Int. J. Heat Mass Transfer. Vol. 35, No. 10, pp. 2705–2716, 1992
Printed in Great Britain

0017–9310/92 $5.00 + 0.00

Artificial pressure for pressure-linked equation

S. L. LEE and R. Y. TZONG
Department of Power Mechanical Engineering, National Tsing-Hua University, Hsinchu, Taiwan 30043, Republic of China

(*Received* 7 *May* 1991 *and in final form* 27 *September* 1991)

Abstract—The discretized pressure-linked equation can be proven to possess a singular coefficient matrix. This implies that the pressure solution corresponding to a given velocity would not exist unless this velocity precisely satisfies the continuity equation. In the present investigation, an artificial source term is added to the pressure-linked equation. This additional term will generate an extra pressure called the 'artificial pressure' for each updated velocity to compensate for their nonzero dilation. The use of the artificial pressure is equivalent to creating an extra mass for the need of the updated velocity in satisfying the continuity equation. This treatment guarantees the existence and uniqueness of the pressure solution such that the pressure can be directly solved without recourse to the conventional pressure correction equation. Based on the concept of artificial pressure, the APPLE (Artificial Pressure for Pressure-Linked Equation) and the NAPPLE (Nonstaggered APPLE) algorithms are developed for incompressible flows. Through two well-known examples, the APPLE algorithm is found to produce essentially the same results with only about 40% CPU time as compared with the SIMPLER algorithm. Due to its simplicity, the NAPPLE algorithm has a potential use for problems with arbitrarily shaped domain as long as the grid size is not large.

INTRODUCTION

THE MAJOR difficulty in calculating an incompressible flow seems to lie in the unknown pressure gradient. Mathematically speaking, the pressure solution should be defined such that the continuity equation is satisfied. Unfortunately, there is no obvious equation for the pressure field. Among the early efforts on solving this problem, Patankar and Spalding [1] proposed a method known as the SIMPLE (Semi-Implicit Method for Pressure-Linked Equations) algorithm. The SIMPLE algorithm converts the continuity equation successfully into a direct algorithm for the pressure solution. However, it needs a heavy underrelaxation in general and thus poses a rather slow convergence rate. To enhance the efficiency of the SIMPLE algorithm, a few SIMPLE-like algorithms such as SIMPLER [2, 3], SIMPLEC [4], SIMPLEST [5], PISO [6] and FIMOSE [7] have been developed.

It is noted that the SIMPLE algorithm and its variants [2–7] all need a pressure correction equation. In addition, all of them should be performed on staggered grid systems [8]. Although the use of a staggered grid could avoid producing an uncontrollable checkerboard pressure, it introduces some inconveniences when the algorithms are applied on a physical domain having an arbitrary shape. To remedy these difficulties, Rhie and Chow [9] proposed the pressure-weighted interpolation method (PWIM) such that the SIMPLE algorithm could be applied on a nonstaggered grid system. A subsequent study by Peric [10] confirmed this observation. Unfortunately, the solution provided by PWIM [9, 10] depends on the value of the underrelaxation factor (or the size of the virtual time step) as pointed out by Patankar [11]. Although this subtle drawback can be improved by employing some particular interpolation techniques [12, 13], the PWIM could produce physically impossible convective velocity especially in a region where the pressure gradient has a rapid variation. This point has been well discussed by Miller and Schmidt [13].

In the present investigation, the characteristics of the pressure-linked equation are examined in detail. An artificial source term then is added to the pressure-linked equation to generate an extra pressure to compensate for the nonzero dilation of an updated velocity. For convenience, the extra pressure will be referred to as the 'artificial pressure' in the present investigation. The use of the artificial pressure can be proven to provide a sufficient condition for the existence of the pressure solution such that the pressure can be directly solved without recourse to a pressure correction equation. This new numerical technique is called the APPLE algorithm. With a minor revision, the NAPPLE algorithm is proposed for nonstaggered grid systems. The performances of both algorithms will be compared with that of the SIMPLER algorithm [2, 3] through two well-known examples.

CHARACTERISTICS OF THE PRESSURE-LINKED EQUATION

For simplicity, the characteristics of the pressure-linked equation will be demonstrated through a two-dimensional flow on Cartesian coordinates with a uniform grid system. As in ref. [14], the dimensionless conservation equations for a two-dimensional incompressible flow can be written as

NOMENCLATURE

$[A]$, $\{B\}$ matrix and vector defined in equation (10)
a, b weighting factors in the momentum equation (4) and in the pressure equation (24)
a_{ij}, b_i elements of matrix $[A]$ and vector $\{B\}$
a_R right-hand side of equation (4)
e proportional constant of the artificial source
$\{F\}$ a vector defined by $\{F\} = \{B\} + [G]\{\tilde{p}\}$
f_N Nth element of vector $\{F\}$
$[G]$ a matrix having only two nonzero diagonals, $[L][U] - [A]$
$[L]$ a lower triangle matrix having only four nonzero diagonals
N total number of grid points
n number of grid points in x- or y-coordinates
p dimensionless pressure
$\tilde{p}$ guessed pressure
p_N pressure at the Nth point
Pr Prandtl number
Ra Rayleigh number
Re Reynolds number
t dimensionless time
$[U]$ an upper triangle matrix with only four nonzero diagonals
u, v dimensionless velocities in x- and y-direction, respectively
$\hat{u}$, $\hat{v}$ pseudo-velocities defined by equation (5b)
x, y system of coordinates.

Greek symbols
γ pseudo-conductivity defined by equation (5c)
Δ difference quantity
ε dilation of the fluid
$\hat{\varepsilon}$ pseudo-dilation of the fluid defined by equation (9b) or (23b)
θ dimensionless temperature
λ time step parameter, $\Delta t/(\Delta x)^2$
σ source term of pressure Poisson equation (29)
ψ stream function
ω vorticity.

Superscript
$*$ matrix elements after a complete forward Gaussian elimination procedure.

Subscripts
E, S, W, P quantity at points E, S, W and P defined in Figs. 1 and 2
e, n, s, w quantity at points e, n, s and w defined in Figs. 1 and 2
N quantities at point N defined in Figs. 1 and 2 or quantities at the Nth grid point
x, y, t differentiations with respect to x, y and t, respectively
0 quantities at the time $t - \Delta t$.

$$u_x + v_y = 0 \tag{1}$$

$$u_t + uu_x + vu_y = -p_x + \frac{1}{Re}(u_{xx} + u_{yy}) \tag{2}$$

$$v_t + uv_x + vv_y = -p_y + \frac{1}{Re}(v_{xx} + v_{yy}) \tag{3}$$

where $u_x = \partial u/\partial x$, $u_y = \partial u/\partial y$, $u_t = \partial u/\partial t$, etc. Upon discretizing the x-momentum equation (2) at point P by using an implicit numerical scheme such as the power-law scheme [3] and the weighting function scheme [15], one obtains

$$a_W u_W + a_E u_E + a_S u_S + a_N u_N + a_P u_P - \left(\frac{\partial p}{\partial x}\right)_P = a_R \tag{4}$$

where the subscripts W, E, S, N denote quantities at the nearest neighbor grid points of point P lying, respectively, to the west, east, south and north of point P. The definition of the weighting factors a_W, a_E, a_S, a_N and a_P can be found elsewhere [15]. The factor a_R is the source term excluding the pressure gradient. Such a notation system has been well accepted [1–7, 9–13, 15]. For convenience, equation (4) is rewritten as

$$u_P = \hat{u}_P - \gamma_P(\partial p/\partial x)_P \tag{5a}$$

$$\hat{u}_P = \gamma_P(a_W u_W + a_E u_E + a_S u_S + a_N u_N - a_R) \tag{5b}$$

$$\gamma_P = -(1/a_P). \tag{5c}$$

Figure 1 shows a staggered grid system for an

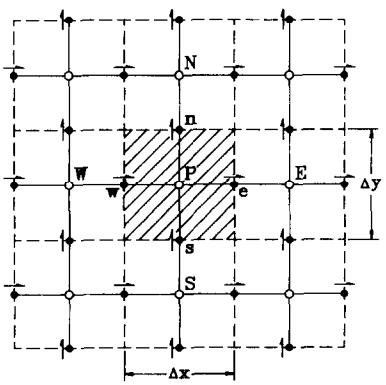

FIG. 1. Notations for staggered grid system.

incompressible flow. The white nodes are for physical quantities such as pressure, temperature, concentration, etc. Each black node with a horizontal arrow is for the velocity component u while that with a vertical arrow is for v. As in equation (5), discretization of the x-momentum equation at points e and w yields the two algebraic equations

$$u_e = \hat{u}_e - \gamma_e(\partial p/\partial x)_e \tag{6a}$$

$$u_w = \hat{u}_w - \gamma_w(\partial p/\partial x)_w. \tag{6b}$$

Similarly, the discretized y-momentum equations at points n and s are expressible as

$$v_n = \hat{v}_n - \gamma_n(\partial p/\partial y)_n \tag{7a}$$

$$v_s = \hat{v}_s - \gamma_s(\partial p/\partial y)_s. \tag{7b}$$

Next, substitute equations (6) and (7) into the discretized continuity equation

$$\frac{u_e - u_w}{\Delta x} + \frac{v_n - v_s}{\Delta y} = 0 \tag{8}$$

for point P. After rearrangement, the continuity equation (8) becomes the pressure-linked equation

$$\frac{\partial}{\partial x}\left(\gamma \frac{\partial p}{\partial x}\right) + \frac{\partial}{\partial y}\left(\gamma \frac{\partial p}{\partial y}\right) = \hat{\varepsilon} \tag{9a}$$

$$\hat{\varepsilon} = (\hat{u}_e - \hat{u}_w)/\Delta x + (\hat{v}_n - \hat{v}_s)/\Delta y \tag{9b}$$

where

$$\frac{\partial}{\partial x}\left(\gamma \frac{\partial p}{\partial x}\right)$$

denotes

$$\left[\gamma_e\left(\frac{\partial p}{\partial x}\right)_e - \gamma_w\left(\frac{\partial p}{\partial x}\right)_w\right]\Big/\Delta x$$

and the pseudo-dilation of the flow $\hat{\varepsilon}$ is not necessarily zero. Although equation (9a) is derived for a grid point away from the boundary, it is equally applicable for boundary grids. For instance, if point w is located at a boundary with a known velocity u_w, equation (6b) will be still valid by assigning $\hat{u}_w = u_w$ and $\gamma_w = 0$.

Using the pressure-linked equation (9) instead of the continuity equation seems to be a clever idea for incompressible flows. Unfortunately, there are serious numerical difficulties in solving equation (9) due to the fact that the pressure-linked equation has a singular coefficient matrix when it is discretized in the matrix form

$$[A]\{p\} = \{B\}. \tag{10}$$

The determinant of coefficient matrix $[A]$ can be proven to be zero by employing the property

$$\sum_{j=1}^{N} a_{ij} = 0 \tag{11}$$

where a_{ij} denotes the elements of matrix $[A]$ and N is the total number of grid points. This singular coefficient matrix might have caused a 'very slow convergence rate' for the solution as encountered in many previous studies [2].

CONCEPT OF ARTIFICIAL PRESSURE

The pressure-linked equation (9) or (10) indeed is identical to a heat conduction with given heat generation and boundary heat flux, if p, γ and $-\hat{\varepsilon}$ are regarded as temperature, thermal conductivity and heat generation, respectively. The solution thus does not exist unless the boundary heat flux precisely balances the heat generation. This sufficient condition for the existence of the pressure solution is equivalent to the compatibility condition needed in the pressure Poisson formulations [16–19]. In practical computations for incompressible flows, iterations are undertaken to update the velocity solution from the momentum equations with a guessed pressure field. A pressure solution corresponding to this velocity then is solved from the pressure equation. For an updated velocity, however, it can be proven that the compatibility condition holds only when this particular velocity 'happens' to satisfy the continuity equation. Unfortunately, this coincidence is very difficult to meet for an updated velocity due to numerical errors.

In the present study, an artificial source term is added to the right-hand side of equation (9a) to force the pressure solution to exist for each updated velocity. Such an artificial source could generate an extra pressure to compensate for the nonzero dilation of the updated velocity as mentioned earlier. However, it should be noted that the use of an artificial source forces only the overall mass conservation law to hold inside the whole flow domain. Its distribution thus is not necessarily uniform. In the present study, the pressure equation is assumed of the form

$$\frac{\partial}{\partial x}\left(\gamma \frac{\partial p}{\partial x}\right) + \frac{\partial}{\partial y}\left(\gamma \frac{\partial p}{\partial y}\right) = \hat{\varepsilon} - e|\varepsilon| \tag{12}$$

where $e|\varepsilon|$ is the artificial source. This assumption is made on three reasons. First, there is no need to correct the pressure for a particular location where the continuity equation has been satisfied ($\varepsilon = 0$). Second, the artificial pressure source should have a single sign (either positive or negative) over the entire physical domain to minimize the magnitude of the required artificial source. Third, an artificial source that is a linear function of the updated velocity was found to provide good numerical stability. Hence, the distribution of the artificial source is assigned proportional to $|\varepsilon|$ with the proportional constant e. The constant e will be defined such that the compatibility equation is always satisfied. The artificial source is expected to vanish if the source term $\hat{\varepsilon}$ precisely balances the boundary pressure gradient.

Now, let equation (12) be discretized in the matrix form

$$[A]\{p\} = \{B\} - e\{|\varepsilon|\} \tag{13a}$$

or

$$[a_{ij}]\{p_j\} = \{b_i\} - e\{|\varepsilon_i|\}. \tag{13b}$$

After applying a complete forward Gaussian elimination, equation (13b) becomes

$$[a_{ij}^*]\{p_j\} = \{b_i^*\} - e\{\varepsilon_i^*\} \tag{14}$$

which has the last row ($i = N$)

$$a_{NN}^* p_N = b_N^* - e\varepsilon_N^* \tag{15}$$

where $a_{NN}^* = 0$ due to the singularity of the coefficient matrix $[A]$. It is important to note that without the use of the artificial source ($e = 0$), equation (15) will become trivial and the value of P_N can be arbitrarily assigned if the values of b_N^* 'happen' to be zero. Otherwise, the pressure solution will not exist.

The requirement of $b_N^* = 0$ for the existence of the pressure solution seems to perfectly reflect the need of the compatibility condition. To compensate for the nonzero b_N^* value arising from the numerical error in each updated velocity, the e-value is assigned by

$$e = b_N^*/\varepsilon_N^*. \tag{16}$$

It appears from equation (16) that no artificial source will be needed if the condition $b_N^* = 0$ holds. Once the constant e is determined, the two vectors on the right-hand side of equation (14) can be added together. Next, assign an arbitrary constant to p_N and then perform a backward Gaussian elimination to obtain the pressure solution.

In the present formulation, the constant e is determined through the procedure of a direct Gaussian elimination. Fortunately, this can be accomplished also in an iterative solver such as the SIS solver [20]. In the use of the SIS solver, a matrix $[G]$ having only two nonzero diagonals is added to the original coefficient matrix $[A]$ such that the new coefficient matrix becomes $[A]+[G] = [L][U]$, where $[L]$ is a lower triangle matrix with only four nonzero diagonals while $[U]$ is an upper triangle matrix having only three nonzero diagonals and a central diagonal of unity. Let $\tilde{p}$ be a guessed pressure solution, then equation (13a) can be rewritten as

$$[L][U]\{p\} = \{F\} - e\{|\varepsilon|\} \tag{17}$$

where $\{F\}$ denotes the vector $\{B\} + [G]\{\tilde{p}\}$. The definition of the matrix $[G]$ can be found in ref. [20]. Next, perform a forward subtraction on equation (17) to yield

$$[U]\{p\} = \{F^*\} - e\{\varepsilon^*\} \tag{18}$$

where $\{F^*\} = [L]^{-1}\{F\}$ and $\{\varepsilon^*\} = [L]^{-1}\{|\varepsilon|\}$. Note that the last algebraic equation of the matrix equation (18) is

$$p_N = f_N^* - e\varepsilon_N^* \tag{19}$$

where p_N is an arbitrarily prescribed constant. The value of e thus can be defined to meet the compatibility equation (19), that is $e = (f_N^* - p_N)/\varepsilon_N^*$. Finally, renew the pressure solution by making a backward Gaussian elimination. This procedure should be iterated until the solution converges. Obviously, the e-value would vary from one iteration to another. Once the pressure solution converges, however, it will approach the same constant as that produced from equation (16) based on the direct Gaussian elimination procedure. This same technique can also be implemented in conventional solvers such as the SOR line-by-line and the ADI methods.

It is important to note that the primitive pressure does not appear in the governing equations (1)–(3). It always shows up in the derivative form. This implies that the sum of p and an arbitrary constant is also a solution if p is a solution. Such a situation seems to be perfectly reflected in the present numerical procedure that allows p_N to have an arbitrary value known as the pressure level [14]. If a pressure level (say, $p_N = 0$) is assigned, the pressure solution will uniquely exist. Hence, the pressure solution can be directly solved as described in this section without recourse to the conventional pressure correction equation [1–7, 9–13].

For convenience, the procedure of the APPLE (Artifical Pressure for Pressure-Linked Equation) algorithm is summarized as follows.

(1) Guess the velocity (u, v) and the pressure p.

(2) Based on the guessed velocity, evaluate the weighting factors (a_W, a_E, etc.) and the pseudo-velocity $\hat{u}$ and $\hat{v}$.

(3) Starting with the guessed pressure, perform *a few* SIS iterations on equation (17). This yields a pressure solution corresponding to the guessed velocity.

(4) Employ an SOR factor to modify the guessed pressure with the pressure solution obtained in step 3.

(5) Based on this modified guessed pressure, solve the momentum equations (4) with one SIS iteration to renew the velocity (u, v).

(6) If the solution (u, v, p) converges within a prescribed tolerance, then stop the computations. Otherwise, modify the updated velocity with an SOR factor and treat it as a new guessed velocity, then return to step 2 and repeat the entire computational procedure.

Strictly speaking, in step 3, the SIS solver should be iteratively performed on equations (17) until the pressure solution converges. The SOR factor used here could be as large as 1.0–1.2, because the pressure equation (12) is linear. This procedure is equivalent to performing a direct Gaussian elimination on the pressure equation. However, the guessed velocity might be not quite accurate before the solution converges. Thus, it is not necessary to obtain a completely converged pressure in step 3. In practical problems, only a few (say 10) SIS iterations are needed. In general, the optimum SOR factor is in the range of 0.3–0.5 in step 4 for the pressure and in the range of 0.1–0.3 in step 6 for the velocity.

THE NAPPLE ALGORITHM

With a minor revision, the APPLE algorithm can also be performed on a nonstaggered grid system. Figure 2 shows a nonstaggered grid system with a conventional notation for the grid points. The symbol Δx_i is the grid size in x-coordinate while $\overline{\Delta x_i}$ denotes $(\Delta x_{i-1}+\Delta x_i)/2$. Let the momentum equations be discretized at points E, W, N, S and the continuity equation be discretized at point P. This gives

$$u_E = \hat{u}_E - \gamma_E(\partial p/\partial x)_E \tag{20a}$$

$$u_W = \hat{u}_W - \gamma_W(\partial p/\partial x)_W \tag{20b}$$

$$v_N = \hat{v}_N - \gamma_N(\partial p/\partial y)_N \tag{20c}$$

$$v_S = \hat{v}_S - \gamma_S(\partial p/\partial y)_S \tag{20d}$$

and

$$\frac{u_E - u_W}{2\overline{\Delta x_i}} + \frac{v_N - v_S}{2\overline{\Delta y_j}} = 0. \tag{21}$$

As in the previous section, substituting equations (20) into equation (21) and making the assumption

$$\left[\gamma_E\left(\frac{\partial p}{\partial x}\right)_E - \gamma_W\left(\frac{\partial p}{\partial x}\right)_W\right]\Big/(2\overline{\Delta x_i}) = \frac{\partial}{\partial x}\left(\gamma\frac{\partial p}{\partial x}\right) = \left[\gamma_e\left(\frac{\partial p}{\partial x}\right)_e - \gamma_w\left(\frac{\partial p}{\partial x}\right)_w\right]\Big/\overline{\Delta x_i} \tag{22}$$

one obtains the pressure-linked equation

$$\frac{\partial}{\partial x}\left(\gamma\frac{\partial p}{\partial x}\right) + \frac{\partial}{\partial y}\left(\gamma\frac{\partial p}{\partial y}\right) = \hat{\varepsilon} \tag{23a}$$

$$\hat{\varepsilon} = (\hat{u}_E - \hat{u}_W)/(2\overline{\Delta x_i}) + (\hat{v}_N - \hat{v}_S)/(2\overline{\Delta y_j}) \tag{23b}$$

where e and w are the middle points of the line segments $\overline{PE}$ and $\overline{WP}$, respectively. The finite difference equation for equation (23a) is expressible as

$$b_W p_W + b_E p_E + b_S p_S + b_N p_N + b_P p_P = b_R$$

$$b_W = \gamma_w/(\Delta x_{i-1}\overline{\Delta x_i}), \quad b_E = \gamma_e/(\Delta x_i\overline{\Delta x_i})$$

$$b_S = \gamma_s/(\Delta y_{j-1}\overline{\Delta y_j}), \quad b_N = \gamma_n/(\Delta y_j\overline{\Delta y_j})$$

$$b_P = -b_W - b_E - b_S - b_N, \quad b_R = \hat{\varepsilon}_P. \tag{24}$$

As mentioned earlier, the coefficient $\gamma_E = -(1/a_P)_E$ is equivalent to the 'thermal conductivity' at point E if the pressure p is regarded as a temperature. However, γ is not a real material property. It comes from the coefficient of a finite difference equation and thus could have abrupt values from one grid point to another. Due to the lack of further reliable information, the value of γ_e is evaluated from the harmonic mean of γ_P and γ_E, i.e.

$$\gamma_e = -2/[(a_P)_P + (a_P)_E]. \tag{25}$$

In the use of equation (24), care must be exercised at the grid points adjacent to a boundary if the size of the boundary grid is large. For instance, when point W is located on the boundary, γ_W could have a zero value as demonstrated in the previous section. This will result in $\gamma_w = 0$ such that the mass flow between points W and P is ignored. Fortunately, this error can be remedied by employing the interpolation method

$$\frac{\partial u}{\partial x} = (u_e - u_w)/\overline{\Delta x_i}$$

$$u_e = \hat{u}_e - \gamma_e(\partial p/\partial x)_e$$

$$\hat{u}_e = (\hat{u}_P + \hat{u}_E)/2$$

$$u_w = (u_W + u_P)/2 \tag{26}$$

such that the weighting factor b_W in equation (24) becomes zero and the pseudo-dilation $\hat{\varepsilon}_P$ is

$$\hat{\varepsilon}_P = (\hat{u}_E + \hat{u}_P - u_P - u_W)/(2\overline{\Delta x_i}) + (\hat{v}_N - \hat{v}_S)/(2\overline{\Delta y_j}). \tag{27}$$

However, this treatment is not necessary if the grid size is sufficiently small along the boundaries. For convenience, this new method will be referred to as the NAPPLE (Nonstaggered APPLE) algorithm. Its computational procedure is the same as that of the APPLE algorithm described in the previous section.

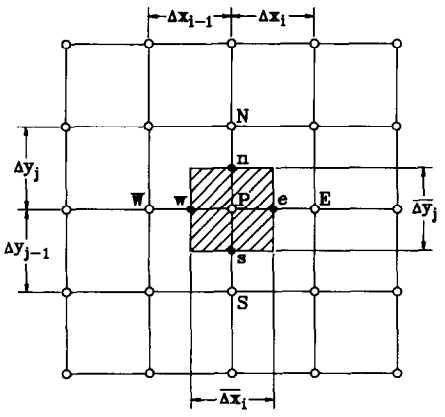

FIG. 2. Notations for nonstaggered grid system.

PERFORMANCES OF THE NEW ALGORITHMS

In this section, the performances of the APPLE and the NAPPLE algorithms will be examined through two well-known examples. The results obtained by using the present algorithms are compared with that based on the SIMPLER algorithm [2, 3] and the stream–vorticity formulation. Example 1 is conducted to study the efficiencies of the APPLE and the NAPPLE algorithms for a viscosity-driven flow in a square cavity. In example 2, a natural convection inside a square enclosure is employed to test the performances of both new algorithms in the presence of buoyancy term. Generally speaking, a good initial guess helps solution convergence in solving highly nonlinear problems like examples 1 and 2. A poor initial guess might even lead to a divergent result. For unsteady flows, the solution at the previous time step ($t_0 = t - \Delta t$) makes a good initial guess for the solution at the present time t as long as the time step Δt is not large. In steady flows, however, the solution

procedure usually has to start from an actually guessed solution. Therefore, solving a steady flow is more difficult than solving an unsteady flow in practical applications. For this reason, no example for unsteady flow is attempted in the present study.

Example 1. *Viscosity-driven flow inside a square cavity*

Consider a two-dimensional incompressible flow inside a square cavity. The lid of the cavity ($y = 1$) moves at a constant speed in the x-direction while the other three walls $x = 0$, $x = 1$ and $y = 0$ are maintained stationary. A circulating flow thus is induced inside the cavity due to viscous force. Such a circulating flow is governed by equations (1)–(3) subject to the boundary conditions

$$u = 0, v = 0 \quad \text{at } x = 0, x = 1, y = 0 \tag{28a}$$

$$u = 1, v = 0 \quad \text{at } y = 1. \tag{28b}$$

To solve this circulating flow, a staggered uniform grid system with $(n-1)\times(n-1)$ control volumes is employed for both SIMPLER and APPLE algorithms. Such a grid system possesses $n\times(n-1)$ grid points for each of the momentum equations (2) and (3). In the use of the NAPPLE algorithm, all of the variables u, v and p are determined on a non-staggered uniform grid system with $n\times n$ grid points. All of the differential equations in this example are solved by the weighting function scheme [15] and the SIS solver [20] for the parameter of $Re = 400$.

Figure 3(a) reveals the results of $u(0.5, y)$ based on the APPLE and SIMPLER algorithms for various grid resolutions ($n = 41$, 61 and 81). The result obtained by the stream–vorticity method [21] with a sufficiently fine grid ($n = 129$) is also plotted in Fig. 3(a) as a benchmark. Similar information is provided in Fig. 3(b) for $v(x, 0.5)$. The physical significance of Fig. 3 has been well discussed in the literature. Thus, it is not repeated here to conserve space.

It is noteworthy that on staggered grid systems, the pressure equation (9) is an *exact combination* of the discretized momentum equation (4) and the discretized continuity equation (8). This implies that the exact solution of the algebraic equations (4) and (9) will automatically satisfy the discretized continuity equation (8). In the solution procedure of the APPLE algorithm, the use of the artificial source $e|\varepsilon|$ is only to help the solution convergence. Once the solution converges, both the artificial source $e|\varepsilon|$ and the dilation of the velocity $|\varepsilon|$ will disappear. In the SIMPLER algorithm, the pressure correction equation is solved to correct only the velocity. Instead of being updated from the pressure correctness, the pressure solution is estimated from the updated velocity by applying an iterative solver on the pressure equation for only one or a few iterations. Such a strategy does not accumulate errors to the pressure solution and thus could produce a converged velocity field if the velocity is not very sensitive to the pressure gradients in the momentum equations. Like the

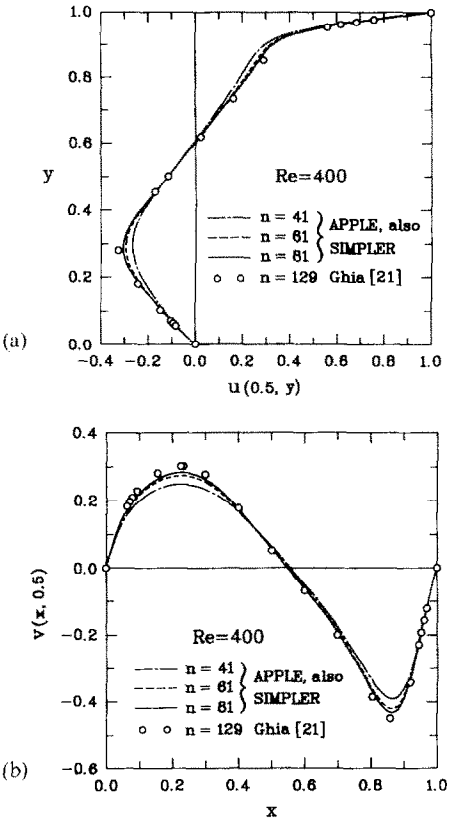

FIG. 3. (a) The $u(0.5, y)$ results provided by APPLE, SIMPLER and ψ–ω methods with various grid resolutions for example 1. (b) The $v(x, 0.5)$ results provided by APPLE, SIMPLER and ψ–ω methods with various grid resolutions for example 1.

APPLE algorithm, the SIMPLER algorithm also provides converged velocity of zero dilation. Due to the fact that both converged velocities based on the APPLE and the SIMPLER algorithms satisfy the same algebraic equations (4) and (8), the difference between them was found to be within the tolerance of the convergence criterion. Hence, the same curves are employed to represent both results in Fig. 3. This is important evidence that the concept of artificial pressure can be used instead of the pressure correction equation without losing accuracy.

In the derivation of the discretized continuity equation (8), a central finite difference scheme was employed. Due to the truncation error, equation (8) could be not quite accurate in some particular locations where the velocity profile has a large curvature. This can be clearly observed from the horizontal velocity $u(0.5, y)$ near $y = 0.3$ in Fig. 3(a) and the vertical velocity $v(x, 0.5)$ near $x = 0.87$ in Fig. 3(b). Nevertheless, the results produced by both APPLE and SIMPLER algorithms approach the 'exact' solution [21] when the number of grid points increases from $n = 41$ to $n = 81$. The accuracy in the above-mentioned locations can be further improved by increasing the grid resolution.

As expected, the CPU time needed by the APPLE

algorithm in example 1 is only about 40% of that required by the SIMPLER algorithm. Other SIMPLE-like algorithms such as SIMPLEC [4] and PISO [6] show similar behaviors as exhibited by the SIMPLER algorithm. The high efficiency of the APPLE algorithm seems to come from two reasons. First, unlike the SIMPLE algorithm and its variants [2–7, 9–13], the APPLE algorithm solves the pressure solution directly without the need of solving an additional pressure correction equation. Second, the existence and uniqueness of the pressure solution due to the use of artificial pressure brings about a fast convergence rate for the APPLE algorithm.

It is important to note that during the iterations, the pressure solution does not exist in the sense of numerical analysis due to its singular coefficient matrix and the nonzero dilation of the updated velocity. Therefore, the pressure solution cannot be really obtained without employing a technique equivalent to the artificial pressure. In the SIMPLE algorithm [1, 2], a pressure correctness is added to the pressure solution in each iteration. Such an attempt to obtain a converged pressure solution thus will undoubtedly fail. This might account for the fact that the solution is very difficult to 'convergence' in the use of the SIMPLE algorithm. In the SIMPLER algorithm, the solution procedure is terminated as long as the dilation $|\varepsilon|$ is below a prescribed tolerance. No converged pressure solution is attempted. In many practical applications, unfortunately, the pressure gradient shows significant influences on the velocity. Under such situations, using the SIMPLER algorithm it would be very difficult to produce converged results as encountered in many previous studies [11] owing to the lack of an accurate pressure solution. In addition, the SIMPLER algorithm will fail also if one uses the direct Gaussian elimination method to solve the pressure equation. Fortunately, the APPLE algorithm poses none of these difficulties.

The APPLE algorithm seems to possess good performances for incompressible flows. However, its application is restricted to staggered grid systems. On nonstaggered grid systems, the continuity equation is very difficult to convert into a pressure equation. This is simply because on nonstaggered grid systems no velocity solution is known at the control surfaces (u_e, u_w, v_n, v_s) as demonstrated in Fig. 2. To circumvent this difficulty, a pressure-weighted interpolation for these particular velocities known as PWIM has been employed by previous investigators [9, 10, 12, 13]. However, the interpolation procedure used in PWIM introduces a numerical error of $O((\Delta x)^2)$ into the velocities such that the resulting pressure equation is no longer an exact combination of the discretized momentum equation and the discretized continuity equation. Therefore, the dilation of the velocity will not vanish even though the solution has perfectly converged. It would be even worse in the PWIM formulation that the nonzero dilation is multiplied by a relaxation factor. For this reason, the solution produced by PWIM depends on the relaxation factor as pointed out by Patankar [11], although the dependence might not be so significant under some particular treatments [12, 13]. Unfortunately, these treatments could give rise to a physically impossible velocity in a region where the pressure gradient has a rapid variation [13].

Logically, the undesired feature of nonzero dilation on nonstaggered grid systems is unavoidable. Such a numerical difficulty seems to exist also in the pressure Poisson formulation [8, 16–19] that satisfies the continuity equation indirectly. In this branch of methodology, an explicit scheme is employed to extrapolate the velocity solution from the momentum equation. A pressure solution corresponding to this extrapolated velocity then is generated by solving a pressure Poisson equation in the form

$$p_{xx}+p_{yy}=\sigma-\varepsilon_t=\sigma+\varepsilon/\Delta t \tag{29}$$

where $\sigma=2(u_xv_y-v_xu_y)$, t is a virtual time coordinate and $\varepsilon_t=(\varepsilon_1-\varepsilon)/\Delta t$. In equation (29), the dilation ε_1 in the time level $t+\Delta t$ has been assigned zero. It should be noted that the virtual time step Δt in equation (29) is equivalent to the relaxation factor of implicit schemes. Its value is prescribed. The pressure solution thus will depend on Δt unless the converged velocity is free of nonzero dilation. In explicit schemes, as remarked by Ghia *et al.* [16], the dilation approaches a constant rather than zero at convergence even though the compatibility condition is well satisfied. This is simply because the compatibility condition emphasizes only the global mass conservation. No zero local dilation is guaranteed. For this reason, solutions based on pressure Poisson equation and explicit scheme are all Δt-dependent. To clarify this point, example 1 is resolved by using the pressure Poisson method proposed by Biringen and Cook [19]. The 'converged' average dilation $\overline{|\varepsilon|}$ and the corresponding virtual unsteady term $\overline{|\varepsilon/\Delta t|}$ are presented in Fig. 4 for various virtual time steps and grid resolutions.

As observable from Fig. 4, for a given grid resol-

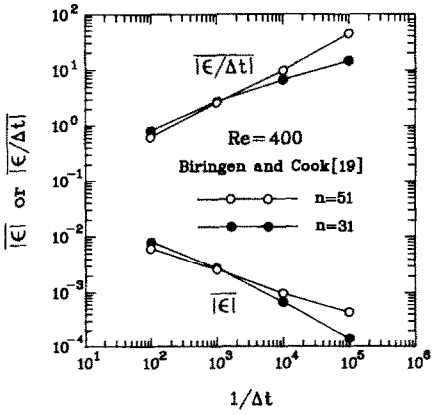

FIG. 4. The residual of the continuity equation $\overline{|\varepsilon|}$ and the corresponding virtual unsteady term $\overline{|\varepsilon/\Delta t|}$ produced by conventional pressure Poisson equation [19] for example 1.

ution, increasing the value of $1/\Delta t$ decreases the dilation $\overline{|\varepsilon|}$. This will achieve a better accuracy for the solution. However, increasing the grid resolution at a given virtual time step is not necessary to improve the solution accuracy. For instance, at $1/\Delta t = 10^5$, the residual dilation is $\overline{|\varepsilon|} = 4.34 \times 10^{-4}$ for $n = 51$ as compared to $\overline{|\varepsilon|} = 1.45 \times 10^{-4}$ for $n = 31$. This interesting phenomenon seems to arise from the inherent numerical instability of the explicit scheme. In the explicit scheme, the accuracy of the solution depends very strongly on the time step parameter $\lambda = \Delta t/(\Delta x)^2$. The solution will even diverge if the parameter λ exceeds its critical value. Hence, the maximum possible time step would reduce to one quarter of its original size if the grid size Δx has to be halved. At $1/\Delta t = 10^5$, the grid resolution $n = 51$ corresponds to $\lambda = 0.025$, while $n = 31$ has a λ-value as small as 0.009. This means that fine grid system possesses large λ-value and thus does not necessarily provide better accuracy for the solution.

In the formulation of the NAPPLE algorithm, the approximation (22) is made to modify the Laplace operator such that no interpolation is needed for the velocities at the control surfaces. However, this particular treatment also introduces a truncation error into the discretized continuity equation (21). Fortunately, unlike $1/\Delta t$ in the pressure Poisson formulation, the e-value appearing in the artificial source is not a prescribed parameter. Hence, the converged solution provided by the NAPPLE algorithm is independent of the SOR factor needed in the SIS solver, even though the dilation of the converged solution is nonzero. In fact, the artificial source $e|\varepsilon|$ depends only on the truncation error. Its value vanishes when the grid size approaches zero. This will be demonstrated later.

Figures 5(a) and (b) show, respectively, the velocity $u(0.5, y)$ and $v(x, 0.5)$ produced by the NAPPLE algorithm on a nonstaggered uniform grid system of $n \times n$ points. As mentioned in the previous paragraph, the converged velocity is independent of the SOR factor. For convenience of comparison, the results of Ghia *et al.* [21] are also plotted in Fig. 5. From Figs. 3 and 5, one sees that for $n = 41$ the accuracy of the NAPPLE algorithm is not as good as that of the APPLE algorithm due to the use of the approximation (22). Fortunately, when the step size is reduced as $n \geqslant 81$, the discrepancy between the results produced by APPLE and NAPPLE algorithms seems negligible.

Figure 6 reveals the pressure distribution corresponding to the velocity shown in Fig. 5. This pressure result is provided by the NAPPLE algorithm on a nonstaggered grid system of 81×81 points. As described in equation (28), the physical domain is defined in the region of $0 \leqslant x \leqslant 1$ and $0 \leqslant y \leqslant 1$. For convenience, the isobars in Fig. 6 are plotted with the pressure level $p(1, 1) = 0$ and an increment of $\Delta p = 0.02$. Essentially the same pressure result was obtained by using the APPLE algorithm.

Figure 7 shows the 'converged' average dilation $\overline{|\varepsilon|}$

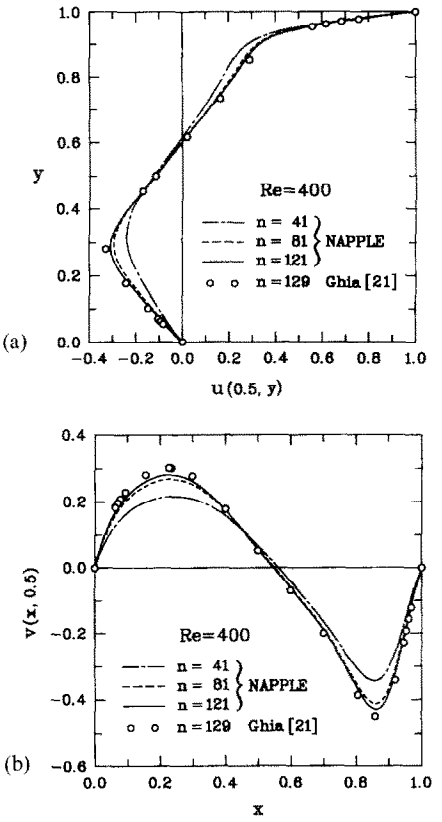

FIG. 5. (a) The $u(0.5, y)$ results provided by the NAPPLE algorithm with various grid resolutions for example 1. (b) The $v(x, 0.5)$ results provided by the NAPPLE algorithm with various grid resolutions for example 1.

and its corresponding average artificial source $-e\overline{|\varepsilon|}$ for various n-values due to the approximation (22) employed in the NAPPLE algorithm. It should be noted that the 'mass source' defined in conventional SIMPLE-like algorithm [2] is $\varepsilon \Delta x \Delta y$. Hence, $\varepsilon = 10^{-2}$ is equivalent to a mass source of 10^{-6}, if the grid size $\Delta x = \Delta y = 0.01$ is used. As expected, both $\overline{|\varepsilon|}$ and $-e\overline{|\varepsilon|}$ have decreasing magnitudes when the truncation error is reduced by increasing the number of the grid points (the n-value). This means that the

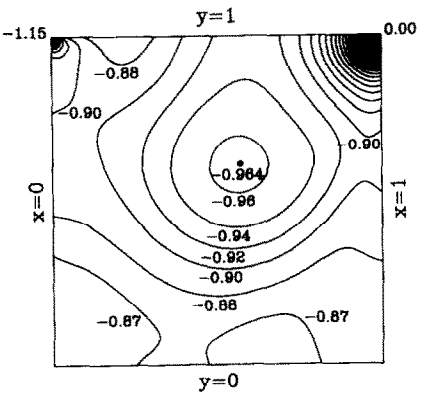

FIG. 6. The pressure distribution produced by the NAPPLE algorithm with $n = 81$ for example 1.

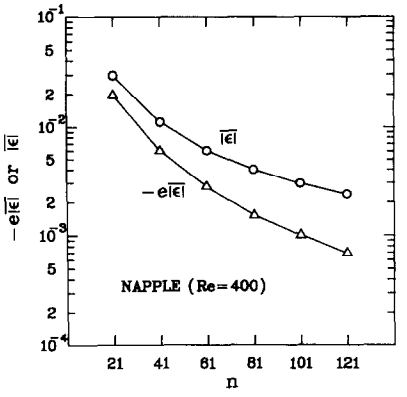

FIG. 7. The residual of the continuity equation $|\varepsilon|$ and its corresponding artificial source/sink required by the NAPPLE algorithm for example 1.

NAPPLE algorithm can be expected to produce accurate results as long as the grid size is sufficiently small. The excellent agreement between the results of the APPLE and the NAPPLE algorithms as observable from Figs. 3 and 5 for $n > 81$ seems to substantiate this point. This important behavior of the NAPPLE algorithm is different from that of the pressure Poisson formulation. As mentioned earlier, when the grid size decreases, the pressure Poisson formulation could produce poorer results due to an increased time step parameter λ.

Example 2. Natural convection inside a square enclosure

The conservation equations for the natural convection inside a rectangular enclosure are expressible as the dimensionless form

$$u_x + v_y = 0 \tag{30a}$$

$$uu_x + vu_y = -p_x + Pr(u_{xx} + u_{yy}) \tag{30b}$$

$$uv_x + vv_y = -p_y + Pr(v_{xx} + v_{yy}) + Ra\,Pr\,\theta \tag{30c}$$

$$u\theta_x + v\theta_y = \theta_{xx} + \theta_{yy}. \tag{30d}$$

The associated boundary conditions are

$$\begin{aligned} &u = v = 0, \quad \theta = 0.5 \quad \text{at } x = 0 \\ &u = v = 0, \quad \theta = -0.5 \quad \text{at } x = 1 \\ &u = v = 0, \quad \partial\theta/\partial y = 0 \quad \text{at } y = 0 \text{ and } y = 1. \end{aligned} \tag{31}$$

As in example 1, the APPLE, NAPPLE and SIMPLER algorithms are employed to solve equations (30) and (31) on a uniform grid system for the case of $Pr = 0.7$ and $Ra = 10^7$. The numerical procedures for each of these algorithms are the same as their counterparts in example 1. The only difference is that example 1 deals with a moving plate while example 2 studies the effect of the buoyancy term $Ra\,Pr\,\theta$. This same problem was also solved by using the stream–vorticity (ψ–ω) formulation. The solution based on the ψ–ω method seems to converge when the number of the grid points is larger than $n \times n = 241 \times 241$. This result will be regarded as the 'exact' solution.

Figures 8(a) and (b) show, respectively, the numerical results of $u(0.5, y)$ and $v(x, 0.5)$ obtained by the APPLE and the SIMPLER algorithms with $n = 41$, 81 and 121 and the stream–vorticity formulation with $n = 241$. Again, the results produced by APPLE and SIMPLER algorithms are essentially the same such that the same curves are used to present both results in Fig. 8. In this example, the natural convection is driven by the buoyancy force $Ra\,Pr\,\theta$ that appears in the source term of equation (30c). After discretizing equation (30c), the right-hand side coefficient a_R in equation (4) is found proportional to the size of the control volume (i.e. $(\Delta x)^2$ in the present case), while the coefficients a_W, a_E, etc., are in order of unity. Hence, the solution accuracy for buoyancy-driven flows can be effectively improved by reducing the grid size.

Figure 8 reveals that both APPLE and SIMPLER algorithms produce good results when the number of the grid points is larger than 81×81. As in the forced flow case, however, higher grid resolution is needed to improve the solution in some particular locations where the curvature of the velocity profile is large. For the case of low Rayleigh number, the curvature of the

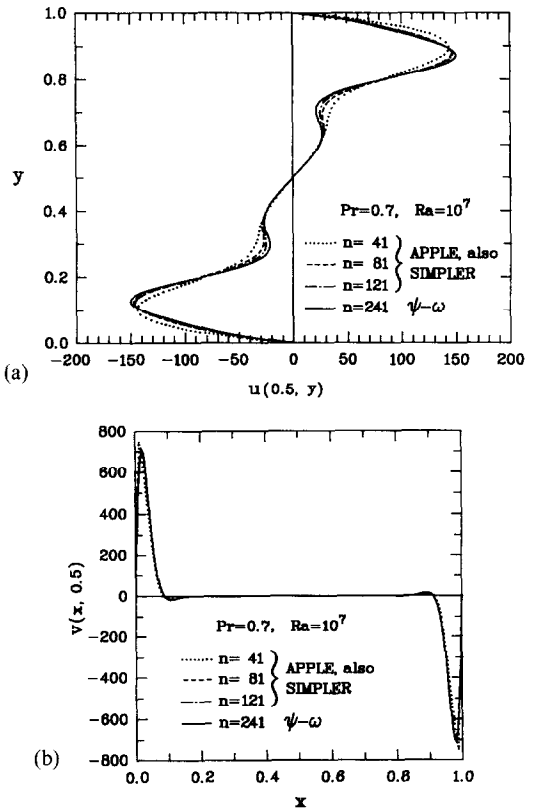

FIG. 8. (a) The $u(0.5, y)$ results provided by APPLE, SIMPLER and ψ–ω methods with various grid resolutions for example 2. (b) The $v(x, 0.5)$ results provided by APPLE, SIMPLER and ψ–ω methods with various grid resolutions for example 2.

velocity profile is generally small. Under this situation, very accurate results can be achieved even on a coarse grid system. For instance, based on a 41×41 grid system, both APPLE and SIMPLER algorithms provide a velocity solution with a maximum error of below 0.5% for $Ra = 10^5$.

The results of $u(0.5, y)$ and $v(x, 0.5)$ based on the NAPPLE algorithm and $n = 81$, 121 and 161 are presented in Figs. 9(a) and (b), respectively. The 'exact' solution based on the ψ–ω formulation with $n = 241$ is also plotted in Fig. 9 for comparison. It is noted that in the formulation of the NAPPLE algorithm, the continuity equation is discretized on a control volume having the size of $4\Delta x_i \Delta y_j$ (see equation (21)). In addition, the approximation (22) is used to avoid a checkerboard error for the pressure solution. These treatments undoubtedly will introduce a truncation error into the pressure solution. Nevertheless, this error can be expected to be negligible if the grid size is sufficiently small. Like the cavity flow in example 1, the buoyancy-driven flow predicted by the NAPPLE algorithm is essentially the same as that by the APPLE algorithm when the number of grids is no less than 81×81. This can be observed by comparing the results in Fig. 9 with their counterparts in Fig. 8. Therefore, it is believed that the truncation error arising from equations (21) and (22) in the use of the NAPPLE algorithm is dominated by the size of the control volume. In many practical situations, small grids equivalent to $n = 81$ are generally needed to obtain a good resolution for the solution. Under such a situation, the truncation error arising from equations (21) and (22) would be essentially negligible as compared with errors from other modes. As a final note, it is mentioned that the programming of the NAPPLE algorithm is very simple because it is performed on non-staggered grid system. Due to its simplicity, the idea of the NAPPLE algorithm can be easily extended and performed on curvilinear coordinates for arbitrarily-shaped domains.

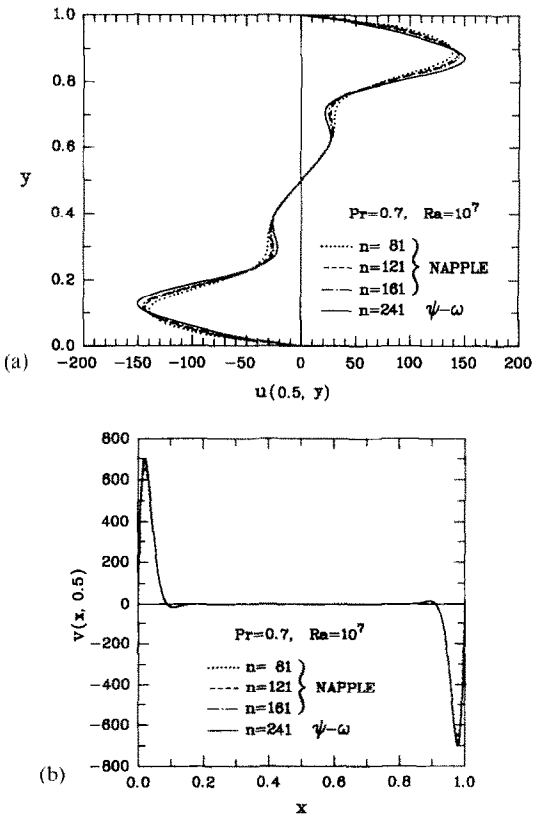

FIG. 9. (a) The $u(0.5, y)$ results provided by the NAPPLE algorithm with various grid resolutions for example 2. (b) The $v(x, 0.5)$ results provided by the NAPPLE algorithm with various grid resolutions for example 2.

CONCLUSION

The discretized pressure-linked equations can be proven to possess singular coefficient matrix. The pressure solution corresponding to a given velocity thus would not exist unless this velocity precisely satisfies the continuity equation. This might be the major reason why the pressure solution is very difficult to solve as reported by many previous investigators. Fortunately, through the use of the concept of artificial pressure proposed in the present study, uniqueness and existence of the pressure solution are guaranteed for each updated velocity. In addition, the pressure level is allowed to be arbitrarily assigned. Based on the concept of artificial pressure, the APPLE and NAPPLE algorithms are proposed for incompressible flows. Both new algorithms directly solve the pressure equation without recourse to the conventional pressure correction equation. The performances of both algorithms are examined through a forced convection example and a natural convection example inside a square enclosure. From these two well-known examples, the APPLE algorithm was found to produce essentially the same results with only about 40% CPU time as compared with the SIMPLER algorithm. This means that the concept of artificial pressure can be used instead of the conventional pressure correction equation without losing accuracy. For both examples, the truncation error due to the approximations made in the NAPPLE algorithm for non-staggered grids is seen to be negligible as long as the grid size is sufficiently small. Due to its simplicity, the NAPPLE algorithm can be easily extended and performed on complicated problems such as a three-dimensional flow with an arbitrarily-shaped domain.

Acknowledgement—The authors wish to express their appreciation to the National Science Council of the Republic of China in Taiwan for the financial support of this work through the project NSC80-0401-E007-12.

REFERENCES

1. 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**, 1787–1806 (1972).

2. S. V. Patankar, *Numerical Heat Transfer and Fluid Flow*, Chap. 6. Hemisphere, New York (1980).
3. S. V. Patankar, A calculation procedure for two-dimensional elliptic situations, *Numer. Heat Transfer* **4,** 409–425 (1981).
4. J. P. Van Doormaal and G. D. Raithby, Enhancement of the SIMPLE method for predicting incompressible flows, *Numer. Heat Transfer* **7,** 147–163 (1984).
5. D. B. Spalding, Mathematical modelling of fluid mechanics, heat transfer and mass transfer processes, Report No. HTS/80/1, Mechanical Engineering Department, Imperial College, London (1980).
6. R. I. Issa, Solution of the implicitly discretized fluid flow equations by operator-splitting, *J. Comput. Phys.* **62,** 40–65 (1985).
7. B. R. Latimer and A. Pollard, Comparison of pressure–velocity coupling solution algorithms, *Numer. Heat Transfer* **8,** 635 652 (1985).
8. F. H. Harlow and J. E. Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface, *Physics Fluids* **8,** 2182–2189 (1965).
9. C. M. Rhie and W. L. Chow, Numerical study of the turbulent flow past an airfoil with trailing edge separation, *AIAA J.* **21,** 1525–1532 (1983).
10. M. Peric, A finite volume method for the prediction of three dimensional fluid flow in complex ducts, Ph.D. Thesis, University of London (1985).
11. S. V. Patankar, Recent developments in computational heat transfer, *J. Heat Transfer* **110,** 1037–1045 (1988).
12. S. Majumdar, Role of underrelaxation in momentum interpolation for calculation of flow with nonstaggered grid, *Numer. Heat Transfer* **13,** 125–132 (1988).
13. T. F. Miller and F. W. Schmidt, Use of a pressure-weighted interpolation method for the solution of the incompressible Navier–Stokes equations on a non-staggered grid system, *Numer. Heat Transfer* **14,** 213–233 (1988).
14. R. J. Roache, *Computational Fluid Dynamics*, pp. 180–185. Hermosa, Albuquerque, New Mexico (1972).
15. S. L. Lee, Weighting function scheme and its application on multidimensional conservation equations, *Int. J. Heat Mass Transfer* **32,** 2065–2073 (1989).
16. K. N. Ghia, W. L. Hankey, Jr and J. K. Hodge, Use of primitive variables in the solution of incompressible Navier–Stokes equations, *AIAA J.* **17,** 298–301 (1979).
17. S. Abdallah, Numerical solutions for the pressure Poisson equation with Neumann boundary conditions using a non-staggered grid, I, *J. Comput. Phys.* **70,** 182–192 (1987).
18. S. Abdallah, Numerical solutions for the incompressible Navier–Stokes equations in primitive variables using a non-staggered grid, II, *J. Comput. Phys.* **70,** 193–202 (1987).
19. S. Biringen and C. Cook, On pressure boundary conditions for the incompressible Navier–Stokes equations using nonstaggered grids, *Numer. Heat Transfer* **13,** 241–252 (1988).
20. S. L. Lee, A strongly implicit solver for two-dimensional elliptic differential equations, *Numer. Heat Transfer* **16B,** 161–178 (1989).
21. U. Ghia, K. N. Ghia and C. T. Shin, High-*Re* solutions for incompressible flow using the Navier–Stokes equations and a multigrid method, *J. Comput. Phys.* **48,** 387–411 (1982).

PRESSION ARTIFICIELLE POUR DES EQUATIONS LIEES A LA PRESSION

Résumé—L'équation discrétisée liée à la pression possède une matrice singulière de coefficient. Ceci implique que la solution de pression correspondant à une vitesse donnée n'existe pas à moins que cette vitesse satisfasse l'équation de continuité. Dans cette étude, un terme source artificiel est ajouté dans l'équation liée à la pression. Ce terme génère une extra-pression appelée 'pression artificielle' pour chaque vitesse, ce qui est équivalent à la création d'une extra-masse afin de satisfaire l'équation de continuité. Ce traitement garantit l'existence et l'unicité de la solution de pression de façon à résoudre directement la pression sans avoir recours à l'équation de correction de pression conventielle. Basés sur le concept de pression artificielle, les algorithmes APPLE et NAPPLE sont développés pour les fluides incompressibles. A travers deux exemples bien connus, l'algorithme APPLE fournit les mêmes résultats avec seulement 40% du temps CPU en comparaison avec l'algorithme SIMPLER. Par sa simplicité, NAPPLE a une potentialité pour résoudre les problèmes sur des domaines de forme arbitraire si la maille n'est pas trop large.

KÜNSTLICHER DRUCK IN EINER DRUCKABHÄNGIGEN GLEICHUNG

Zusammenfassung—Es kann gezeigt werden, daß die diskretisierte druckabhängige Gleichung eine singuläre Koeffizientenmatrix besitzt. Daraus ergibt sich, daß zu einer gegebenen Geschwindigkeit eine Lösung für den Druck nur dann existiert, wenn die Kontinuitätsgleichung exakt erfüllt ist. In der vorliegenden Arbeit wird in die druckabhängige Gleichung ein künstlicher Quellterm eingefügt. Dieser Zusatzterm erzeugt einen zusätzlichen Druck, einen sogenannten "künstlichen Druck", um bei jeder neu berechneten Geschwindigkeit deren Nullpunktsabweichung zu eliminieren. Die Verwendung eines Zusatzdruckes entspricht der Erzeugung zusätzlicher Masse zur Erfüllung der Kontinuitätsgleichung. Hierdurch wird die Existenz und Eindeutigkeit der Lösung für den Druck garantiert, und die übliche Rekursionslösung wird überflüssig. Auf dem Konzept des künstlichen Druckes basierend wurden zwei Algorithmen, APPLE und NAPPLE, für inkompressible Strömungen entwickelt. Mit Hilfe zweier wohlbekannter Beispiele wird gezeigt, daß APPLE im Vergleich zum SIMPLER-Algorithmus bei nahezu gleichen Ergebnissen nur 40% CPU Zeit benötigt. Wegen seiner Einfachheit kann der NAPPLE-Algorithmus auf Probleme bei beliebig geformten Berandungen angewandt werden, solange die Gitterweite nicht zu groß ist.

"ИСКУССТВЕННОЕ" ДАВЛЕНИЕ В УРАВНЕНИИ ДВИЖЕНИЯ

Аннотация—Можно доказать, что дискретизированное уравнение, содержащее давление, имеет сингулярную матрицу коэффициентов. Это означает, что решения для давления, соответствующего даннрй скорости, не будет существовать, если скорость не будет точно удовлетворять уравнению неразрывности. В настоящем исследовании в уравнение для давления вводится слагаемое с искусственным источником, которое приводит к избыточному так называемому "искусственному" давлснию для каждого значсния скорости, компенсирующему отличное от нуля объемное расширение. Использование "искусственного" давления равноценно созданию дополнительной массы с той целью, чтобы новое значение скорости удовлетворяло уравнению неразрывности. Предложенный подход обеспечивает существование и единственность решения для давления, так что давление можно определить непосредственно. На основе концепции "искусственного" давления разработаны алгоритмы APPLE (искусственное давление для уравнения, содержащего давление) и NAPPLE (непрерывное APPLE) для несжимаемых течений. На двух известных примерах показано, что алгоритм APPLE дает выигрыш во времени 40% по сравнению с алгоритмом SIMPLER. Благодаря своей простоте алгоритм NAPPLE может использоваться для решения задач с произвольной конфигурацией области при условии мелкой сетки.