advective Jacobian matrix and $\mathbf{K}_{ij}$ is one of the components of the tensor $\mathbf{K}$ representing the diffusive Jacobian matrix, with $\mathbf{F}_{\mathrm{d}}^i = \mathbf{K}_{ij}\mathbf{U}_{,j}$. Expressions for the above Jacobian matrices may be found in several references.[2,10,11,14]

3. SPATIAL DISCRETIZATION BY FINITE ELEMENT METHOD

In order to get a numerical solution of the continuum problem presented in Section 2, we have to discretize the problem using a particular numerical method. In this work we use an SUPG technique that is very popular in the context of the finite element method and is one of the most referenced in

the CFD area.[2–14] It is based on the Petrov–Galerkin weighted residual method which allows one to use test functions that can be different from the interpolation ones and not necessarily continuous. This method introduces the numerical dissipation needed to stabilize the system in advection-dominated problems, keeping the consistency with the continuum problem. For each node a there is an interpolation function $\mathbf{N}_a$ (hat type in 1D, bilinear in 2D and multilinear in general) and a test function $\mathbf{W}_a = \mathbf{N}_a + \mathbf{P}_a$, where $\mathbf{P}_a$ is called the perturbation function. The standard Galerkin method is recovered when we impose $\mathbf{P}_a \equiv 0$. The $\mathbf{W}_a$ (and, of course, $\mathbf{P}_a$) are not necessarily continuous through the inter-element boundaries. Next we describe the variational formulation employed and in Section 3.2 we explain how to get the perturbation function.

3.1. Variational formulation

Given a finite element partition of the original domain Ω into elements Ω_e, $e = 1, \ldots, N_{\text{el}}$, with N_{el} the number of elements in the mesh, the problem is to find $\mathbf{U}^h \in \mathscr{S}$ such that $\forall \mathbf{N} \in \mathscr{V}$

$$\int_\Omega (\mathbf{N}_a^{\mathrm{T}} \mathbf{A}_i \mathbf{U}^h_{,i} + \mathbf{N}^{\mathrm{T}}_{a,i} \mathbf{K}_{ij} \mathbf{U}^h_{ij}) \, \mathrm{d}\Omega + \sum_{e=1}^{N_{\text{el}}} \int_{\Omega_e} \mathbf{P}_a^{e\mathrm{T}} [\mathbf{A}_i \mathbf{U}^h_{,i} - (\mathbf{K}_{ij} \mathbf{U}^h_{,j})_{,i} - \mathscr{F}] \, \mathrm{d}\Omega = \int_\Omega \mathbf{N}_a^{\mathrm{T}} \mathscr{F} \, \mathrm{d}\Gamma + \int_{\Omega_h} \mathbf{N}_a^{\mathrm{T}} \mathbf{h} \, \mathrm{d}\Gamma \tag{11}$$

is satisfied, where $\mathbf{h}$ is the diffusive flux imposed on the boundary Γ_h and $\mathbf{U}^h$ is the finite element approximation of $\mathbf{U}$. $\mathscr{S}$ and $\mathscr{V}$ are the trial and weighting function spaces respectively, commonly used in this context.[2,11]

The Euler–Lagrange form is obtained through classical integration by parts:

$$\sum_{e=1}^{N_{\text{el}}} \int_{\Omega_e} \mathbf{W}_a^{\mathrm{T}} (\mathbf{A}_i \mathbf{U}^h_{,i} - \mathbf{K}_{ij} \mathbf{U}^h_{,ij} - \mathscr{F}) \, \mathrm{d}\Omega + \int_{\Gamma_{\text{int}}} \mathbf{N}_a^{\mathrm{T}} [n_i \mathbf{K}_{ij} \mathbf{U}^h_{,j}] \, \mathrm{d}\Gamma + \int_{\Gamma_h} \mathbf{N}_a^{\mathrm{T}} (n_i \mathbf{K}_{ij} \mathbf{U}^h_{,j} - \mathbf{h}) \, \mathrm{d}\Gamma = 0,$$

where

$$[n_i \mathbf{K}_{ij} \mathbf{U}^h_{,j}](\mathbf{x}) = n_i(\mathbf{x}) [(\mathbf{K}_{ij} \mathbf{U}^h_{,j})(\mathbf{x}^+) - (\mathbf{K}_{ij} \mathbf{U}^h_{,j})(\mathbf{x}^-)] \tag{12}$$

is the jump in the diffusive flux throughout the inter-element boundary, $\mathbf{x}$ is a point which lies there, $\mathbf{x}^\pm$ are points belonging to each side of the boundary and Γ_{int} is the inter-element contour. Consistency is guaranteed because the continuum solution is also a solution of this variational formulation.

3.2. Perturbation function

As mentioned earlier, the goal of the SUPG scheme is the modification of the standard basis functions in order to stabilize the discretized problem. Since the publication of the original work in 1982 by Brooks and Hughes,[5] a lot of work has developed around this subject, but a complete list of contributions is outside the scope of this work. It is important to mention that several different perturbation function approaches have arisen since the original paper. Here we adopt that most commonly used for the Euler and Navier–Stokes equations,[7,10,13,14] i.e.

$$\mathbf{P}_a^e = \boldsymbol{\tau}^{\mathrm{T}} \mathbf{A}_i^{\mathrm{T}} \mathbf{N}_{,i}. \tag{13}$$

In this expression there appears a matrix $\boldsymbol{\tau}$ that varies with different authors and represents the critical point of the design of the stabilization method. Sometimes called the *intrinsic time scale matrix*, it can be deduced through its application to simple situations where an exact analytical

solution can be found. Its generalization is very difficult and relies on heuristic arguments. In the next subsection we present the definition used for the $\boldsymbol{\tau}$-matrix.

3.3. *Definition of $\boldsymbol{\tau}$ matrix*

Suppose that we are interested in the solution of the Euler equations and we start by taking the one-dimensional case. In this situation and by extrapolation of the advection scalar equation we choose $\boldsymbol{\tau}$ as the inverse of the norm of the advective Jacobian matrix, i.e.

$$\boldsymbol{\tau} = \|\mathbf{B}\|^{-1}, \tag{14}$$

where $\mathbf{B}$ represents the cited Jacobian but transformed to the master element. As is well known, in the one-dimensional case this Jacobian is diagonalizable and we can get the right $\boldsymbol{\tau}$ to produce the exact solution. When trying to extend to general three-dimensional situations, this advantage disappears and the generalization to produce good results is not so obvious. Hughes and Mallet[7] proposed solving an eigenvalue problem for the matrix

$$\|\mathbf{B}\| = (\mathbf{B}_1^2 + \mathbf{B}_2^2 + \mathbf{B}_3^2)^{1/2}. \tag{15}$$

The above is equivalent to applying the L_2-norm to the vector of matrices $\mathbf{B}_i$, $i = 1, 2, 3$. In order to avoid the solution of the eigenvalue problem and consequently to save CPU time, we adopt the L_1-norm instead of L_2. As each Jacobian $\mathbf{B}_i$ is diagonalizable in its own basis, we can analytically compute each diagonalization and we only have to invert the final matrix:

$$\boldsymbol{\tau} = \|\mathbf{B}\|^{-1} = (|\mathbf{B}_1| + |\mathbf{B}_2| + |\mathbf{B}_3|)^{-1}, \tag{16}$$

with $|\mathbf{B}_i| = \mathbf{S}_i^{-1}|\boldsymbol{\Lambda}_i|\mathbf{S}_i$, where $\mathbf{S}$ and $\boldsymbol{\Lambda}$ are the corresponding eigenvector and eigenvalue matrices respectively.[21]

The extension to the Navier–Stokes equations is possible by several different methods and here we mention only some of them. One alternative could be to affect the intrinsic time scale matrix by only some scalar terms for the whole system of equations. This method was proposed by Shakib and Fortin[12] and is based on the definition of a *magic function* that depends on the Peclet number defined by the Reynolds number and assuming that the Prandtl number is $O(1)$. Another technique is based on the projection of the diffusivity tensor over the basis where the advective Jacobians are diagonalizable.[10] It is also possible to define the above matrix by changing the metric of the problem and diagonalizing the system using the basis of the regularized Peclet matrix $\mathbf{K}^{*-1}\mathbf{B}$ where $\mathbf{K}^{*-1}$ is the inverse of some regularization of the diffusivity tensor.[25,26] While the first is supported by heuristic arguments and the last two have a stronger theoretical basis, all three methods work reasonably well in practical applications and in our experience we have not detected substantial differences.

4. SOLVING THE NON-LINEAR ALGEBRAIC SYSTEM OF EQUATIONS

Using

$$\mathbf{U}^h = \sum_a N_a \mathbf{U}_a^h \tag{17}$$

as the approximation for the unknown variables and replacing the above definition of the intrinsic time scale matrix (equation (16)) in the expression for the perturbation function (equation (13)), applying it to equation (11) and adding the contribution of each element to the corresponding nodes

in the finite element grid, we obtain a non-linear algebraic system of equations, a discretized version of our original non-linear PDE system:

$$\mathbf{G}(\mathbf{U}) = \mathbf{0}, \quad \mathbf{G}: \mathbb{R}^n \to \mathbb{R}^n. \tag{18}$$

To solve this steady state problem, we have used a time-marching scheme to get the solution of the former as the stationary state of the latter, the temporal variable being a relaxation parameter. This kind of method is very popular nowadays. In this case the system of non-linear algebraic equations becomes a system of non-linear ordinary differential equations of the form

$$\mathscr{L}\mathbf{U} = \mathbf{U}_{,t} + \mathbf{G}(\mathbf{U}) = \mathbf{0}. \tag{19}$$

Then, given $\mathbf{U}^{(n)}$, the solution at $t = t_n$, we want to add a contribution $\Delta\mathbf{U}$ so that $\mathscr{L}\mathbf{U} = \mathbf{0}$. Expanding the non-linear term using a Taylor series gives

$$\mathbf{G}(\mathbf{U} + \Delta\mathbf{U}) = \mathbf{G}(\mathbf{U}) + \mathbf{G}_{,\mathbf{U}}\Delta\mathbf{U} + \text{higher-order terms}. \tag{20}$$

Replacing this in (19), we get

$$(\mathbf{U} + \Delta\mathbf{U})_{,t} + \mathbf{G}(\mathbf{U} + \Delta\mathbf{U}) = \mathbf{0}, \qquad \Delta\mathbf{U}_{,t} + \mathbf{G}_{,\mathbf{U}}\Delta\mathbf{U} = -\mathbf{U}_{,t} - \mathbf{G}(\mathbf{U}). \tag{21}$$

As we are interested in steady state solutions, the transient term on the right-hand side is neglected and in order to achieve the numerical solution we include a time discretization. Finally we arrive at the system

$$\left(\frac{1}{\Delta t}\mathbf{M} + \mathbf{C}(\mathbf{U})\right)\Delta\mathbf{U} = \mathbf{R}(\mathbf{U}), \tag{22}$$

where $\mathbf{M}$, $\mathbf{C} = \mathbf{G}_{,\mathbf{U}}$ and $\mathbf{R} = -\mathbf{G}(\mathbf{U})$ represent the mass matrix associated with the discretization of temporal terms, the residual Jacobian matrix and the residual vector respectively.

We arrive at the system to be solved, namely

$$\tilde{\mathbf{M}}\Delta\mathbf{U} = \mathbf{R}(\mathbf{U}), \tag{23}$$

defining an effective mass matrix $\tilde{\mathbf{M}} = (1/\Delta t)\mathbf{M} + \mathbf{C}(\mathbf{U})$.

4.1. Time local stepping

The computation of steady solutions using a time-marching algorithm allows us to select the time step according to temporal stability and convergence rate criteria. In the case of an explicit solver we known that there exists a critical time step that is a function of several numerical and physical parameters. Regardless of the right expression for this critical time step,[11] we know that one of the principal factors influencing its determination is the element size. If we adopt a global time step (the same for all the elements), we penalize those elements of large size. Since accuracy in time is not our concern, we can adopt a very popular strategy called *time local stepping*[27] in which each element of the mesh has its own time step in order to improve the rate of convergence.

As usual, the criterion is to equalize the Courant number for all the elements. This strategy is especially needed when one has a strong refinement in the mesh due to the physical properties of the flow involved in the computation, such as shocks, boundary layers, etc. Even though this strategy is specially adapted for explicit computations, it implementation within an implicit solver produces some interesting improvement in the convergence rate.[2]

4.2. *Linear system solution*

As presented in (23), we need to solve a linear system

$$\tilde{\mathbf{M}}\Delta\mathbf{U} = \mathbf{R}. \tag{24}$$

In the explicit case, $\tilde{\mathbf{M}}$ is usually lumped in order to get a diagonal matrix. Then the solution of the linear system decouples and it is not necessary either to invert or to store any matrix. These advantages and its own simplicity make the explicit solver one of the most popular ways to solve linear systems. Its disadvantage is associated with the strong limitations imposed by stability criteria. In the general implicit case we need to store and invert the left-side matrix and for this task we can choose among direct or iterative solvers. Direct solvers are frequently used in small- and medium-size computations such as those that appear in 2D problems, but their generalization to 3D problems is restricted owing to memory and CPU time resources. In this case we are constrained to use iterative methods. In this work we have used matrix-free GMRES, one of the most referenced iterative solvers in CFD applications during recent years.[18,19]

To accelerate the convergence of the GMRES algorithm, we have used two different kinds of preconditioners. The first one is introduced into the temporal term like a mass matrix affecting the time evolution of the solution, keeping the same steady solution. This matrix, called $\boldsymbol{\Gamma}$,[22] improves the convergence rate for all Reynolds and Mach numbers, modifying the characteristic speeds without altering the physical sense of the problem:

$$\boldsymbol{\Gamma} = \begin{pmatrix} \psi\|\mathbf{u}\|^2/2 & -\psi u & -\psi v & -\psi w & \psi \\ \frac{1}{2}(\psi\|\mathbf{u}\|^2 - 2)u & 1 - \psi u^2 & -\psi uv & -\psi uw & \psi u \\ \frac{1}{2}(\psi\|\mathbf{u}\|^2 - 2)v & -\psi uv & 1 - \psi v^2 & -\psi vw & \psi v \\ \frac{1}{2}(\psi\|\mathbf{u}\|^2 - 2)w & -\psi uw & -\psi vw & 1 - \psi w^2 & \psi w \\ \Theta_1 & \Theta_2 u & \Theta_2 v & \Theta_2 w & 1 - \Theta_2 \end{pmatrix},$$

where

$$\psi = \frac{\gamma - 1}{\beta M_r^2}, \qquad \Theta_1 = -\tfrac{1}{2}\|\mathbf{u}^2\|(1 + \Theta_2) - \frac{c^2}{\gamma - 1}, \Theta_2 = (1 - \gamma)(1 + \Xi), \qquad \Xi = \frac{\rho e + p}{\rho\beta M_r^2} - \delta,$$

$$\begin{aligned} &\beta M_r^2 = \epsilon c^2, \quad \text{with } \epsilon = \max(\epsilon_{\text{inv}}, \epsilon_{\text{vis}}), \\ &\epsilon_{\text{inv}} = M_r^2, \qquad \epsilon_{\text{vis}} = \max_j\left(\frac{\alpha_j(\alpha_j - 1)}{(\alpha_j - 1 + c^2/u_j^2)}\right), \\ &\alpha_j = \frac{CFL}{\sigma Re_{\Delta x_j}}, \quad j = x, y, z, \end{aligned} \tag{25}$$

c is the speed of sound, $[u_x; u_y; u_z] = [u; v, w]$ and M_r is a reference Mach number used to avoid singularities when the velocity is locally zero, with $M_{\min} = 10^{-6}$ in this work. This value is defined as

$$M_r = \begin{cases} M_{\min}, & M < M_{\min}, \\ M, & M_{\min} < M < 1, \\ 1, & M > 1. \end{cases} \tag{26}$$

σ represents the algorithmic Fourier number and $Re_{\Delta x_j}$ is the element Reynolds number with Δx_j as the characteristic length. δ is an arbitrary constant that plays the role of a coefficient of the time

derivative of pressure. When $\delta = 0$, the time derivative is dropped. When $\delta = 1$, we recover the standard energy equation.

More details about this preconditioner and its influence on the definition of the stabilized scheme and boundary conditions are presented in References 1 and 22. The second preconditioner plays the role of a scaling for our GMRES method and in this work we have used a right preconditioner based on a nodal block diagonal matrix.[1,2,11] To circumvent memory resource restrictions in large 3D computations, we have used a matrix-free version of this algorithm.[2]

5. BOUNDARY CONDITIONS

We have implemented the following boundary conditions: (a) Dirichlet boundary conditions; (b) slip and symmetric boundary conditions; (c) natural boundary conditions; (d) periodic boundary conditions; (e) absorbent boundary conditions.

As described earlier, we have used *conservative* variables within our code. One of the most common sets of variables to define *Dirichlet* boundary conditions is the *primitive* variables. The relation between these two variables is non-linear and the procedure used to update the conservative variables restricted to satisfying the boundary conditions imposed by *primitive* variables is the following:

$$\begin{aligned}
&\text{if } \rho \text{ is fixed, then } U_2 = g_1;\\
&\text{if } u_x \text{ is fixed, then } U_2 = U_1 g_2;\\
&\text{if } u_y \text{ is fixed, then } U_3 = U_1 g_3;\\
&\text{if } u_z \text{ is fixed, then } U_4 = U_1 g_4;\\
&\text{if } p \text{ is fixed, then } U_5 = g_5/(\gamma - 1) + \tfrac{1}{2}(U_2^2 + U_3^2 + U_4^2)/U_1;\\
&\text{if } \theta \text{ is fixed, then } U_5 = U_1 g_6 C_v + \tfrac{1}{2}(U_2^2 + U_3^2 + U_4^2)/U_1;
\end{aligned} \tag{27}$$

Here $(g_1; g_2; g_3; g_4; g_5; g_6)^T$ is a vector representing those values of the primitive variables that are fixed. Owing to the fact that the pressure or temperature imposes the value of U_5, it is not possible to fix both primitive variables simultaneously.

The treatment of essential boundary conditions inside the inner loop of the implicit solver is based on the above identities:

$$\begin{aligned}
&\text{if } \rho \text{ is fixed, then } \delta U_1 = 0;\\
&\text{if } u_x \text{ is fixed, then } \delta U_2 = g_2 \delta U_1;\\
&\text{if } u_y \text{ is fixed, then } \delta U_3 = g_3 \delta U_1;\\
&\text{if } u_z \text{ is fixed, then } \delta U_4 = g_4 \delta U_1;\\
&\text{if } p \text{ is fixed, then } \delta U_5 = \frac{1}{2}\frac{-U_2^2 + U_3^3 + U_4^2}{U_1^2}\delta U_1 + \frac{U_2}{U_1}\delta U_2 + \frac{U_3}{U_1}\delta U_3 + \frac{U_4}{U_1}\delta U_4;\\
&\text{if } \theta \text{ is fixed, then } \delta U_5 = \left(\frac{1}{2}\frac{-U_2^2 + U_3^2 + U_4^2}{U_1^2} + C_v g_6\right)\delta U_1 + \frac{U_2}{U_1}\delta U_2 + \frac{U_3}{U_1}\delta U_3 + \frac{U_4}{U_1}\delta U_4.
\end{aligned} \tag{28}$$

Here δU_i denotes the increment of the unknown.

In this way we satisfy the essential boundary conditions in an implicit way without degrading the convergence rate.

Slip boundary conditions are treated implicitly or explicitly according to the solver selection. While the explicit implementation consists of dropping the normal contribution of the momentum

equations in the update stage of the computation, the implicit version uses a rotation to the local frame, restricting the normal component to be fixed as a Dirichlet boundary condition.[2,9]

In the case of the *symmetric plane* boundary condition we have imposed

$$u_n = 0, \qquad q_n = 0, \qquad \tau_{rn} = \tau_{sn} = 0. \tag{29}$$

Periodic boundary conditions are very useful when the problem contains a repetitive structure of the variables involved in the computation. Examples of this kind of problem can be found in e.g. turbomachinery simulation, where the rotor can be divided into N_{blades} repetitive portions, or the flow around a cascade of bodies of similar shape.

Absorbent boundary conditions are very important at far-field boundaries. As is well known, the location of these boundaries is very important to decrease the computational cost while keeping a good rate of convergence and accurately. In this sense, absorbent boundary conditions allow one to diminish the reflection of error waves on the far-field boundary, thus improving the convergence rate. Much work has been done on designing absorbent boundary conditions for inviscid equations.[28,29] Our code was implemented by imposing those characteristics that are entering the domain and extrapolating those that are leaving the domain. The treatment of the Navier–Stokes absorbent boundary condition is a much more difficult problem.[30] In this work we have used the same inviscid absorbent boundary condition for viscous flows.

6. VECTOR CODE IMPLEMENTATION ON CRAY C90

The optimization techniques employed were the more basic ones, trying to maximize the number of vectorized loops. To do this, we put the longest loop as the innermost and, as is common in finite element computation, the loops are represented by loops over elements or loops over nodes. In other cases we have used other standard techniques such as splitting loops, unwinding the inner loop or even swapping loops. We have tried to minimize memory conflicts by an adequate use of array lay-out or avoiding indirect addressing whenever possible. The global performance for a single-processor configuration of the three-dimensional code, using multitasking with eight processors available on the Cray C90, the elapsed time was reduced by a factor of approximately three. This is equivalent to a global performance greater than 900 Mflops, which places this code almost within the lower bound of parallel machines in the supercomputing world.

7. NUMERICAL EXAMPLES

In this section we present several results that were obtained with the developed code. As mentioned, the goal was to show the ability of this vector code to solve the nearly incompressible and compressible Euler and Navier–Stokes equations using an explicit or an implicit solver and covering a broad range of different situations: (a) inviscid flow over a parabolic arc bump at subsonic, transonic and supersonic speed; (b) viscous supersonic flow over a flat plate; (c) viscous supersonic flow over a compression corner; (d) viscous flow over an aerofoil under incompressible, subsonic and transonic conditions; (e) viscous incompressible and subsonic flow over a circular cylinder; (f) viscous incompressible and subsonic flow over a sphere.

7.1. Parabolic arc bump

We start with the inviscid flow over a parabolic arc bump at several Mach numbers ranging from the subsonic regime ($M = 0{\cdot}1$ and $0{\cdot}5$) through the transonic one ($M = 0{\cdot}84$) and finally to the

supersonic case ($M = 1{\cdot}4$). The equation for the bump is

$$y = b[1 - (2x)^2],$$

where b is the ratio between the maximum arc thickness and the length. The computational box contains the bump in the centre of the bottom boundary and it extends L units in front of and behind the bump and H units above it, with both L and H normalized by the arc length. The bottom boundary satisfies the no-penetration condition and the other three boundaries have the absorbent boundary condition, except for the supersonic case where the top boundary also has the no-penetration condition.

7.1.1. Mach numbers 0·1 and 0·5. For this case we use $L = 8$ and $H = 3$ with $b = 5\%$. The mesh consisted of 12 elements in the y-direction and 47 elements in the x-direction, giving a total of 564 elements. Figure 1 shows the pressure coefficient at $M = 0{\cdot}1$ using an explicit solver and Figure 2 plots the same for $M = 0{\cdot}5$ using the GMRES solver. Both results are in good agreement with *thin body theory*.[31] Figure 3 shows the rate of convergence of GMRES using absorbent boundary conditions. We see a decrease in residual norm by eight orders of magnitude in 170 outer iterations.

7.1.2. Mach number 0·84. In this example we have adopted a freestream Mach number of 0·84 and have used the same mesh as in the previous case. This flow develops a shock wave over the surface of the bump, producing a supersonic region in part of the domain.

Figure 4 shows the pressure coefficient over part of the bottom boundary, especially concentrated on the bump surface. We can note the formation of the shock at around $x = 0{\cdot}3$ and this shock was captured by only two elements without any overshoot or undershoot effect. This result has been successfully compared with another numerical computation.[3]

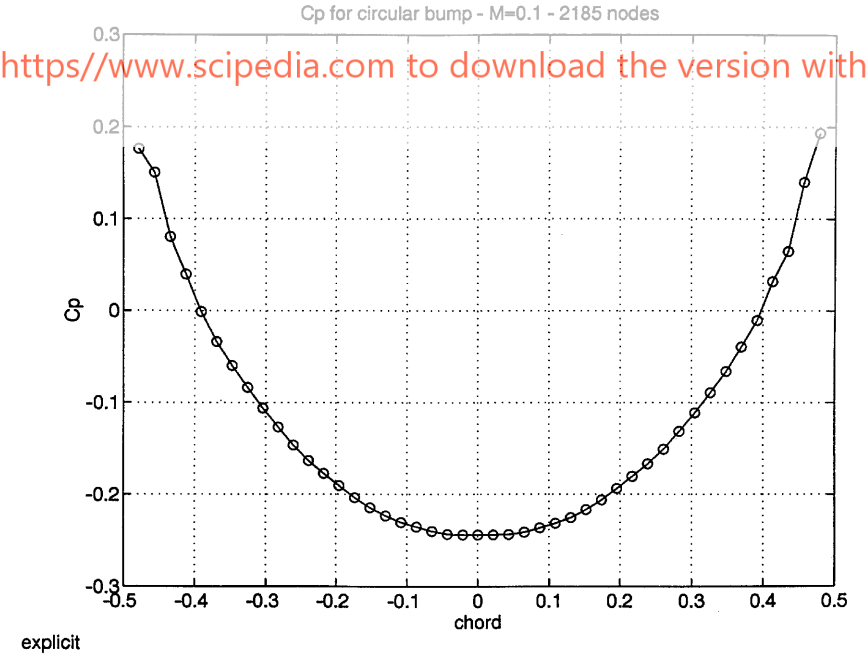

Figure 1 C_p for parabolic arc bump at $M = 0{\cdot}1$

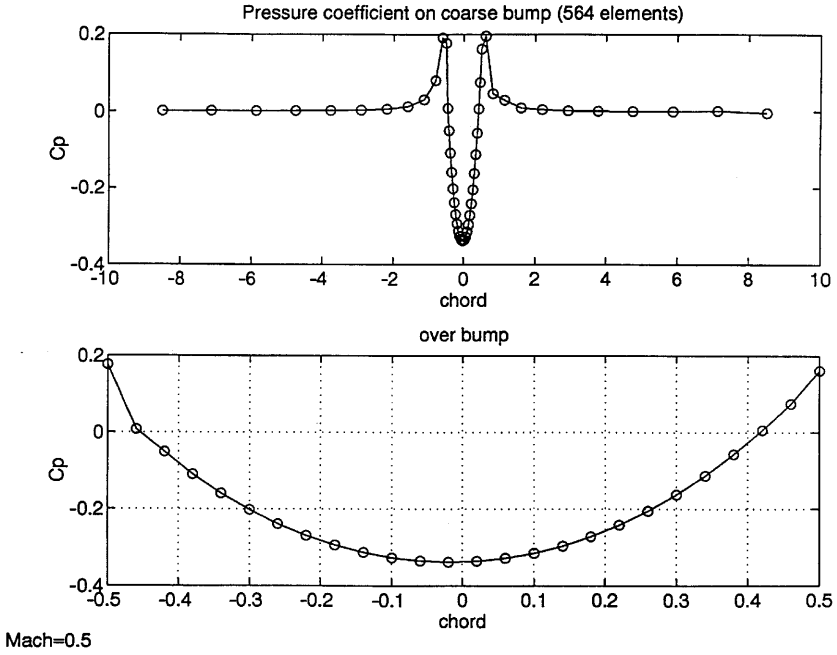

Figure 2 C_p for parabolic arc bump at $M = 0{\cdot}5$

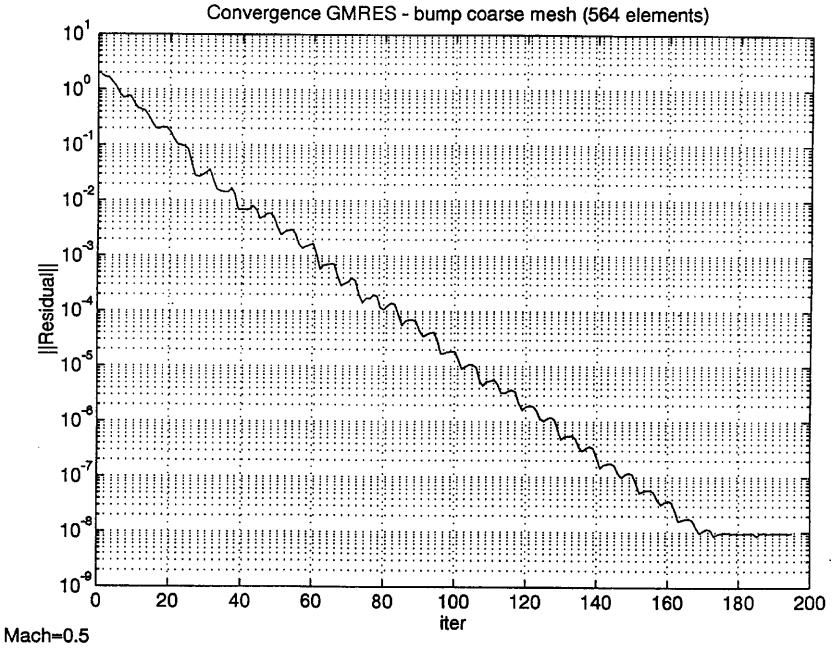

Figure 3. GMRES convergence rate for parabolic arc bump at $M = 0{\cdot}5$

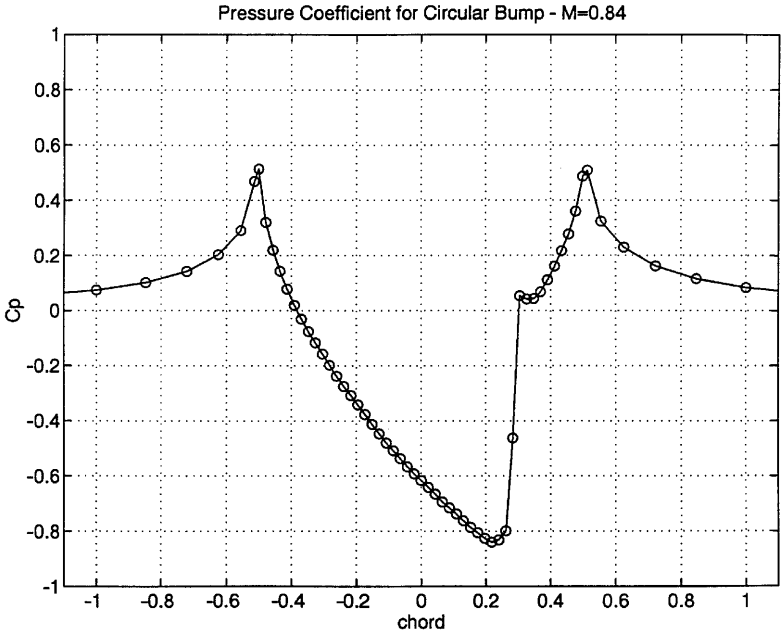

Figure 4. C_p for parabolic arc bump at $M=0{\cdot}84$

7.1.3. Mach number 1·4. This supersonic flow consists of a 1·4 freestream Mach number impinging over a thinner bump. In this example we have used $b=4\%$ and have changed the mesh, taking $L=H=1$. The mesh had 30 elements in the y-direction and 92 elements in the x-direction, giving a total of 2760 elements. As we can see in Plate 1 from the pressure isocurves, the problem consists of a leading edge shock that reflects off the upper boundary where we have imposed the no-penetration boundary condition, crosses the trailing edge shock and reflects again, merging finally with the trailing edge shock. Figures 5 and 6 show the density profiles at $y=0{\cdot}5$ and 1 respectively. We can note that the formation captures well the expansion waves caused by the curvature in the arc and the decrease in strength of the reflected shock caused by the expansion waves. As in the transonic example, no undershoot or overshoot was found in all the discontinuities appearing here.[3]

7.2. *Flow over a flat plate*

The second problem, called the Carter problem,[32] consists of a viscous flow over an infinitely thin flat plate at zero angle of attack in the supersonic regime ($M=3$), with a Reynolds number of 1000 based on the freestream values and the distance from the leading edge of the plate. In this case we show the behaviour of the code in problems where a strong curved shock interacts with a boundary layer starting at the leading edge of the plate. We have used the Sutherland viscosity law

$$\mu = \frac{0{\cdot}0906\theta^{1{\cdot}5}}{\theta + 0{\cdot}0001406}$$

corresponding to a freestream temperature of 216·7 K.

The computational domain covers the area $-0{\cdot}2 \leqslant x \leqslant 1{\cdot}2$ and $0 \leqslant y \leqslant 0{\cdot}8$, with the leading edge of the plate placed at $x=0$. On the inflow and top boundaries we have imposed all the variables. On

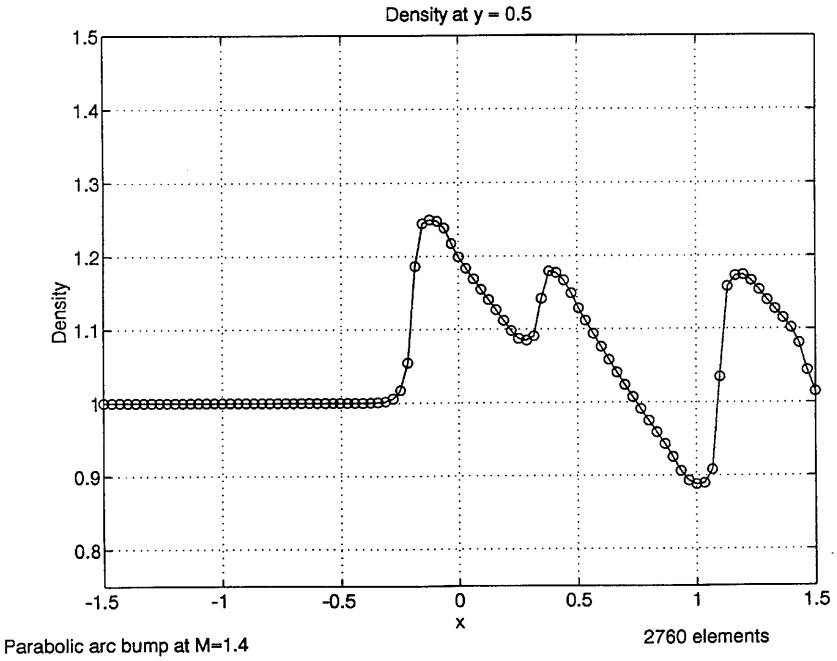

Figure 5. $\rho(x)|_{y-1/2}$ for parabolic arc bump at $M = 1{\cdot}40$

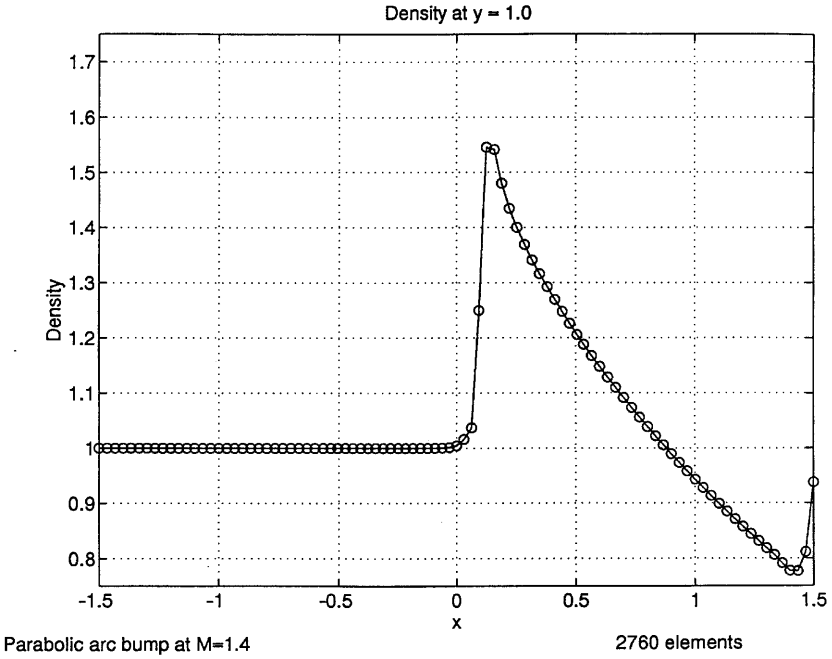

Figure 6. $\rho(x)|_{y=1}$ for parabolic arc bump at $M = 1{\cdot}40$

the symmetry line ($y = 0, x < 0$) we have imposed symmetric conditions, along the plate the non-slip condition and the stagnation temperature

$$\theta_{\text{stag}} = \theta_{\text{inflow}}\left(1 + \frac{\gamma - 1}{2} M^2_{\text{inflow}}\right)$$

were prescribed and, finally, on the outflow boundary all the variables were free.

The mesh used to solve this problem consisted of 32 elements in the y-direction and 56 elements in the x-direction, giving a total of 1792 elements. Plate 2 shows the density isocurves and Figure 7 plots the velocity profiles at several x-stations. In both Plate 2 and Figure 7 we can visualize the development of the boundary layer and the shock wave. Figure 8 shows the distribution of the pressure along the wall relative to its freestream value and Figure 9 plots the skin friction computed using our code (circles) and using the modified Blasius formula (full curve).[2] Both figures show reasonable agreement with Carter's results.[32] Figure 10 shows the x-velocity profile at the outlet boundary as presented in Carter's paper.[32] Finally, in Figure 11 the rate of convergence of this problem using the GMRES solver is presented.

7.3. Compression corner

The third problem is viscous flow over a compression corner at $M = 3$ and $Re = 16{,}800$ where there is an additional complexity compared with the flat plate problem because it involves separated flow close to the corner and reattachment produced by a compression fan.[32,33] This viscous problem consists of a Mach 3 flow passing over a compression corner at an angle of 10°. The Reynolds number, based on the freestream values and the distance from the leading edge of the plate to the corner, is 16,800. The same Sutherland law as in the plate example was used. The computational domain covers the area $-0{\cdot}2 \leqslant x \leqslant 1{\cdot}8$ and $0 \leqslant y \leqslant 0{\cdot}59$ on the plate and a height of 0·59 is kept above the wall past the corner. The leading edge is placed at $x = 0$ and the corner at $x = 1$. On the inflow and top boundaries all the variables were fixed, we have applied symmetry boundary conditions at the boundary at $y = 0$ and $x < 0$ and we have imposed the no-slip condition on the wall

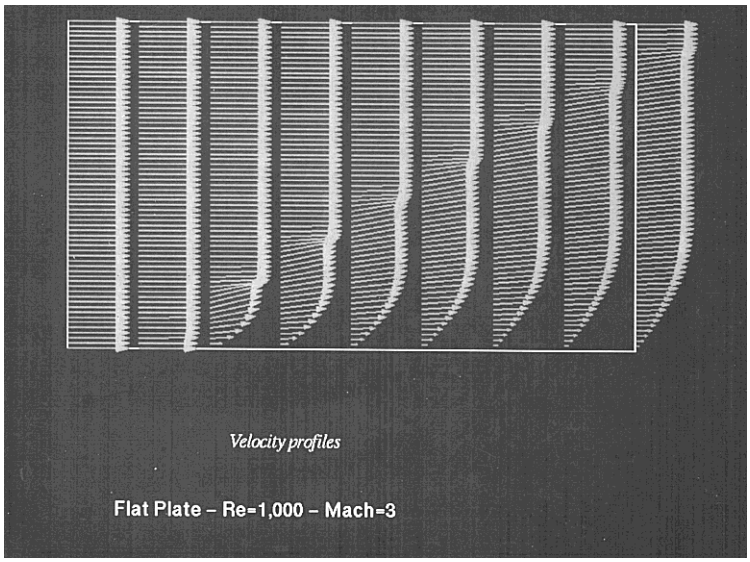

Figure 7. Velocity field for flat plate at $M = 3$ and $Re = 1000$

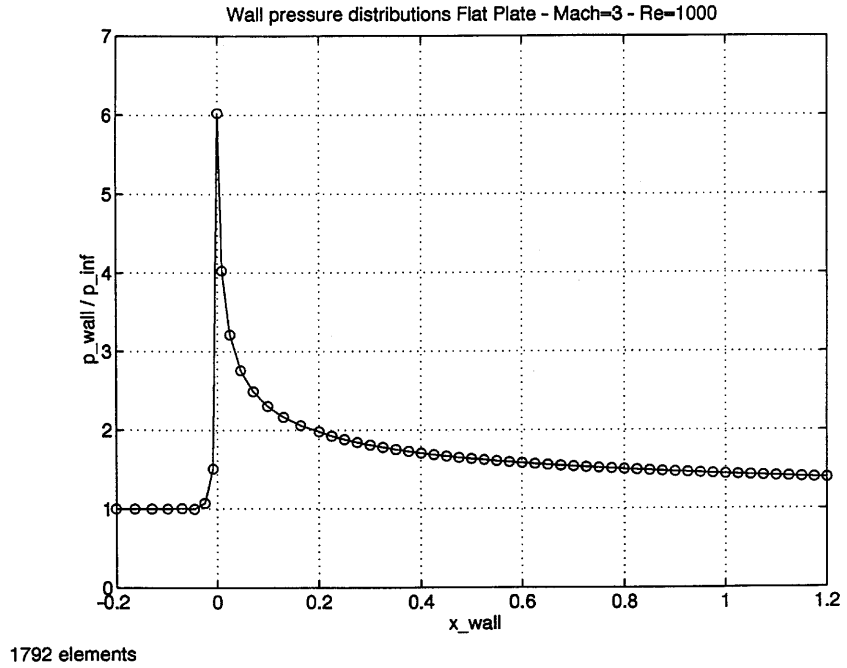

Figure 8. Pressure distribution over flat plate at $M = 3$ and $Re = 1000$

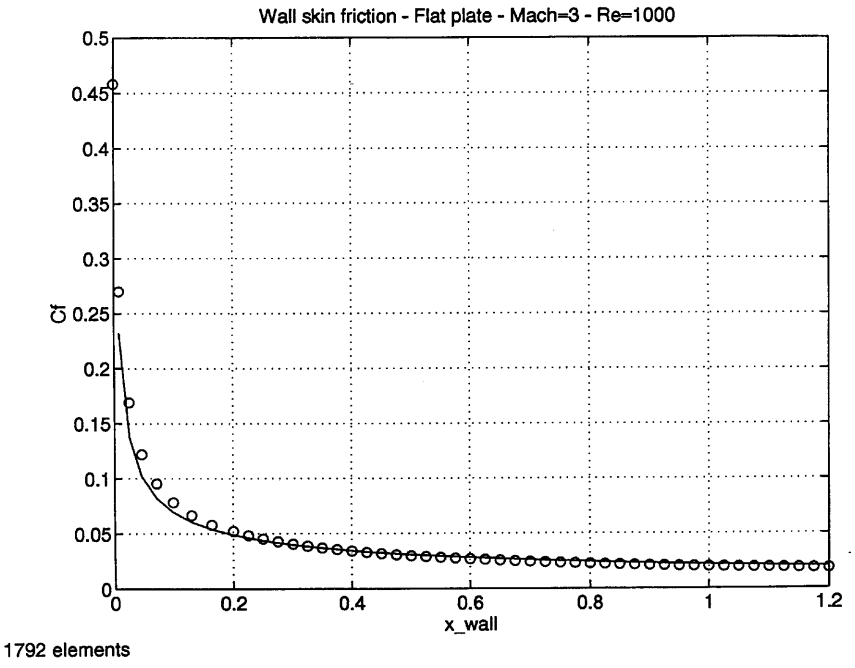

Figure 9. Skin friction over flat plate at $M = 3$ and $Re = 1000$

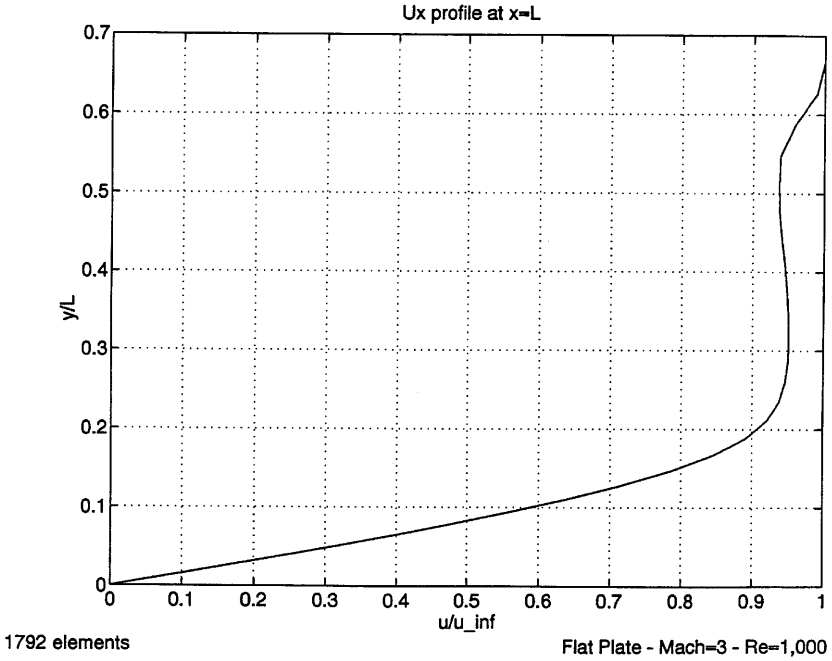

Figure 10. Outlet u_x-profile for flat plate at $M=3$ and $Re=1000$

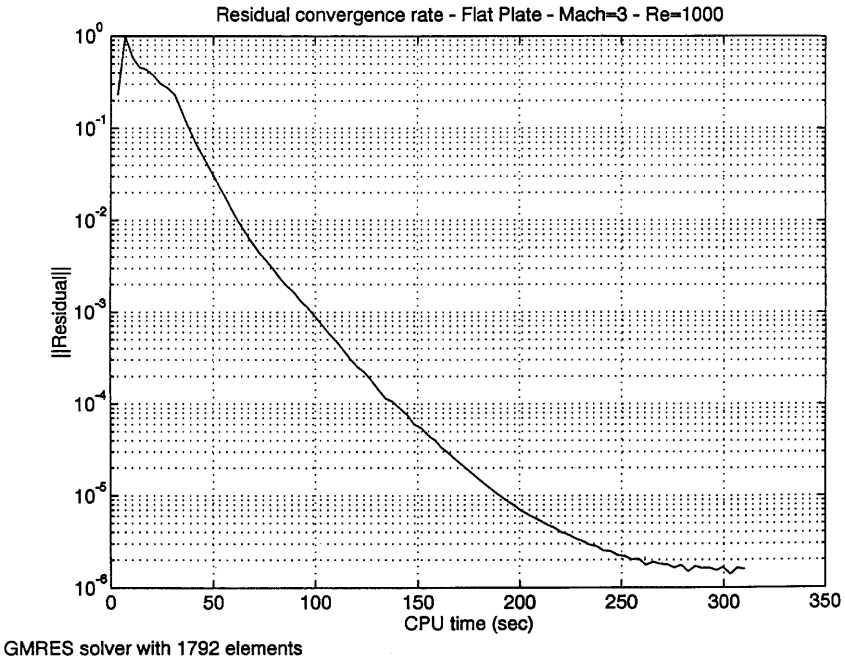

Figure 11. GMRES convergence rate for flat plate at $M=3$ and $Re=1000$

with a fixed stagnation temperature. No condition is imposed on the outflow boundary. To solve this problem, we have used 39 elements in the y-direction and 104 elements in the x-direction, giving a total of 4056 elements. In order to capture the recirculation close to the corner, we have used a grading function in the cross-flow direction similar to Carter's experiments,[32] getting $\Delta y_{\min} = 6 \times 10^{-4}$.

According to Carter's description, we plot in Plate 3 the density isocurves that show some interesting features of this kind of fluid flow. The supersonic flow over a plate followed by a 10° degree ramp has attracted the attention of fluid dynamicists for years. The present rise generated by the ramp upstream along the flat plate results in a complex interaction between the boundary layer and the outer inviscid stream. This interaction leads to flow separation for certain ranges of Mach numbers, Reynolds number and ramp angle. In addition to its theoretical interest, the problem is of practical importance in predicting the pressure and heat loads on a wing–flap junction on a supersonic aircraft, reducing the flap effectiveness when separation occurs with severe heating of surfaces in the reattachment region. Figure 12 shows how the boundary layer is thickened by the compression ramp through the adverse pressure gradient. The reduction of the total pressure within the boundary layer can produce a separation from the surface without overcoming the adverse pressure gradient. The separated boundary layer becomes a free shear layer external to a steady and recirculating inner flow near the corner. This recirculation is evident in Figure 13 through the sign change of the skin friction or by the streamlines in Plate 3. Downstream the shear layer impinges on the ramp in the reattachment region; the flow accelerates until the boundary layer reaches a minimum thickness at the neck, returning to its normal state downstream of the neck but at a new Mach number. There is also a shock wave that appears at the leading edge and separates from the solid surface as in the previous flat plate example, but it now has a stronger interaction with the boundary layer through the separation and reattachment compression fan produced close to the corner. Figure 12 shows the

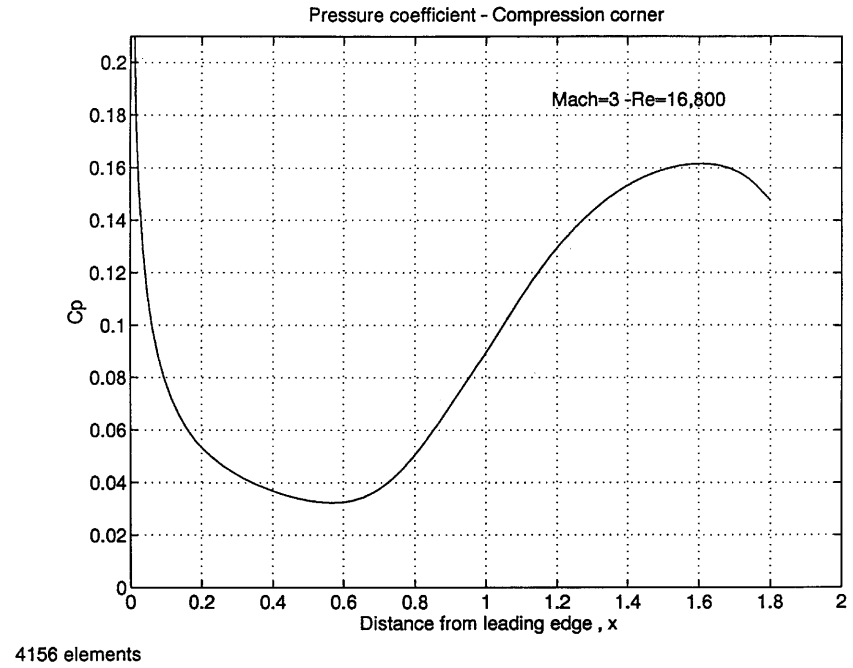

Figure 12. Pressure coefficient for compression corner at $M = 3$ and $Re = 16{,}800$

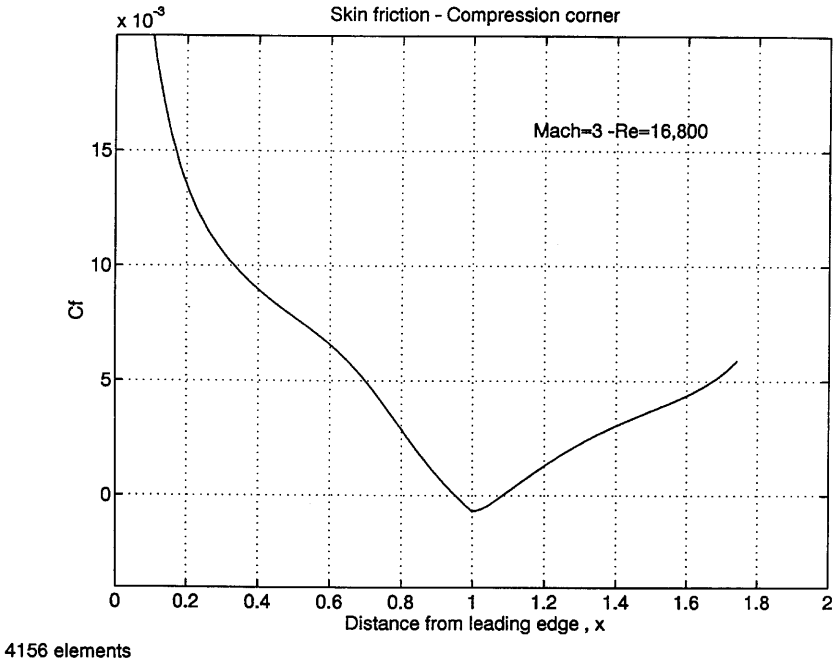

Figure 13. Skin friction for compression corner at $M = 3$ and $Re = 16{,}800$

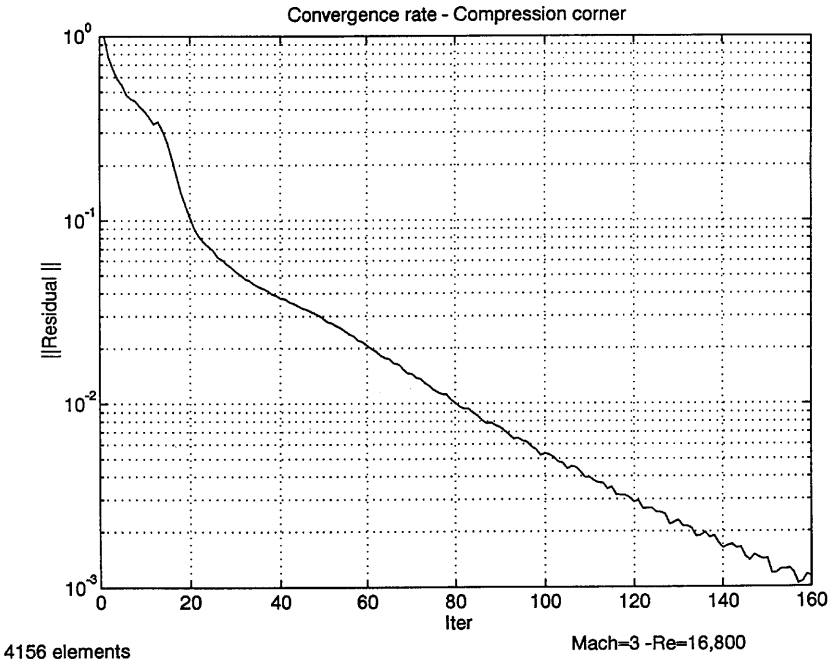

Figure 14. GMRES convergence rate for compression corner at $M = 3$ and $Re = 16{,}800$

pressure coefficient distribution over the wall from leading edge ($x = 0$) through the corner at $x = 1$ until the outlet of the computational box at $x = 1{\cdot}8$. In Figure 13 we plot the skin friction factor. Both these important parameters are in good agreement with the values reported by Shakib[11] for a similar mesh and very close to the experimental ones reported by Carter[32] and Hung and McCormack.[33] Increasing the refinement close to the wall will improve these results. Figure 14 plots the rate of convergence of the GMRES solver for this problem.

7.4. NACA 0012 aerofoil

We start this series of NACA 0012 numerical simulations with an inviscid, nearly incompressible flow where the efficiency of the code is checked by the convergence rate and the accuracy of the total pressure coefficient. The mesh employed for this example consists of a C-grid with 39 nodes on the aerofoil, 20 nodes along the wake and 31 nodes in the direction normal to the streamwise direction, with an aerofoil chord size equal to two units and the far-field boundary placed at more than 28 units. The angle of attack was zero and we meshed the lower and the upper side of the aerofoil. The boundary conditions used were the absorbent boundary condition on the far-field boundary and no-slip velocities and zero heat flux along the profile. Figure 15 shows the rate of convergence of each of the five variable increments (left) and the residual (right). We can note a reduction of seven orders of magnitude in the left plot, with a uniform convergence rate for all the equations, and five orders in the right plot. This kind of behaviour is only possible by the use of an optimal preconditioner.[1]

It is important to observe the stagnation in the residual convergence after 130 iterations produced by round-off errors. Figure 16 shows the pressure coefficient (left) and the total pressure coefficient (right), two important parameters to validate the code for this regime. The theoretical solution of this problem should produce a constant total pressure coefficient equal to one, a symmetric C_p with an increment of the pressure at the leading edge equal to the dynamic pressure of the freestream ($C_p = 1$). We can observe that our preconditioned code produces accurate results without numerical

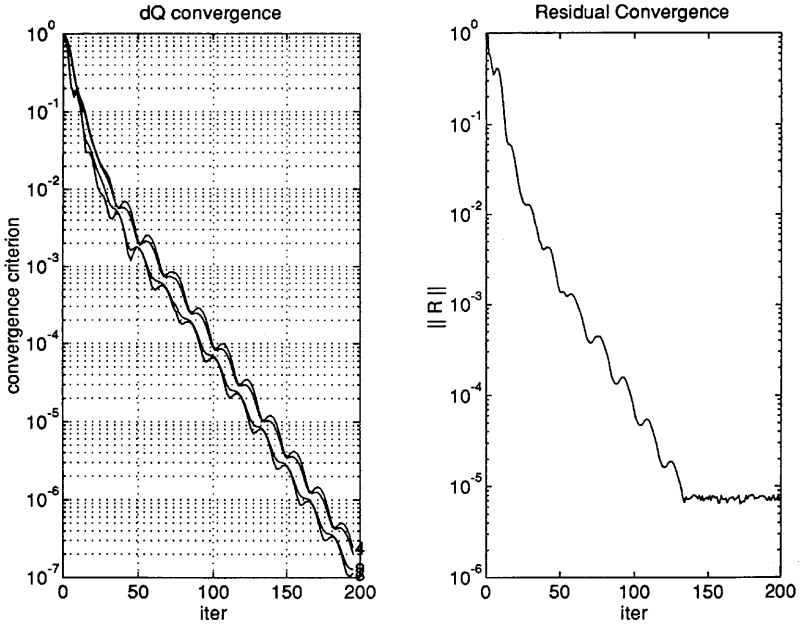

Figure 15. Rate of convergence for NACA 0012 aerofoil at $M = 0{\cdot}001$

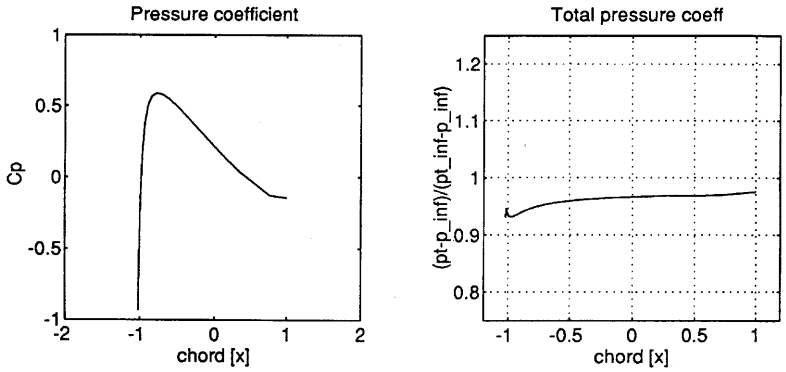

Figure 16. C_p and total C_p for NACA 0012 aerofoil at $M = 0{\cdot}001$

oscillations, while the standard SUPG formulation exhibits such drawbacks produced by incompressibility effects. The second case was a viscous subsonic flow and Figure 17 shows a rate of convergence of nine orders of magnitude in 350 outer iterations for the increment of each of the five variables (left), with an equivalent reduction in the residual (right). This simple example was included to show that the good behaviour of the preconditioner is retained also for a set of parameters that does not present difficulties in the non-preconditioned case.

Finally we run a viscous transonic case that consists of a flow at Mach 0·85 approaching an NACA 0012 aerofoil at zero angle of attack. The Reynolds number based on the aerofoil chord is 500 using a constant viscosity. We have used a C-mesh with 158 elements in the chordwise direction and 39 elements in the normal direction, giving a total of 6162 elements. We have imposed all the variables

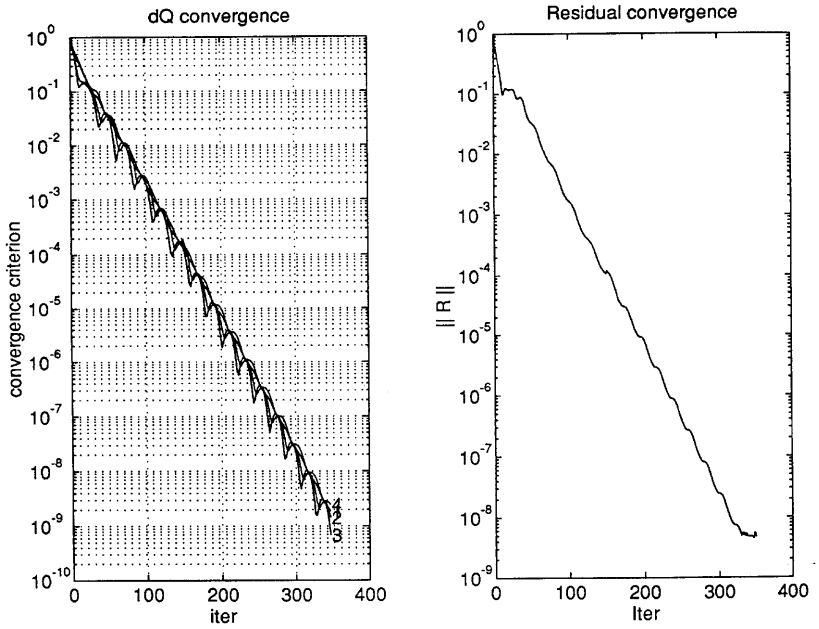

Figure 17. Rate of convergence for NACA 0012 aerofoil at $M = 0{\cdot}3$ and $Re = 200$

at the inlet boundary without imposing any at the outflow, using a no-slip boundary condition over the aerofoil with a prescribed temperature. Plates 4 and 5 display the density and temperature isocurves respectively, showing a symmetric pair of detached supersonic pockets followed by a thick wake behind the aerofoil. Figures 18 and 19 show the pressure coefficient and the skin friction over the chord of the aerofoil respectively. We have found good agreement with the results reported by Shakib.[11]

7.5. *Circular cylinder*

The circular cylinder problem is a very good test because we have numerous approaches for its solution. There is an analytical exact solution for the Stokes equation in creeping incompressible flow ($RE \to 0$) and some analytical approximations for low Reynolds numbers and lightly compressible flow[34] that allows us to validate our code. For this problem, $Re < 40$ based on the diameter of the cylinder is the critical point for stationary flow. We mesh the entire domain using two zones: the first one, close to the body, is built with an O-mesh with 80 nodes around the cylinder and 20 nodes in the radial direction; the second zone is an H-mesh with 20 nodes in the downstream direction. The diameter of the cylinder is one unit length and we have placed the inlet, upper and lower boundaries at nine units and the outlet boundary at 27 units. The boundary conditions were no slip around the cylinder with zero heat flux and we have imposed all the variables at the inlet, upper and lower boundaries while using free traction and zero heat flux at the outlet. We start with nearly incompressible flow at $M = 0{\cdot}001$ and $Re = 1$ and 20. Again the low Mach number employed here is a very strong condition to check the convergence efficiency. Figures 20 and 21 show the convergence rate of each variable increment (left) and residual (right) for the two Reynolds numbers. A constant slope in the residual of almost six orders of magnitude in 300 outer iterations seems to be evidence of the efficiency of the preconditioner.[1] The next simulation deals with a subsonic flow at $M = 0{\cdot}27$ and

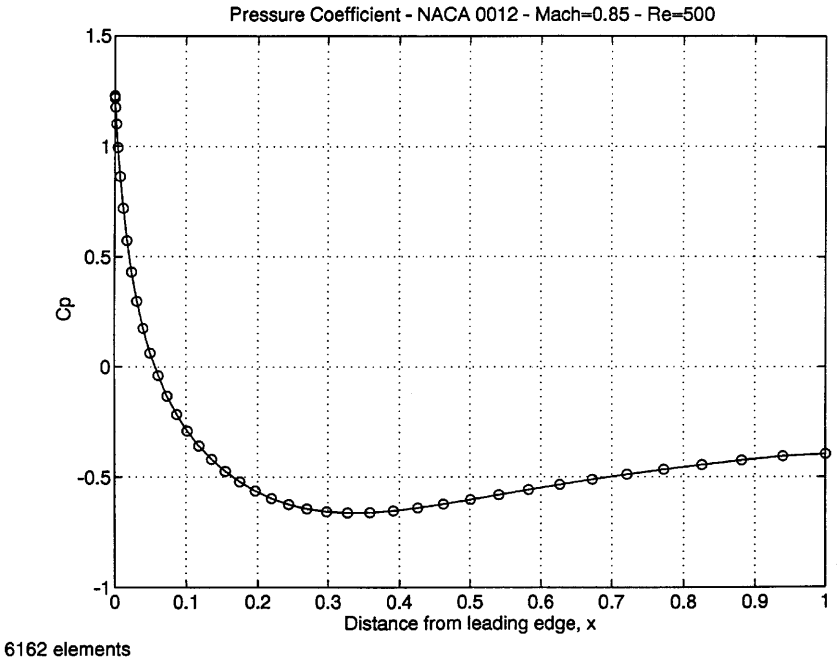

Figure 18. Pressure coefficient for NACA 0012 aerofoil at $M = 0{\cdot}85$ and $Re = 500$

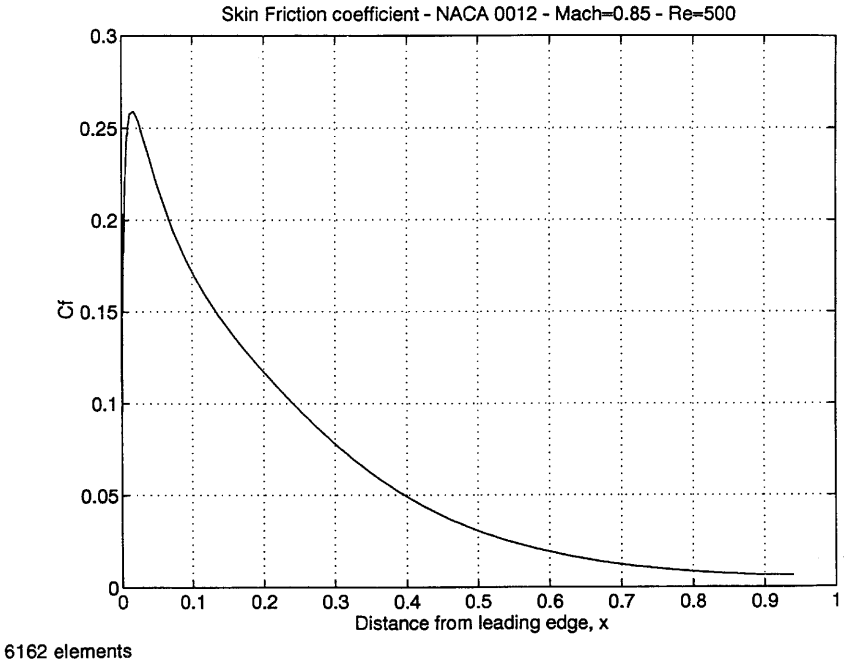

Figure 19. Skin friction for NACA 0012 aerofoil at $M = 0{\cdot}85$ and $Re = 500$

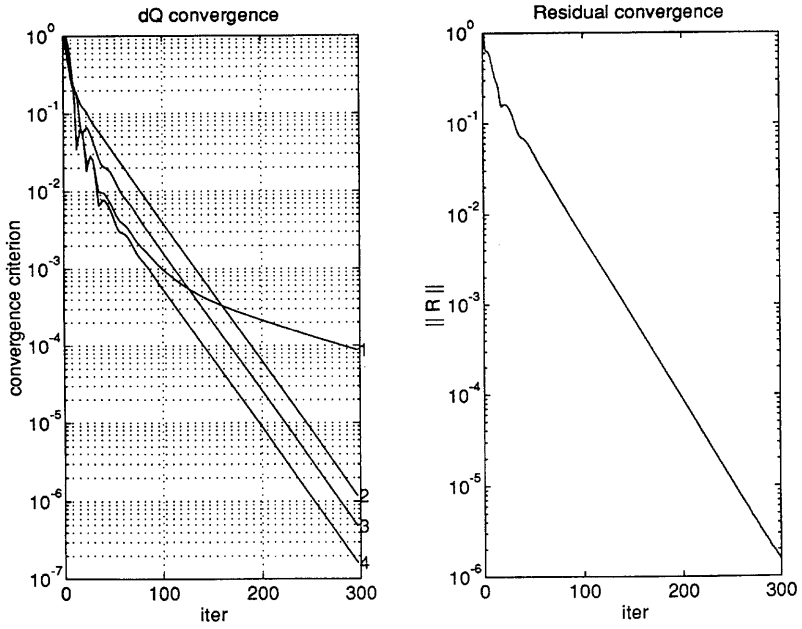

Figure 20. Rate of convergence for circular cylinder at $M = 0{\cdot}001$ and $Re = 1$

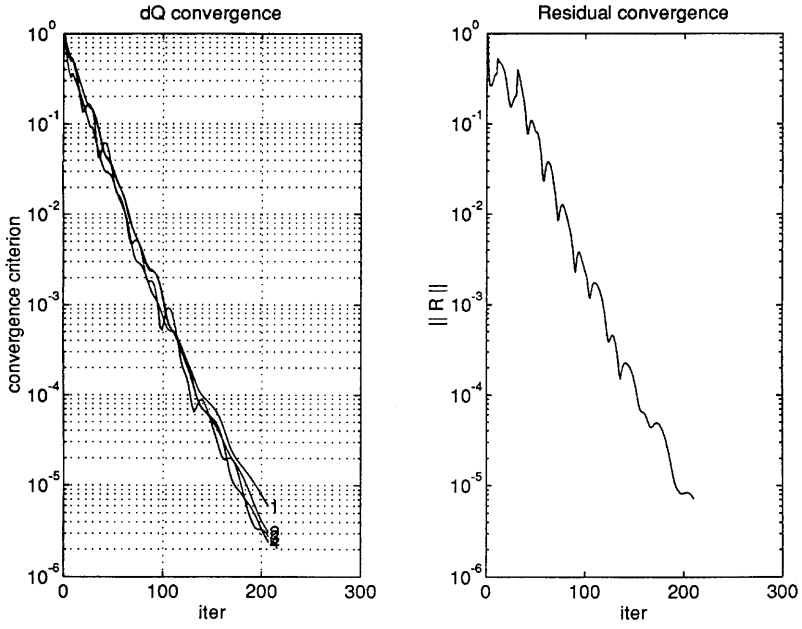

Figure 21. Rate of convergence for circular cylinder at $M = 0{\cdot}001$ and $Re = 20$

$Re = 10$ that was run using the same mesh. This Reynolds number based on the diameter of the cylinder is very close to the beginning of the formation of the recirculation zone at the rear part of the body. Plate 6 shows the pressure isocurves and downstream velocity field in the same plot. We can note the symmetry in the results and the stagnation of the fluid immediately behind the cylinder. The last example for the circular cylinder consists of increasing the Reynolds number to 20 in order to test the formation of the recirculation pocket and its length. Plate 7 shows the streamlines produced by the velocity field. The size of the pair of vortices is in good agreement with analytical approximations and experiments.[34]

7.6. Sphere

The last example of this paper is flow around a sphere. The goal of this simulation is to show the efficiency of the code in a three-dimensional case. We have used a mesh composed of 19,942 nodes and 15,840 elements distributed inside a computational domain represented by a box with $-3 \leqslant x \leqslant 10$, $-3 \leqslant y \leqslant 3$, $-4 \leqslant z \leqslant 3$ and the sphere diameter is equal to one. Boundary conditions were similar to the circular cylinder case. We begin by showing the rate of convergence in nearly incompressible flow. Figure 22 plots the residual convergence for $M = 0{\cdot}002$ and $Re = 10$ (left) and 100 (right). The almost constant residual decrease of six orders of magnitude in approximately 150 iterations shows again the optimal behaviour of the preconditioner in such a severe condition. Plate 8(a) plots the distribution of the pressure field over the sphere at $M = 0{\cdot}002$ and $Re = 100$, showing the peak of the pressure in the zone where the fluid is impinging on the body (pole of the sphere), the suction zone at the equatorial plane and the partial recovery of the pressure behind the sphere. In Plate 8(b) we can see the streamlines at $z = 0$ for the same conditions. The critical point for this stationary flow is approximately $Re < 300$. The accuracy of the recirculation region size is good according to several references.[34,35] Finally we have extended the simulation to low subsonic flow at $M = 0{\cdot}2$ for several Reynolds numbers. Here we show only the results for $Re = 30$ and 100. Plates

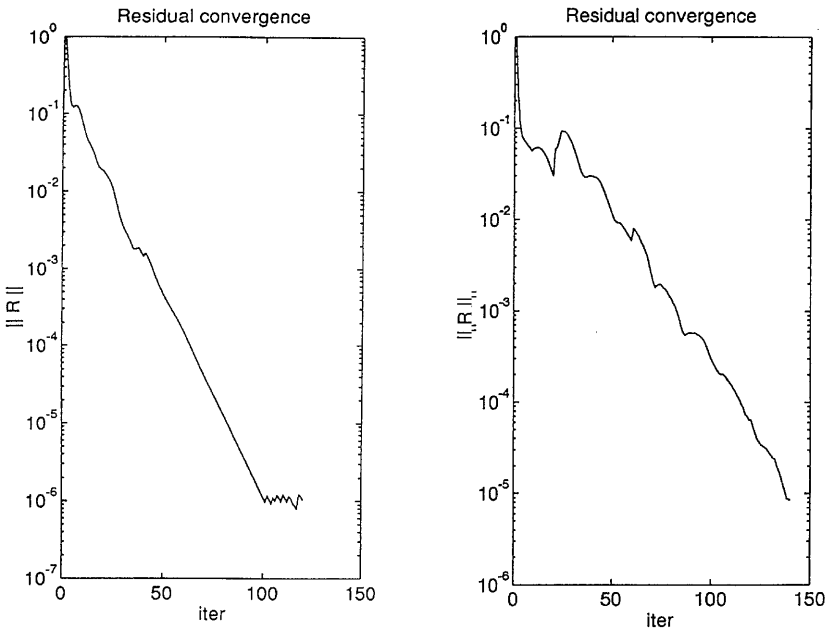

Figure 22. Residual convergence for sphere at $M = 0{\cdot}002$ and $Re = 10$ (left) and 100 (right)

9(a) and 9(b) show the streamlines for $Re = 30$ and 100 respectively that were checked with other numerical results.[2]

8. CONCLUSIONS

In this work we have presented an efficient vector code (close to 1 Gflop) for solving the compressible and incompressible Euler and Navier–Stokes equations. We have applied the software to different kinds of interesting problems in order to validate its usage for very challenging applications.

ACKNOWLEDGEMENTS

The authors wish to express their gratitude to Consejo Nacional de Investigaciones Científicas y Técnicas (CONICENT, Argentina) for its financial support and Tayfun Tezduyar for his guidance at Minnesota Supercomputer Institute. Cray C90 time was provided by the University of Minnesota Supercomputer Institute.

REFERENCES

1. N. Nigro, M. Storti, S. Idelsohn and T. Tezduyar, 'Physics based GMRES preconditioner for compressible and incompressible Navier–Stokes equations', *Comput. Methods Appl. Mech. Eng.*, in press (1997).
2. S. Aliabadi, 'Parallel finite element computations in aerospace applications', *Ph.D. Thesis*, Department of Aerospace Engineering and Mechanics, University of Minnesota, 1994.
3. S. Aliabadi, S. Ray and T. Tezduyar, 'SUPG finite element computation of viscous compressible flows based on the conservation and entropy variables formulations', *Comput. Mech.*, **11**, 300–312 (1993).
4. C. Baumann, M. Storti and S. Idelsohn, 'A Petrov-Galerkin technique for the solution of transonic and supersonic flows', *Comput. Methods Appl. Mech. Eng.*, **95**, 49–70 (1992).

5. A. Brooks and T. Hughes, 'Streamline upwind Petrov–Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier–Stokes equations', *Comput. Methods Appl. Mech. Eng.*, **32**, 199–259 (1982).
6. T. Hughes, M. Mallet and A. Mizukami, 'A new finite element method for CFD: II. Beyond SUPG', *Comput. Methods Appl. Mech. Eng.*, **54**, 341–355 (1986).
7. T. Hughes and M. Mallet, 'A new finite element method for CFD: III. The generalised streamline operator for multidimensional advection–diffusion systems', *Comput. Methods Appl. Mech. Eng.*, **58**, 305–328 (1986).
8. T. Hughes and M. Mallet, 'A new finite element method for CFD: IV. A discontinuity-capturing operator for multidimensional advective-diffusive systems', *Comput. Methods Appl. Mech. Eng.*, **58**, 329–336 (1986).
9. G. Le Beau, S. Ray, S. Aliabadi and T. Tezduyar, 'SUPG finite element computation of compressible flows with the entropy and conservation variables formulations', *Comput. Methods Appl. Mech. Eng.*, in press.
10. M. Mallet, 'A finite element method for CFD', *Ph.D. Thesis*, Department of Civil Engineering, Stanford University, 1985.
11. F. Shakib, 'Finite element analysis of the compressible Euler and Navier–Stokes equations', *Ph.D. Thesis*, Department of Mechanical Engineering, Stanford University, 1988.
12. A. Soulaimani and M. Fortin, 'Finite element solution of compressible viscous flows using conservative variables', *Comput. Methods Appl. Mech. Eng.*, **118**, 319–350 (1994).
13. T. Tezduyar and T. Hughes, 'Finite element formulations for convection dominated flows with particular emphasis on the compressible Euler equations', *AIAA Paper 83-0125*, 1983.
14. T. Hughes and T. Tezduyar, 'Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible Euler equations', *Comput. Methods Appl. Mech. Eng.*, **45**, 217–284 (1984).
15. L. Franca and E. Dutra do Carmo, 'The Galerkin–gradient least squares method', *Comput. Methods Appl. Mech. Eng.*, **74**, 41–54 (1989).
16. T. Hughes and L. Franca, 'A new finite element formulation for computational fluid dynamics: VII. The Stokes problem with various well-posed boundary conditions: symmetric formulations that converge for all velocity/pressure spaces', *Comput. Methods Appl. Mech. Eng.*, **65**, 85–96 (1987).
17. T. Hughes, L. France and G. Hulbert, 'A new finite element formulation for computational fluid dynamics: VIII. 'The Galerkin/least square method for advective–diffusive equations', *Comput. Methods Appl. Mech. Eng.*, **73**, 173–189 (1989).
18. Y. Saad, 'A flexible inner–outer preconditioned GMRES algorithm', *SIAM J. Sci. Comput.*, **14**, 461–469 (1993).
19. Y. Saad and M. Schultz, 'GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems', *SIAM J. Sci. Stat. Comput.*, **7**, 856–869 (1986).
20. M. Storti, C. Baumann and S. Idelsohn. 'A preconditioning mass matrix to accelerate the convergence to the steady Euler solution using explicit schemes', *Comput. Methods Appl. Mech. Eng.*, **34**, 519–541 (1992).
21. M. Storti, N. Nigro and S. Idelsohn, 'Steady state incompressible flows using explicit schemes with an optimcal local preconditioning', *Comput. Methods Appl. Mech. Eng.*, **124**, 231–252 (1995).
22. Y. Choi and C. Merkle, 'The application of preconditioning in viscous flows', *J. Comput. Phys.* **105**, 207–223 (1993).
23. S. Venkateswaran, J. Weiss, C. Merkle and Y. Choi, 'Propulsion-related flowfields using the preconditioned Navier–Stokes equations', *AIAA Paper 92–3437*, 1992.
24. N. Nigro, M. Storti and S. Idelsohn, 'Fluid flows around turbomachinery using an explicit pseudo-temporal Euler FEM', *Commun. Appl. Numer. Methods*, **11**, 199–211 (1995).
25. N. Nigro, M. Storti and S. Idelsohn, 'Two-phase flow in gas-stirred liquid vessels with SUPG-stabilized equal order interpolations', *Int. j. numer. methods fluids*, **19**, 1–22 (1994).
26. M. Storti, N. Nigro and S. Idelsohn, 'Equal-order interpolations: a unified approach to stabilize the incompressible and convective effects', *Comput. Methods Appl. Mech. Eng.*, **143**, 3–4, 317, 331 (1997).
27. V. van Leer, W. Lee and P. Roe, 'Characteristic time-stepping or local preconditioning of the Euler equations', *AIAA Paper 91-1552-CP*, 1991.
28. B. Gustafsson and A. Sundstrom, 'Incompletely parabolic problems in fluid dynamics', *SIAM J. Appl. Math.*, **35**, 343–357 (1978).
29. J. Oliger and A. Sundstrom, 'Theoretical and practical aspects of some initial boundary value problems in fluid dynamics'. *SIAM J. Appl. Math.*, **35**, 419–446 (1978).
30. D. Rudy and J. Strikwerda, 'A nonreflecting outflow boundary condition for subsonic Navier–Stokes calculations', *J. Comput. Phys.*, **36**, 55–70 (1980).
31. H. Ashley and M. Landhal, *Aerodynamics of Wings and Bodies*, General, Toronto, 1965.
32. J. Carter, 'Numerical solution of the Navier–Stokes equations for the supersonic laminar flow over two dimensional compression corner', *NASA TR R-385*, 1972.
33. C. Hung and R. MacCormack, 'Numerical solutions of supersonic and hypersonic laminar compression corner flows', *AIAA J.*, **14**, 475–481 (1976).
34. M. Van Dyke, *Perturbation Methods in Fluid Mechanics*, Parabolic, Stanford, CA, 1975.
35. G. Batchelor, *Introduction to Fluid Dynamics*, Cambridge University Press, Cambridge, 1970.

