INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, VOL. 18, 489–507 (1994)

MULTIGRID PRESSURE CORRECTION TECHNIQUES FOR THE COMPUTATION OF QUASI-INCOMPRESSIBLE INTERNAL FLOWS

P. LUCHINI AND A. D'ALASCIO

Istituto di Gasdinamica, Facoltà di Ingegneria, Università di Napoli, P. le Tecchio, I-80125 Napoli, Italy

SUMMARY

A study is reported on the possibility of improving the speed of convergence of existing numerical programmes for the simulation of flow in combustion chambers by applying the multigrid method to the pressure correction phase only. A version of the multigrid algorithm is introduced for this purpose which achieves a 1 : 10 residual reduction in a single $V(1, 1)$ cycle. The overall decrease in computation time with respect to an industry-standard SIMPLE algorithm with single-grid pressure correction ranges from four to five times for SIMPLE itself and several other well-known algorithms to six times for a newly developed pressure correction strategy we call difference operator triangularization (DOT).

KEY WORDS Multigrid Pressure correction Combustion chamber Navier–Stokes equations DGS

1. INTRODUCTION

Multigrid is today the most promising technique for the fast numerical solution of partial differential equations.[1] Nevertheless, at the industrial application level the majority of the existing computer codes for the calculation of quasi-incompressible internal flows (e.g. in combustion chambers) are still based on single-grid iterative algorithms. These codes have often been developed to a high degree of complexity,[2–4] with the inclusion of the effects of chemical reactions, turbulence (acting on both the velocity field and the chemical reactions themselves) and multiple phases (liquid fuel drops in a gas). The need therefore exists for an incremental multigrid modification capable of achieving a worthwhile reduction in the computation time of such programmes, even if not one as large as could be provided by a multigrid method designed from scratch, while preserving most of the already existing code.

Internal combustion flows are quasi-incompressible in the sense that although important density modifications take place, the Mach number is usually small. Therefore the compressible form of the Navier–Stokes equations must be assumed as the fundamental model of such flows, but the iterative algorithms used for their numerical solution closely resemble those designed for incompressible flow. Among these algorithms, very well known and probably the most widespread in industry is the SIMPLE method of Patankar and Spalding,[5] either in its original form or in one of its variants (SIMPLER, SIMPLEC). A recent competitor is PISO,[6] which may be seen as an evolution of the old MAC of Harlow and Welch[7] and, like MAC, was born as a method for unsteady flows and later adapted to steady flows by a false transient approach.

CCC 0271–2091/94/050489–19

Received June 1993
Revised October 1993

Many recent papers deal with a multigrid modification of one or other of these algorithms; for instance, Miller and Schmidt[8] do so for SIMPLEC, while Shyy *et al.*[9] propose and compare multigrid forms of the SIMPLE, SIMPLEC and PISO methods for the solution of the Navier–Stokes equations.

In all these papers a complete, already functional, iterative algorithm is encased in a multigrid 'outer box' taking the role of the 'smoother' (that block of code in a multigrid strategy which has the purpose of damping the small-wavelength part of the error spectrum). While this approach certainly does improve the convergence rate of the starting algorithm, the iteration block borrowed from a previous method is unlikely to be competitive against a smoother expressly designed for its purpose, e.g. DGS.[10] On the other hand, this approach requires a major rewriting of the computer programme.

Nearly all the well-established iterative algorithms for incompressible flow distinguish a stage in which the fluid velocity is updated at constant pressure from one in which a pressure correction is applied. Whereas the greatest part of the programming effort is required by the algorithm that deals with velocity and other transported quantities (chemical species, thermal energy, turbulence energy or other parameters involved in the turbulence model), the critical step as to computation time generally turns out to be the pressure correction. Starting from this observation, we began to study the possibility of applying a multigrid algorithm to the pressure correction phase only. The results described below show the potential of improving by several times the computational speed of an existing programme while modifying only a small fraction of the code.

2. OVERVIEW OF A FEW STANDARD SINGLE-GRID ITERATION SCHEMES

We shall consider some very-well-known iterative algorithms for the solution of the steady incompressible (or compressible at low Mach number) 2D Navier–Stokes equations and apply them for testing purposes to one and the same discretization of the momentum and continuity equations. Since we are mainly interested in comparing computation times, the same discretization as in the original SIMPLE scheme is adopted, i.e. first-order upwind convective terms and central diffusive terms on a staggered square grid. The adoption of more sophisticated high-order discretizations such as QUICK[11] and a possibly unstaggered and curvilinear grid, one or all of which can be foreseen to be often present in modern applications, is not likely to upset the relative order of magnitude of the computation times in which we are interested here. On the same grounds, we shall conduct our comparison using a simple constant-total-enthalpy condition as the energy equation in the examples below, so that only the continuity and momentum differential equations need be solved for the unknowns velocity and pressure. The results as to relative computation time are likely to remain at least qualitatively valid with more complex and general models.

2.1. *The discretized governing equations*

In the staggered grid we denote by (i, j) the points where the pressure is sampled on the ith row and jth column and by $(i + \frac{1}{2}, j)$ and $(i, j + \frac{1}{2})$ the points where the u- and v-components of velocity are sampled.

The discretized form of the Navier–Stokes equations (u- and v-momentum equations and continuity equation respectively) is represented as

$$a^{(u)}_{i+1/2,j}u_{i+1/2,j-1} + b^{(u)}_{i+1/2,j}u_{i-1/2,j} + c^{(u)}_{i+1/2,j}u_{i+1/2,j} + d^{(u)}_{i+1/2,j}u_{i+3/2,j} + e^{(u)}_{i+1/2,j}u_{i+1/2,j+1} + p_{i+1,j} - p_{i,j} = 0, \tag{1}$$

$$a^{(v)}_{i,j+1/2}v_{i,j-1/2} + b^{(v)}_{i,j+1/2}v_{i-1,j+1/2} + c^{(v)}_{i,j+1/2}v_{i,j+1/2} + d^{(v)}_{i,j+1/2}v_{i+1,j+1/2} + e^{(v)}_{i,j+1/2}v_{i,j+3/2} + p_{i,j+1} - p_{i,j} = 0, \tag{2}$$

$$(\rho u)_{i+1/2,j} - (\rho u)_{i-1/2,j} + (\rho v)_{i,j+1/2} - (\rho v)_{i,j-1/2} = 0, \tag{3}$$

where

$$\begin{aligned}
a^{(u)}_{i+1/2,j} &= -\tfrac{1}{2}[(\rho v)_{i+1/2,j-1/2} + |(\rho v)_{i+1/2,j-1/2}|] - 1/Re_h,\\
b^{(u)}_{i+1/2,j} &= -\tfrac{1}{2}[(\rho u)_{i,j} + |(\rho u)_{i,j}|] - 1/Re_h,\\
d^{(u)}_{i+1/2,j} &= \tfrac{1}{2}[(\rho u)_{i+1,j} - |(\rho u)_{i+1,j}|] - 1/Re_h,\\
e^{(u)}_{i+1/2,j} &= \tfrac{1}{2}[(\rho v)_{i+1/2,j+1/2} - |(\rho v)_{i+1/2,j+1/2}|] - 1/Re_h,\\
c^{(u)}_{i+1/2,j} &= -(a^{(u)}_{i+1/2,j} + b^{(u)}_{i+1/2,j} + d^{(u)}_{i+1/2,j} + e^{(u)}_{i+1/2,j})
\end{aligned}$$

and

$$\begin{aligned}
a^{(v)}_{i,j+1/2} &= -\tfrac{1}{2}[(\rho v)_{i,j} + |(\rho v)_{i,j}|] - 1/Re_h,\\
b^{(v)}_{i,j+1/2} &= -\tfrac{1}{2}[(\rho u)_{i-1/2,j+1/2} + |(\rho u)_{i-1/2,j+1/2}|] - 1/Re_h,\\
d^{(v)}_{i,j+1/2} &= \tfrac{1}{2}[(\rho u)_{i+1/2,j+1/2} - |(\rho u)_{i+1/2,j+1/2}|] - 1/Re_h,\\
e^{(v)}_{i,j+1/2} &= \tfrac{1}{2}[(\rho v)_{i,j+1} - |(\rho v)_{i,j+1}|] - 1/Re_h,\\
c^{(v)}_{i,j+1/2} &= -(a^{(v)}_{i,j+1/2} + b^{(v)}_{i,j+1/2} + d^{(v)}_{i,j+1/2} + e^{(v)}_{i,j+1/2}).
\end{aligned}$$

The velocity components $u_{i,j}$, $u_{i+1,j}$, $v_{i+1/2,j-1/2}$ and $v_{i+1/2,j+1/2}$ necessary for the calculation of the coefficients of the u-momentum equation and $v_{i,j}$, $v_{i+1,j}$, $u_{i-1/2,j+1/2}$ and $u_{i+1/2,j+1/2}$ necessary for the calculation of the coefficients of the v-momentum equation are obtained by linear interpolation between the two nearest grid values. The pressures $p_{i+1/2,j+1/2}$, $p_{i+1/2,j-1/2}$ and $p_{i-1/2,j+1/2}$ necessary for the calculation of the coefficients of the u- and v-momentum equations are obtained by a bilinear interpolation of the four nearest grid values. The density, a known function of the pressure and velocity modulus through the constant-total-enthalpy condition, has been determined at every required point only after interpolating, where necessary, the pressure and velocity. Re_h denotes the cell Reynolds number referred to the constant discretization step h.

2.2. *The velocity predictor step*

All the methods discussed below first derive tentative velocity increments δu and δv by a single step of one amongst many possible iterative matrix inversion schemes (Gauss–Seidel, line relaxation, ADI, incomplete LU factorization, etc.) applied in turn to each of equations (1) and (2) (and, had the model included any, to all other transport equations) with the pressure unchanged and then add a correction by modifying the pressure by an amount δp and the velocity components by amounts $\delta u'$ and $\delta v'$ in a coupled manner and imposing the satisfaction of the continuity equation.

The type of coupling introduced between the velocity and pressure corrections during the second stage is what makes the difference between the various algorithms.

We can describe in a unified manner the solution of the momentum equations (1) and (2) with a given pressure field, necessary to obtain a prediction of the velocity field, because this step is common to all the algorithms of solution of the Navier–Stokes equations discussed in this work. Subsequently we shall describe the algorithm adopted for the correction of the flow field through the continuity equation in each particular method.

For the sake of homogeneity of presentation we shall adopt a linearized delta form in which the object of an interation is the calculation of a correction to the main unknowns (velocity and pressure) from the approximate solution of a set of linearized equations.

The linearized equations (1) and (2) may in all cases be written as

$$\begin{aligned} &a^{(u)}_{i+1/2,j}\delta u_{i+1/2,j-1} + b^{(u)}_{i+1/2,j}\delta u_{i-1/2,j} + (c^{(u)}_{i+1/2,j} + \alpha)\delta u_{i+1/2,j} \\ &\quad + d^{(u)}_{i+1/2,j}\delta u_{i+3/2,j} + e^{(u)}_{i+1/2,j}\delta u_{i+1/2,j+1} = -t^{(u)}_{i+1/2,j}, \end{aligned} \tag{4}$$

$$\begin{aligned} &a^{(v)}_{i,j+1/2}\delta v_{i,j-1/2} + b^{(v)}_{i,j+1/2}\delta v_{i-1,j+1/2} + (c^{(v)}_{i,j+1/2} + \alpha)\delta v_{i,j+1/2} \\ &\quad + d^{(v)}_{i,j+1/2}\delta v_{i+1,j+1/2} + e^{(v)}_{i,j+1/2}\delta v_{i,j+3/2} = -t^{(v)}_{i,j+1/2}, \end{aligned} \tag{5}$$

where the right-hand sides $t^{(u)}_{i+1/2,j}$ and $t^{(v)}_{i,j+1/2}$ denote the residuals of equations (1) and (2).

The added coefficient α may be equivalently interpreted either as a false transient term simulating the effect of time evolution or as a relaxation parameter intended to give an SOR-like character to the scheme and is commonly used as a 'turning knob' to optimize the convergence rate of each method.

2.3. *The SIMPLE method*

In the pressure correction stage of the SIMPLE algorithm the two required relations between the corrections $\delta u'$ and $\delta v'$ of the velocity components and the pressure correction δp are derived by neglecting in the momentum equations (4) and (5) the four satellite terms with coefficients a, b, d and e and assuming that the prediction values have already zeroed the residuals of these equations. (This is not exactly true, because only a single iteration of matrix inversion has been performed upon a matrix that represents an approximate linearized version of the problem.) The relations are the following:

$$\delta u'_{i+1/2,j} = -\frac{1}{c^{(u)}_{i+1/2,j} + \alpha}(\delta p_{i+1,j} - \delta p_{i,j}), \tag{6}$$

$$\delta v'_{i,j+1/2} = -\frac{1}{c^{(v)}_{i,j+1/2} + \alpha}(\delta p_{i,j+1} - \delta p_{i,j}). \tag{7}$$

Substituting equations (6) and (7) into the discretized form of the continuity equation (3) gives

$$a^{(p)}_{i,j}\delta p_{i,j-1} + b^{(p)}_{i,j}\delta p_{i-1,j} + c^{(p)}_{i,j}\delta p_{i,j} + d^{(p)}_{i,j}\delta p_{i,j-1} + e^{(p)}_{i,j}\delta p_{i,j+1} = -t^{(p)}_{i,j}, \tag{8}$$

where

$$a^{(p)}_{i,j} = -\frac{\rho_{i,j-1/2}}{c^{(v)}_{i,j-1/2} + \alpha}, \qquad b^{(p)}_{i,j} = -\frac{\rho_{i-1/2,j}}{c^{(u)}_{i-1/2,j} + \alpha}, \qquad d^{(p)}_{i,j} = -\frac{\rho_{i+1/2,j}}{c^{(u)}_{i+1/2,j} + \alpha},$$

$$e^{(p)}_{i,j} = -\frac{\rho_{i,j+1/2}}{c^{(v)}_{i,j+1/2} + \alpha}, \qquad c^{(p)}_{i,j} = -(a^{(p)}_{i,j} + b^{(p)}_{i,j} + d^{(p)}_{i,j} + e^{(p)}_{i,j}).$$

On the right-hand side of equation (8) $t_{i,j}^{(p)}$ denotes the residual of equation (3) as evaluated after the predictor step.

One iteration of the SIMPLE method is composed of a velocity prediction as described in Section 2.2 and a pressure correction obtained from an approximate solution of equation (8) (generally, one or more iterations of one of the already-mentioned matrix inversion algorithms: point and line relaxation, ADI, ILU, etc.). Once the pressure correction δp has been calculated, the pressure is updated as

$$p = p + \alpha_p \delta p,$$

where we have introduced a second relaxation parameter α_p for the pressure field, which may optionally be used to tune up the convergence rate.

2.4. *The MAC method*

The MAC and PISO methods were initially devised for the unsteady Navier–Stokes equations. Afterwards they have been applied to stationary problems using a false transient approach.

MAC's velocity prediction step is indistinguishable from that of SIMPLE, with the reciprocal of the false transient time step Δt taking the place of the relaxation parameter α in equations (4) and (5), but in the pressure correction stage a Poisson equation takes the place of equation (8). For reasons related to the unsteady origin of the algorithm, which we do not repeat here, the velocity correction is derived from a potential φ as

$$\delta \mathbf{v}' = -\nabla \varphi, \tag{9}$$

where $\delta \mathbf{v}'$ denotes the correction of velocity in vectorial form, with components $(\delta u', \delta v')$. The potential φ is directly proportional to the pressure correction δp:

$$\delta p_{i,j} = \frac{1}{\Delta t}\,\varphi = \alpha \varphi,$$

with the same α that appears as a relaxation parameter in equations (4) and (5).

In a discretized form relation (9) becomes

$$h\delta v'_{i,j+1/2} = -(\varphi_{i,j+1} - \varphi_{i,j}), \qquad h\delta u'_{i+1/2,j} = -(\varphi_{i+1,j} - \varphi_{i,j}).$$

Together with the continuity equation (3), it yields the following discrete Poisson equation:

$$\varphi_{i+1,j} + \varphi_{i-1,j} + \varphi_{i,j+1} + \varphi_{i,j-1} - 4\varphi_{i,j} = \frac{1}{\rho_{i,j}}\, t_{i,j}^{(p)}, \tag{10}$$

with $t_{i,j}^{(p)}$ denoting as before the residual of equation (3) after the predictor step.

2.5. *The PISO method*

Of the PISO method we have tested the incompressible form only. As its performance in connection with multigrid turned out inferior to that of the other methods, it has been deemed unnecessary to perform our tests in the quasi-incompressible case.

The PISO algorithm, after the common predictor step performed according to Section 2.2, provides two subsequent corrector steps for pressure and velocity. We shall denote the velocity and pressure corrections corresponding to each step by $\delta \mathbf{v}'$, $\delta p'$ and $\delta \mathbf{v}''$, $\delta p''$.

Introducing the notation

$$N_0(u_{i+1/2,j}) = a^{(u)}_{i+1/2,j}u_{i+1/2,j-1} + b^{(u)}_{i+1/2,j}u_{i-1/2,j} + d^{(u)}_{i+1/2,j}u_{i+3/2,j} + e^{(u)}_{i+1/2,j}u_{i+1/2,j+1},$$

$$N_0(v_{i,j+1/2}) = a^{(v)}_{i,j+1/2}v_{i,j-1/2} + b^{(v)}_{i,j+1/2}v_{i-1,j+1/2} + d^{(v)}_{i,j+1/2}v_{i+1,j+1/2} + e^{(v)}_{i,j+1/2}v_{i,j+3/2},$$

we can define $N_0(\mathbf{v})$ as the vector whose components are $N_0(u_{i+1/2,j})$ and $N_0(v_{i,j+1/2})$. As a consequence of grid staggering, the two components do not refer to the same spatial point but rather each to the point where the corresponding velocity component is evaluated.

Denoting for the sake of conciseness the discretized forms of the operators ∇ and div by the same symbol as their continuous counterparts, even though the two components are not evaluated at the same grid point, we may describe the PISO algorithm as follows.

Predictor step

$$(\alpha + c)\delta\mathbf{v} = -[c\mathbf{v} + N_0(\mathbf{v}) + \nabla p]. \tag{11}$$

This is, apart from notation, the same as the general predictor step of Section 2.2 when Jacobi point relaxation is chosen as the approximate matrix inversion algorithm. Notice that the symbol c stands for $c^{(u)}_{i+1/2,j}$ when applied to the u-component and for $c^{(v)}_{i,j+1/2}$ when applied to the v-component of velocity.

First corrector step

$$(\alpha + c)\delta\mathbf{v}' = -\nabla\delta p', \tag{12}$$

where $\delta p'$ is obtained from the solution of the difference equation

$$\text{div}\,[(\alpha + c)^{-1}\nabla\delta p'] = \text{div}\,(\mathbf{v} + \delta\mathbf{v}). \tag{13}$$

This is the same as the corrector step of SIMPLE. In the time-dependent form of PISO it is required that equation (13) be solved with good precision, but in the false transient form a compromise must be sought in order to optimize the time of computation.

Second corrector step

$$(\alpha + c)\delta\mathbf{v}'' = -N_0(\delta\mathbf{v}') - \nabla\delta p'', \tag{14}$$

where $\delta p''$ is obtained from the solution of the second elliptic equation

$$\text{div}\,[(\alpha + c)^{-1}\nabla\delta p''] = -\text{div}\,[(\alpha + c)^{-1}N_0(\delta\mathbf{v}')]. \tag{15}$$

Finally, once the second pressure correction $\delta p''$ has been determined, the second velocity correction is given by

$$\delta\mathbf{v}'' = -\{(\alpha + c)^{-1}[N_0(\delta\mathbf{v}') + \nabla\delta p'']\}. \tag{16}$$

3. MULTIGRID

As stated in Section 1, the purpose of this work is to test a multigrid modification of a few standard algorithms (and of a new one which we shall introduce below) in which the multigrid technique is applied in black-box style to the pressure correction phase only. In this way all that is needed is the multigrid solution of a single scalar elliptic equation, which is a well-studied problem and can be obtained with high efficiency.[1]

The multigrid method we use is quite standard but does contain some points that may deserve discussion, particularly in the restriction procedure and in the treatment of boundary conditions. We shall now describe this method with reference to the Poisson equation (as is relevant e.g. to MAC); the extension to a variable-diffusivity-type elliptic equation (as is encountered e.g. in connection with SIMPLE) is straightforward.

Let us consider the Poisson equation

$$\nabla^2 \varphi = b, \tag{17}$$

with the Neumann condition $\partial\varphi/\partial n = 0$ on part of the boundary of a rectangular domain and the Dirichlet condition $\varphi =$ constant on the rest.

The discretized form of equation (17) is

$$\varphi_{i+1,j} + \varphi_{i-1,j} + \varphi_{i,j+1} + \varphi_{i,j-1} - 4\varphi_{i,j} = b_{i,j}, \tag{18}$$

where $b_{i,j} = h^2 b$ and h is the discretization step.

In a certain grid sequence of increasing mesh size we shall indicate with the superscript 'f' (fine) the functions computed in turn on any given grid of the domain and with the superscript 'c' (coarse) the functions computed on the next coarser grid of the sequence.

A $V(1, 1)$ multigrid cycle[1] is adopted in which the coarsest grid encompasses two meshes along the shorter side of the rectangular domain. Each step in the multigrid cycle is composed of three operations: *smoothing* of the error, i.e. damping of the spectral components that on any given grid have a wavelength comparable with the grid size; a fine-to-coarse transfer operation, also called *restriction*; and a coarse-to-fine interpolation of correction, also called *prolongation*.

We shall now describe these three operations in detail.

3.1. *Smoothing and prolongation*

The smoothing algorithm we use is red–black point relaxation. This is very simple, its smoothing properties are as good as any other's for the Poisson equation or a not too unbalanced variable-diffusivity equation, and in addition it simplifies the operation of prolongation.

In fact, in the coarse-to-fine grid transfer one must correct the values of the current approximation on the fine grid with the ones obtained on the coarse grid by a certain operation of interpolation. Since the red–black relaxation algorithm is such that values of the potential at the 'red' points can be fully recovered from a knowledge of the 'black' only and vice versa, it is sufficient to correct the fine-grid solution (by a linear interpolation of the difference between the coarse-grid and fine-grid values) at the odd-numbered points only, because the following step of the smoother will determine the even-numbered ones completely.

The ordinary smoothing factor of red–black relaxation is of the order of 1:3 in a $V(1, 1)$ configuration, meaning that the error norm is reduced by a factor of approximately $\frac{1}{3}$ for each complete multigrid cycle, which is already quite good. However, we have found that a simple modification of the restriction operation, giving a bias to the starting values transferred to the coarse grid (as we shall describe in greater detail below), changes this factor from 1:3 to better than 1:10, at least with Dirichlet boundary conditions.

3.2. *Restriction*

Let us call h^{f} the mesh size of the fine grid and h^{c} the mesh size of the coarse grid (here $h^{\mathrm{f}} = h^{\mathrm{c}}/2$). The multigrid philosophy approximates equation (18) on the fine grid by a similar equation written on the coarse grid with a suitably modified right-hand side. The restriction operation consists of calculating the values of both the unknown φ and the right-hand side b on the coarse mesh so as to verify the two conditions that the coarse-grid equations be a discrete approximation of the same differential problem as the fine-grid equations are and that the coarse difference equations be exactly satisfied when the fine ones are. This can be done in more than one way.

The simplest approach, to copy the fine-grid values of φ on to the corresponding coarse-grid points and to put the RHS b^c equal to the sum of the LHS calculated on the current coarse-grid values and the fine-grid residual at the same point (multiplied by a factor depending on the different discretization steps), is usually discarded in favour of the use of weighted averages of the fine-grid residuals at a number of neighbouring points, because pointwise transfer from the fine to the coarse grid causes a possibly harmful decoupling between the coarse-grid solution and the residuals at the fine-grid points that have no correspondent on the coarse grid. However, when the smoother is so efficient as to produce an error reduction by a factor of $\frac{1}{3}$ in a $V(1, 1)$ cycle, weighted averaging of the residuals may draw a significant tribute in terms of computation time. Fortunately, red–black relaxation makes this averaging unnecessary: since at any given iteration step either the 'red' or the 'black' residuals are identically zero, considering just a single non-zero residual is equivalent to averaging five neighbours. Doing so yields an overall 1:3 error reduction per multigrid cycle for a Poisson equation with Dirichlet boundary conditions. Improvements are possible, however.

The improvement that we have been using for some time, but which we present for the first time here, stems from the observation that, even when the values of the unknown and the residuals are transferred pointwise from one to another grid to minimize the time of computation, the simultaneous availability of value and residual can be exploited for a better extrapolation of the unknown to the coarser grid. In particular, adding to the local value of the unknown the residual weighted by a suitable factor is equivalent to an additional relaxation step, which can be gained at the expense of a single multiplication and addition per coarse-grid point. By trial and error we have determined that the optimum weighting factor for the Poisson equation is two. The following restriction algorithm results.

For each coarse-grid point (i, j):

1. Compute the fine-grid residual $r^f = b^f_{2i,2j} - L\varphi^f_{2i,2j}$. ($L$ is the discretized Laplace operator, i.e. the LHS. of equation (18).)
2. Define the right-hand side on the coarse grid as $b^c_{i,j} = L^c\varphi^f_{2i,2j} + 2r^f$. (This is the standard way of proceeding, except that the coarse Laplace operator is applied to the fine-grid values at positions $2i + 2$, $2i - 2$, $2j + 2$ and $2j - 2$, because these have not been copied to the coarse grid.)
3. Perform the restriction of φ-values by the formula $\varphi^c_{i,j} = \varphi^f_{2i,2j} + 2r^f$ (rather than just copying as $\varphi^c_{i,j} = \varphi^f_{2i,2j}$).

This little modification, which corrects the initial coarse-grid values of the potential function by the fine-grid residuals multiplied by two, changes the error reduction per cycle of the $V(1, 1)$ multigrid algorithm with red–black smoothing from 1:3 to better than 1:10, at least when Dirichlet boundary conditions are imposed.

3.3. Boundary conditions

Dirichlet boundary conditions are straightforward. For the pressure correction equation, however, be it of the Poisson or the variable-diffusivity type, we also need to impose Neumann boundary conditions on the parts of the boundary where the normal velocity is given. The implementation of Neumann boundary conditions strongly interacts with the multigrid algorithm and, if overlooked, this interaction is apt to slow down the speed of convergence by orders of magnitude. Once again restriction is the critical step.

Let us consider the scheme of Figure 1, which represents the grid points involved in the computation of the Neumann boundary condition. Dots and crosses distiguish the 'red' and 'black' points respectively; we recall that at any particular iteration step either the 'red' or the 'black' residuals are identically zero. On the finest grid the boundary passes just in the middle between two adjacent rows of grid points, so that the Neumann boundary condition may be written with second-order accuracy as

$$\varphi_{1,j} - \varphi_{0,j} = b_{0,j}, \tag{19}$$

where $b_{0,j} = hb$.

Since the Neumann boundary condition involves the derivative of the unknown function, it must be treated just as an additional differential equation and its restriction to coarser and coarser grids involves the calculation of a new right-hand-side $b^{c}_{0,j}$ under the same general rules that apply to internal points. Namely, $b^{c}_{0,j}$ must equal the sum of the LHS of equation (19) calculated on the coarser grid and the current residual of the same equation on the finer grid multiplied by a suitable step change factor. However, even though pointwise restriction turns

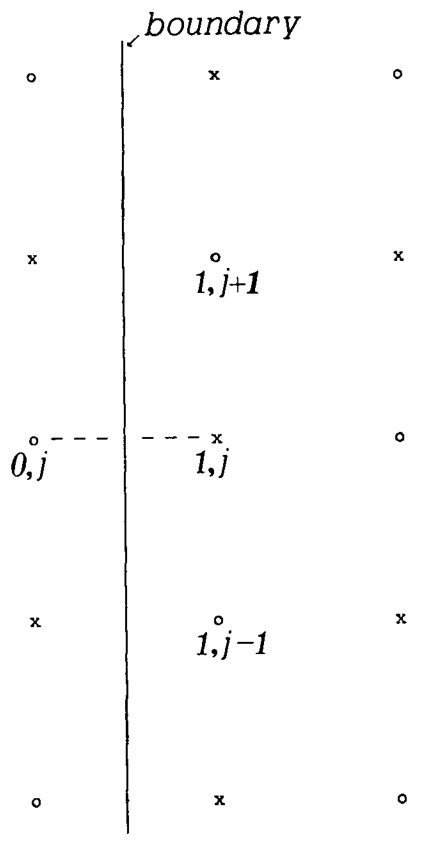

Figure 1. Position of the grid points involved in the computation of the Neumann boundary condition

out satisfactory for the internal points, doing so for the Neumann boundary conditions substantially deteriorates the rate of convergence. The generally advised remedy,[1] to use a weighted average of neighbouring residuals, works for boundary points as well. However, account must be taken of the fact that in red–black relaxation residuals of one colour are identically zero when those of the other are not; therefore, with reference to Figure 1, the closest neighbours whose residuals can be non-trivially combined with that of point $(0, j)$ are points $(1, j+1)$ and $(1, j-1)$. (Considering the residuals of points in column 1 is also suggested by the fact that the new fake boundary corresponding to the coarser grid passes through these points.) On choosing the weights by trial and error, we have arrived at the following boundary restriction algorithm.

For every boundary point where Neumann conditions are applied:

1. Compute the fine-grid residuals $r^{\mathrm{f}}_{0,j}$ (from equation (19)), $r^{\mathrm{f}}_{1,j+1}$ and $r^{\mathrm{f}}_{1,j-1}$ (from equation (18)).
2. Define $e_{0,j} = 0{\cdot}5 r^{\mathrm{f}}_{0,j} + 0{\cdot}25(r^{\mathrm{f}}_{1,j+1} + r^{\mathrm{f}}_{1,j-1})$.
3. Define the right-hand side for the boundary condition on the coarse grid as $b^{\mathrm{c}}_{0,j/2} = \varphi^{\mathrm{f}}_{2,j} - \varphi^{\mathrm{f}}_{0,j} + 4e_{0,j}$ and the extrapolated coarse-grid value as $\varphi^{\mathrm{c}}_{0,j/2} = \varphi^{\mathrm{f}}_{0,j} + 4e_{0,j}$.

Attention must also be paid to the corners of the rectangular computational box if Neumann conditions are imposed along both sides that meet at the corner.

The algorithm achieves a 1:5 error reduction ratio per $V(1,1)$ multigrid cycle with mixed Neumann and Dirichlet boundary conditions. Although not as good as with Dirichlet conditions alone, this performance is quite satisfactory for all applications. The extension from the Poisson to the variable-diffusivity equation is straightforward and achieves a similar convergence rate.

3.4. *Application of the multigrid technique to the pressure correction stage of Navier–Stokes computations*

The multigrid technique just described has been used in this work for the iterative solution of the pressure correction equation.

The SIMPLE and PISO methods are characterized by elliptic equations with non-constant coefficients, while the MAC method and the new method DOT, which we shall introduce below, are characterized by Poisson equations. In each of these methods the standard pressure correction step (effected by one or more iterations of either point or line relaxation) has been replaced by one $V(1, 1)$ multigrid cycle on the relevant equation (the pressure correction equation (8) of the SIMPLE method, the two elliptic equations (13) and (15) of the first and second corrector steps of the PISO method, the Poisson equations (10) of the MAC method and (26) of the DOT method), without changing any other detail of the algorithm or its programming. The results obtained with the various algorithms will be described and compared in Section 5.

4. DGS AND THE NEW METHOD DOT

The DGS (distributive Gauss–Seidel) method was introduced by Brandt and Dinar[10] for the simultaneous solution of the momentum and continuity equations by multigrid. Not surprisingly given its origin, we have found this method unsuitable for a mixed setting in which multigrid is applied to the pressure correction step only; nevertheless, the philosophy of DGS suggested to us a modification that works, which we have called DOT (difference operator triangularization).

At the basis of DGS there is the approximate triangularization of an operatorial form of the steady Navier–Stokes differential equations. If we introduce the advection–diffusion operator Q as

$$Q = \frac{1}{Re}\nabla^2 - u\frac{\partial}{\partial x} - v\frac{\partial}{\partial y}, \tag{20}$$

the incompressible Navier–Stokes equations can be written in matrix form as

$$\begin{bmatrix} -Q & 0 & \partial/\partial x \\ 0 & -Q & \partial/\partial y \\ \partial/\partial x & \partial/\partial y & 0 \end{bmatrix}\begin{bmatrix} u \\ v \\ p \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

and their linearized δ-form as

$$\begin{bmatrix} -Q & 0 & \partial/\partial x \\ 0 & -Q & \partial/\partial y \\ \partial/\partial x & \partial/\partial y & 0 \end{bmatrix}\begin{bmatrix} \delta u \\ \delta v \\ \delta p \end{bmatrix} = \begin{bmatrix} -t^{(u)} \\ -t^{(v)} \\ -t^{(c)} \end{bmatrix} \tag{21}$$

If we now work as if the operators Q and $\partial/\partial x$, $\partial/\partial y$ commuted with each other (this is where an approximation is made, because Q involves non-constant coefficients), we can triangularize the above matrix by, for instance, Gauss elimination. The first and second rows of equation (21) thus give

$$\delta u = Q^{-1}(t^{(u)} + \partial\delta p/\partial x), \qquad \delta v = Q^{-1}(t^{(v)} + \partial\delta p/\partial y),$$

which, when substituted into the third, yield

$$\frac{\partial}{\partial x}\left(Q^{-1}\frac{\partial\delta p}{\partial x}\right) + \frac{\partial}{\partial y}\left(Q^{-1}\frac{\partial\delta p}{\partial y}\right) = -t^{(c)} - \frac{\partial}{\partial x}Q^{-1}t^{(u)} - \frac{\partial}{\partial y}Q^{-1}t^{(v)}. \tag{22}$$

If we let operators commute, we may now introduce a potential $\varphi = -Q^{-1}\delta p$ and equation (22) thus becomes

$$\nabla^2\varphi = t'^{(c)}, \tag{23}$$

where $t'^{(c)}$ represents the residual of the continuity equation after the predicted values of velocity, $u + Q^{-1}t^{(u)}$ and $v + Q^{-1}t^{(v)}$, have been inserted. Once the potential has been calculated from the solution of the Poisson equation (23), the corrected values of the velocity components become $u = u - \varphi_x$ and $v = v - \varphi_y$ and the pressure correction becomes

$$\delta p = -Q\varphi. \tag{24}$$

As may be seen, Brandt and Dinar's DGS, although derived from a quite different line of reasoning, is not in practice very different from the other methods we have presented, except in the pressure correction phase where formula (24) is adopted. The critical step is, however, the inversion of the diffusion–advection operator Q in the velocity prediction step, which must be fairly accurate for equation (24) to be consistent. DGS does work when multigrid is used for this purpose, but turned out unsatisfactory in our tests where we wanted to apply a standard method, say line relaxation, to the momentum equation (i.e. to the inversion of Q) and multigrid to the Poisson equation (23) only.

The failure of DGS to provide satisfactory convergence in a mixed setting where multigrid is applied to the pressure correction alone may be ascribed to inconsistency between the loose approximation of the differential operator Q^{-1} provided by a single iteration of point or line relaxation and the full operator Q used in equation (24). We may notice, however, that any such relaxation algorithm involves the exact inversion of an approximate difference operator obtained by deleting from the discretization of Q one or more of the five points in its difference stencil (four in Jacobi point relaxation, two in Gauss–Seidel point relaxation or Jacobi line relaxation, one in Gauss–Seidel line relaxation).

If we call Q_d the difference operator corresponding to any particular approximate inversion method that is chosen for the momentum equation, we may rewrite equation (21) as

$$\begin{bmatrix} -Q_\mathrm{d} & 0 & \partial/\partial x \\ 0 & -Q_\mathrm{d} & \partial/\partial y \\ \partial/\partial x & \partial/\partial y & 0 \end{bmatrix} \begin{bmatrix} \delta u \\ \delta v \\ \delta p \end{bmatrix} = \begin{bmatrix} -t^{(u)} \\ -t^{(v)} \\ -t^{(c)} \end{bmatrix} \tag{25}$$

and, correctly this time, assume that we are able to calculate Q_d^{-1} quickly and exactly.

All the previous line of reasoning applies unchanged, but the pressure correction formula (24) is now replaced by

$$\delta p = -Q_\mathrm{d}\varphi, \tag{26}$$

where the approximate difference operator Q_d takes the place of the differential operator Q. The result is what we call DOT (difference operator triangularization). Notice that, as we set up as our main requirement, the already existing velocity prediction algorithm need not be modified, all changes being limited to the pressure correction stage.

The main difference between our proposed method DOT and Brandt and Dinar's DGS is that they, not distinguishing between Q_d and Q, obtain a method that is only suitable in conjunction with a multigrid algorithm applied to both the momentum and pressure correction equations. In contrast, in our method the multigrid pressure correction can be coupled to any one of several operators usable in the iterative process for the momentum equations.

5. RESULTS

This section shows the results regarding time and number of iterations needed to reach a prefixed level of convergence for the SIMPLE method with and without application of multigrid to the pressure correction equation and for the MAC, PISO and DOT methods with application of multigrid to the pressure correction equation.

The test problem is a very idealized combustion chamber: a rectangular cell of 3:1 aspect ratio with a certain number of inflows and an outflow. While on the solid walls of the chamber both components of velocity are zero, there are several possibilities for the boundary conditions to be given at the inflow and outflow sections.

In general we are allowed two conditions for the three unknowns u, v and p at the inlet and outlet. At the inlet we have given both components of velocity, specifically zero tangential component and constant normal component; at the outlet we have given a constant value of the pressure and zero tangential velocity. We might alternatively have given the pressure at the inlet and the normal velocity at the outlet or the two pressures with little effort. At any rate, giving a velocity condition allows us to approximately fix the flow rate, and the pressure at the outlet (which occupies the whole of a shorter side of the rectangle) is quite close to the pressure that prevails throughout most of the cell.

The same discretization of the test problem is used for all methods, so that the final numerical results at convergence are always exactly the same: a 48×16 uniform staggered grid with the boundary of the rectangle passing between two adjacent rows of pressure points.

Two geometries have been chosen for the tests. In both the outflow is located on a shorter side of the rectangle, while two different possibilities are considered for the inflow: (i) a single inlet occupying the first six grid points of one longer side; (ii) two inlets located on opposite longer sides of the domain, the first involving the grid points from the fifth to the ninth and the second from the 20th to the 23rd, with numeration starting from the side opposite the outlet.

With all methods the iteration was stopped and the computation time recorded when the sum of the absolute values of the continuity and momentum equation residuals (which were calculated in the same way in all methods) had been reduced by four orders of magnitude with respect to its starting value.

Once the geometry and the equation of state (that of a perfect gas with $\gamma = 1{\cdot}4$) have been defined, the characteristics of the flow are completely determined by two dimensionless numbers, which may be defined as a Reynolds number and a Mach number in terms of suitable velocity, pressure and enthalpy boundary conditions.

Let H_0 denote the given constant total enthalpy of the flow and p_{out} the given pressure at the outlet. A reference density can be defined as the stagnation density that would be found at enthalpy H_0 and pressure p_{out}, i.e. by the formula

$$\rho_r = \gamma p_{\text{out}}/(\gamma - 1)H_0. \tag{27}$$

As a reference length L_r we take the width of the rectangle, while our reference velocity v_r is the given inlet velocity multiplied by the ratio between inlet and outlet width (i.e. the velocity that would prevail at the outlet if the flow were uniform and incompressible).

In terms of the above quantities the Reynolds and Mach numbers we have used to parametrize the test results are

$$Re = \rho_r v_r L_r/\mu, \tag{28}$$

$$M^2 = \rho_r v_r^2/\gamma p_{\text{out}}. \tag{29}$$

The ranges considered are $0 < M < 0{\cdot}3$ (the limit for the application of some of the methods described) and $0 < Re < 100$ (the point where the recirculation cell that forms inside the domain extends all the way to the outlet, making the zero-tangent-velocity outlet condition unrealistic). Concerning these limits, it is to be observed that $M = 0{\cdot}3$ (when the Mach number is defined by equation (29)) is about twice the maximum Mach number that is likely to be encountered in a real combustion chamber; $Re = 100$, despite looking desperately low with respect to the physical Reynolds number at which a combustion chamber operates, is quite realistic as a value of the *numerical* Reynolds number constructed with the eddy viscosity of any turbulence model that the actual modelling of a combustion chamber might use.

Figures 2–5 show the streamlines and the pressure field for the two geometrical configurations considered, with Reynolds number 100 and Mach number 0·2. Notice that, given the particular non-dimensionalization we have chosen for the pressure, the pressure values in Figures 3–5 may also be read (apart from a factor of γ) as the inverse square of the local Mach number in the chamber.

5.1. *The SIMPLE method*

Of SIMPLE we have tested five versions, differing in the iterative methods applied to the momentum and pressure correction equations and in the way a relaxation parameter is introduced into the momentum equations.

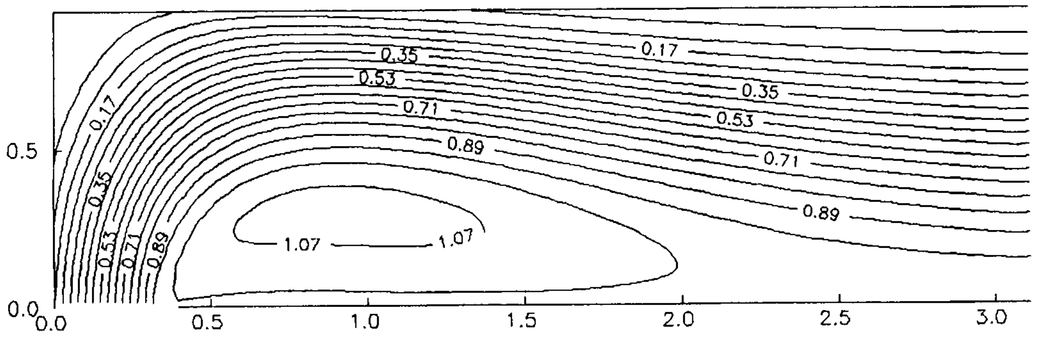

Figure 2. Streamlines for the flow in a single-inlet geometry at a Reynolds number of 100 and a Mach number of 0·2

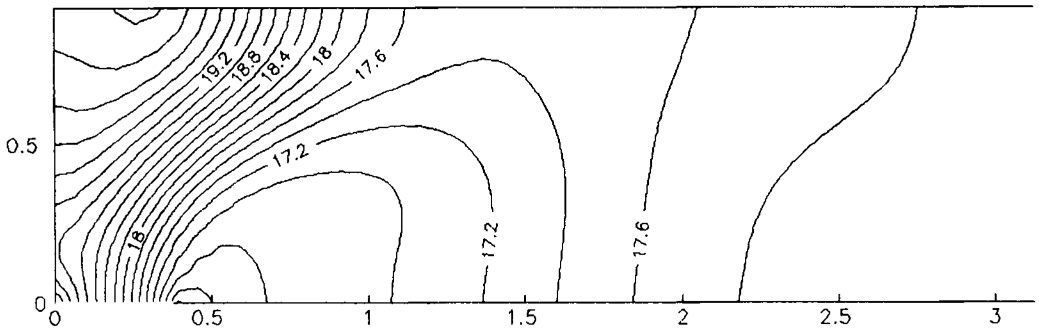

Figure 3. Contour map of non-dimensional pressure for the flow in a single-inlet geometry at a Reynolds number of 100 and a Mach number of 0·2

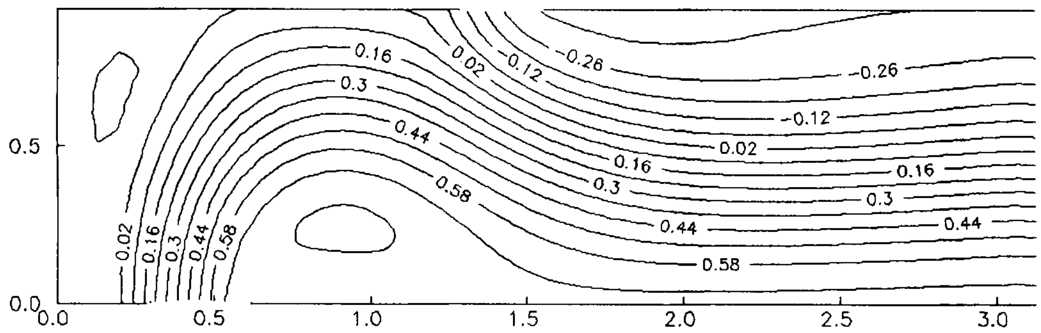

Figure 4. Streamlines for the flow in a double-inlet geometry at a Reynolds number of 100 and a Mach number of 0·2

A. Line relaxation is applied to the momentum equation and point Gauss–Seidel relaxation to the pressure correction equation. The relaxation parameter α as shown in equations (4) and (5) is chosen so as to minimize the computation time.

B. Same as A, but for a different choice of the relaxation parameter. The α that appears in equations (4) and (5) is set to zero and instead the increments δu and δv are only partially summed to the current velocity, as $u = u + \alpha\delta u$ and $v = v + \alpha\delta v$. The value of α is again optimized by trial and error.

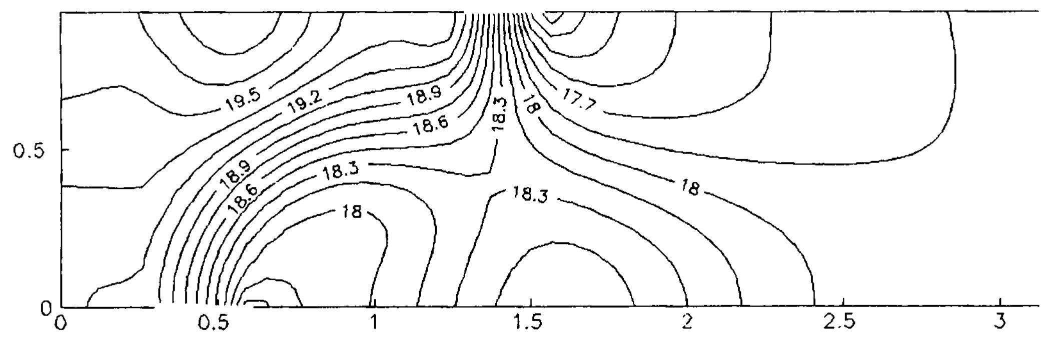

Figure 5. Contour map of non-dimensional pressure for the flow in a double-inlet geometry at a Reynolds number of 100 and a Mach number of 0·2

C. Line relaxation is used for both the momentum and pressure correction equations, with the relaxation parameter as in A.

D. Line relaxation is used for the momentum equations and multigrid (as described in Section 3) for the pressure correction equation.

E. Point Gauss–Seidel is used for the momentum equations and multigrid for the pressure correction equation.

Among the first three versions, all of which do not use multigrid, A turned out the fastest, then C and finally B. It is therefore more convenient to use the form of relaxation parameter indicated in equations (4) and (5) rather than a partialization of the velocity increments, and subordinately to use point rather than line relaxation for the pressure correction equation (line relaxation *is* preferable for the momentum equations when multigrid is not used). The time to convergence for A is about 15% lower than for C and 30% lower than for B.

Version E, which uses multigrid, is the fastest, as will appear from the detailed results given below together with those of other methods.

Version D, in which the multigrid pressure correction technique is coupled to a line relaxation method for the momentum equations, does not converge for any values of the pressure and velocity relaxation parameters.

5.2. A comparison among SIMPLE, MAC, PISO and DOT in incompressible flow

Tables I and II report the time and number of iterations to convergence (defined as the point where the sum of the absolute values of the residuals of all equations has decreased by a factor of 10^{-4} with respect to its initial value) for the best single-grid SIMPLE and the four methods SIMPLE, MAC, PISO and DOT with multigrid pressure correction. We recall that all these

Table I. Time to convergence for the case of incompressible flow in a single-inlet geometry

Reynolds number	SIMPLE	SIMPLE + MG		MAC + MG		PISO + MG		DOT + MG	
	Time	Time	Ratio	Time	Ratio	Time	Ratio	Time	Ratio
30	4'53"	1'24"	3·5	54"	5·4	1'15"	3·9	38"	7·7
100	7'21"	1'15"	5·9	1'10"	6·3	1'49"	4·0	51"	8·6

Table II. Number of iterations to convergence for the case of incompressible flow in a single-inlet geometry

Reynolds number	SIMPLE	SIMPLE + MG	MAC + MG	PISO + MG	DOT + MG
30	317	110	45	28	28
100	477	98	57	41	38

methods share the same velocity prediction step, so that the only actual difference is in how the pressure correction is applied.

The relaxation parameters α and α_p (where applicable) have been separately optimized and the best result for each method is reported. The optimum relaxation parameters at a Reynolds number of 30 turned out to be $\alpha = 0$ and $\alpha_p = 1$ (no over- or underrelaxation at all) for SIMPLE, $\alpha = 0{\cdot}15$ and $\alpha_p = 0{\cdot}80$ for the SIMPLE + (MG), $\alpha = 1{\cdot}00$ for MAC + MG, $\alpha = 0{\cdot}29$ for PISO + MG and $\alpha = 0{\cdot}35$ for DOT + MG. At a Reynolds number of 100 the optimum values of the relaxation parameters were different: $\alpha = 0{\cdot}60$ and $\alpha_p = 0{\cdot}80$ for SIMPLE, $\alpha = 0{\cdot}05$ and $\alpha_p = 0{\cdot}80$ for SIMPLE + MG, $\alpha = 0{\cdot}50$ for MAC + MG, $\alpha = 0{\cdot}45$ for PISO + MG and $\alpha = 0{\cdot}25$ for DOT + MG.

From Table III we can observe that SIMPLE + MG and PISO + MG are almost equivalent, MAC is slightly faster and DOT is the fastest to converge.

5.3. A comparison among SIMPLE, MAC and DOT in compressible flow

The times and numbers of iterations to convergence obtained in quasi-incompressible compressible flow (i.e. when compressibility cannot be neglected but the Mach number is small) are reported in Tables III–VIII. The Mach number range we consider is 0·01 (which is practically indistinguishable from incompressible flow) to 0·3, which is the value where more or less all the algorithms tested begin to fail to converge. Mach 0·3 is, at any rate, far beyond the maximum practically interesting value for combustion chambers.

The optimum values of the relaxation parameters in these tests turned out to increase with Mach number for all methods.

Tables III, V and IV, VI show the relative times and numbers of iterations to convergence respectively for the methods SIMPLE, SIMPLE + MG, MAC + MG and DOT + MG in the case of the single-inflow geometry. We have not tested PISO + MG in compressible flow because

Table III. Time to convergence for the case of compressible flow at Reynolds number 30 in a single-inlet geometry

Mach number	SIMPLE	SIMPLE + MG		MAC + MG		DOT + MG	
	Time	Time	Ratio	Time	Ratio	Time	Ratio
0·01	5'00"	1'30"	3·3	1'00"	5·0	44"	6·8
0·1	4'13"	1'29"	3·2	1'00"	4·2	44"	5·7
0·2	3'37"	1'27"	2·5	1'09"	3·1	42"	5·2
0·3	3'38"	1'51"	2·0	1'37"	2·2	47"	4·6

Table IV. Number of iterations to convergence for the case of compressible flow at Reynolds number 30 in a single-inlet geometry

Mach number	SIMPLE	SIMPLE + MG	MAC + MG	DOT + MG
0·01	317	110	45	28
0·1	267	109	45	28
0·2	229	106	52	27
0·3	231	135	75	30

Table V. Time to convergence for the case of compressible flow at Reynolds number 100 in a single-inlet geometry

Mach number	SIMPLE	SIMPLE + MG		MAC + MG		DOT + MG	
	Time	Time	Ratio	Time	Ratio	Time	Ratio
0·01	7'34"	1'19"	5·7	1'16"	6·0	1'00"	7·6
0·1	6'38"	1'18"	5·1	1'16"	5·2	1'00"	6·6
0·2	5'39"	1'14"	4·6	1'14"	4·6	55"	6·2
0·3	4'34"	1'46"	2·6	1'26"	3·1	Divergent	

Table VI. Number of iterations to convergence for the case of compressible flow at Reynolds number 100 in a single-inlet geometry

Mach number	SIMPLE	SIMPLE + MG	MAC + MG	DOT + MG
0·01	480	97	57	38
0·1	421	96	56	38
0·2	358	90	56	35
0·3	290	129	65	Divergent

Table VII. Time to convergence for the case of compressible flow at Reynolds number 100 in a double-inlet geometry

Mach number	SIMPLE	SIMPLE + MG		MAC + MG		DOT + MG	
	Time	Time	Ratio	Time	Ratio	Time	Ratio
0·2	4'39"	58"	4·8	49"	5·7	43"	6·5
0·3	4'32"	58"	4·7	48"	5·7	1'05"	4·2

Table VIII. Number of iterations to convergence for the case of compressible flow at Reynolds number 100 in a double-inlet geometry

Mach number	SIMPLE	SIMPLE + MG	MAC + MG	DOT + MG
0·2	295	71	37	27
0·3	288	71	37	40

it turned out less competitive than DOT and MAC in the previous test, despite being considerably more complicated then either.

The analysis of Tables III and V shows that the performance of all the methods gets worse with increasing Reynolds number. The reason is probably that the recirculation cell present inside the chamber grows bigger and bigger and eventually approaches the outflow section where incompatible boundary conditions are imposed. With increasing Mach number for a fixed Reynolds number all the methods initially get faster and then lose their efficiency.

Tables VII and VIII show the relative times and numbers of iterations to convergence for the double-inflow geometry at a Reynolds number of 100 and Mach numbers of 0·2 and 0·3. This is a fairly tough test, with the values of the Reynolds and Mach numbers that in the single-inlet geometry appeared to provide the most severe conditions and a position of the inlet sections that, in order to have a fluid flow resembling as closely as possible the one occurring in a real combustion chamber, has been chosen so as to create a complex flow with three recirculation cells. As may be seen from Table VII, all three methods fitted with a multigrid pressure correction surpassed standard SIMPLE in this test by roughly a factor of five to six. DOT turned out best, albeit by a small margin, at a Mach number less than or equal to 0·2, but was more sensitive than the others to a further increase in Mach number.

6. CONCLUSIONS

The tests described in this paper appear to clearly indicate the potential of improving already existing industrial simulation programmes for quasi-incompressible flows, e.g. in combustion chambers, by retrofitting a multigrid pressure correction stage. The usually adopted velocity prediction with frozen pressure, which is common to the standard methods SIMPLE and MAC, to the newer PISO and to the DOT method proposed in this paper, can be coupled with a pressure correction that adopts multigrid in its interior in a hidden fashion, without requiring modifications to the remaining code. Such a strategy is suggested by the pressure correction being at the same time the most time-consuming phase of the procedure and that to which multigrid can be applied to the greatest advantage.

Within the context of our tests of several otherwise standard methods, two new developments have been proposed: a simple modification of the restriction operation of the multigrid cycle which, at an absolutely negligible computational cost, changes the error reduction factor from 1:3 to better than 1:10 per cycle (with Dirichlet boundary conditions; from 1:2 to 1:5 with Neumann or mixed boundary conditions); and the method DOT, which, even though not by a large margin, achieved the smallest overall computation times among the ones tested.

The time reduction obtainable by multigrid pressure correction cannot be stated, of course, without reference to a specific problem, and the results given in the previous section are quite diversified; nevertheless, it appears fair to say that in all cases where pressure correction absorbs a significant fraction of the total time of computation the gain is definitely worthwhile. In the double-inlet geometry we tested with a 16×48 staggered computational grid, a factor of about five was gained with all multigrid-fitted methods.

ACKNOWLEDGEMENT

The authors are indebted to Alfa Romeo Avio (Pomigliano d'Arco, NA, Italy) for providing a fellowship to one of them (A.D'A.), and in particular to P. Di Martino and A. Terlizzi for bringing this problem to their attention and for several useful discussions.

REFERENCES

1. A. Brandt, 'Multigrid techniques: 1984 guide with application to fluid dynamics', *GMD Studien Nr. 85*, Gesellschaft für Mathematik und Datenverarbeitung MBH, Bonn, 1984.
2. P. Di Martino, E. Narciso and G. Cinque, 'Numerical model for prediction of reverse flow combustor aerothermal characteristic', *Proc. IUTAM Symp. on Aerothermodynamics in Combustors*, Taipei, June 1991, Springer, Berlin, 1991, pp. 339–353.
3. P. Di Martino and A. Terlizzi, 'A fully Navier–Stokes solver for 3-D incompressible laminar flow in complex geometries', in A. Sousa, C. A. Brebbia and G. M. Carlomagno (eds), *Computational Methods and Experimental Measurements V*, Elsevier, London, 1991, pp. 3–13.
4. P. Di Martino and G. Cinque, 'Flow computation in gas turbine combustors with complex geometries', *Proc. Joint Meet. French, Italian and Swedish Sections of the Combustion Institute*, Capri, September 1992, Napoli, 1992, pp. III.6–III.8.
5. S. V. Patankar and D. B. Spalding, 'A calculation procedure for heat, mass and momentum transfer in three-dimensional parabolic flows', *Int. J. Heat Mass Transfer*, **15**, 1787 (1972).
6. R. I. Issa, 'Solution of the implicitly discretized fluid flow equations by operator-splitting', *J. Comput. Phys.*, **62**, 66–82 (1985).
7. F. H. Harlow and J. E. Welch, 'Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface', *Phys. Fluids*, **8**, 2182–2189 (1965).
8. T. F. Miller and F. W. Schmidt, 'Evaluation of a multilevel technique applied to the Poisson and Navier–Stokes equations', *Numer. Heat Transfer*, **13**, 1–26 (1988).
9. W. Shyy, M.-H. Chen and C.-S. Sun, 'Pressure-based multigrid algorithm for flow at all speeds', *AIAA J.*, **30**, 2660–2669 (1992).
10. A. Brandt and N. Dinar, 'Multigrid solution to elliptic flow problems', in S. Parter (ed.), *Numerical Methods for Partial Differential Equations*, Academic, New York, 1979, pp. 53–147.
11. B. P. Leonard, 'A stable and accurate convective modelling procedure based on quadratic upstream interpolation', *Comput. Methods Appl. Mech. Eng.*, **19**, 59–98 (1979).