Numerical Heat Transfer, Part B, 46: 529–548, 2004

ISSN: 1040-7790 print/1521-0626 online
DOI: 10.1080/10407790490487622

TEMPORAL SECOND-ORDER ACCURACY OF SIMPLE-TYPE METHODS FOR INCOMPRESSIBLE UNSTEADY FLOWS

Ming-Jiu Ni and Mohamed A. Abdou
UCLA MAE Department, Engineering IV,
Los Angeles, California, USA

SIMPLE-type methods are presented in a more concise formulation. This formulation is used to analyze temporal accuracy for unsteady flows. Detailed error formulations are given. Analysis shows that SIMPLE-type methods have second-order temporal accuracy if a second-order temporal updating technique is employed to update both the convective and diffusion terms. Two algorithms for unsteady flows are presented. Algorithm A is an iteration method, which will cost more time than algorithm B, a noniteration algorithm. Also several second-order updating techniques are presented. A classical validation example is employed to validate the temporal accuracy in this article. A new four-step SIMPLE-type method is presented, in which the pressure Poisson equation, not the pressure difference Poisson equation is solved.

1. INTRODUCTION

The dimensionless unsteady incompressible Navier–Stokes equations in primitive variables can be written as

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{u} = -\nabla p + \frac{1}{\mathrm{Re}} \nabla^2 \boldsymbol{u} \tag{1a}$$

$$\nabla \cdot \boldsymbol{u} = 0 \tag{1b}$$

where $\boldsymbol{u}$ and p are the nondimensional velocity vector and kinetic pressure, respectively, and Re is the Reynolds number. A central issue in the design of numerical methods for Navier-Stokes equations is the development of an appropriate discrete form of the incompressibility constraint. Famous primitive-variable numerical methods include the MAC method [1], projection methods [2–8], and the SIMPLE-type method (Semi-Implicit Method for Pressure-Linked Equations) [9–17]. PISO [18] is also a SIMPLE-type method. They all have been used extensively and have

Received 22 December 2003; accepted 27 April 2004.
Address correspondence to Ming-Jiu Ni, UCLA MAE Department, 43-133 Engineering IV, 420 Westwood Plaza, Los Angeles, CA 90095-1597, USA. E-mail: mingjiu@fusion.ucla.edu

served well. MAC is an explicit transient algorithm. The projection and SIMPLE methods are implicit algorithms. Implicit methods are attractive as a means of avoiding restrictions on the explicit time step. Projection methods have been extensively applied to unsteady flows, whereas SIMPLE-type methods have been successfully applied to steady flows, especially for the numerical calculations of heat transfer. The standard SIMPLE method [9] was developed in 1972. Patankar [10] produced one of the seminal works in the introduction of numerical techniques of incompressible flows. SIMPLE-type methods include SIMPLE [9, 10], SIMPLER [11], SIMPLEC [12], PISO [18], and others. Recent development of the SIMPLE method can be found in [13, 14]. Although the SIMPLE method was developed primarily to simulate steady flows, it has also been applied to unsteady flows [19], unsteady interfacial flows [13, 15, 16], and even direct numerical simulation (DNS) research turbulence [17]. However, no detailed temporal accuracy analysis has been done for the calculation of unsteady flows using Patankar's SIMPLE method. This article presents detailed error formulations of SIMPLE-type methods.

In Section 2, new concise formulations of the standard SIMPLE method are shown. The temporal accuracy analysis is done based on the concise formulations. In Section 3, underrelaxation techniques for unsteady flows are analyzed and the SIMPLEC method for unsteady flows is shown. A four-step SIMPLE-type method is also presented in this section. In Section 4, two algorithms of SIMPLE-type methods for unsteady flows are given. Several second-order updating techniques are employed to obtain second-order-temporal-accuracy SIMPLE methods. An example is simulated using SIMPLE-type methods to validate the analysis in Section 5. Section 6 summarizes the conclusions of this article.

2. CONCISE FORMULATIONS AND ACCURACY ANALYSIS OF STANDARD SIMPLE METHOD

Straightforward discretization of Eqs. (1*a*) and (1*b*) can be written as

$$\frac{\boldsymbol{v}^{n+1} - \boldsymbol{v}^n}{\Delta t} + \boldsymbol{\theta}^n \nabla p^{n+1} = \left(\frac{\nabla^2 \boldsymbol{v}}{\mathrm{Re}}\right)^{n+1/2} - (\boldsymbol{v} \cdot \nabla \boldsymbol{v})^{n+1/2} - (\mathrm{I} - \boldsymbol{\theta}^n)\nabla p^n \tag{2a}$$

$$\nabla \cdot \boldsymbol{v}^{n+1} = 0 \tag{2b}$$

Here Euler backward technique is employed for the updating of the time derivative term. $\boldsymbol{v}$ and p are the unknown discrete velocity vector and pressure, $(\nabla^2 \boldsymbol{v}/\mathrm{Re})^{n+1/2}$ and $-(\boldsymbol{v} \cdot \nabla \boldsymbol{v})^{n+1/2}$ are the temporal updating of the diffusion and convective terms respectively. The pressure term is updated by $\boldsymbol{\theta}^n \mathbf{G} p^{n+1} + (\mathrm{I} - \boldsymbol{\theta}^{\mathrm{n}})\mathbf{G} p^n$, and

$$\boldsymbol{\theta}^n = \begin{bmatrix} \theta_x^n & & \\ & \theta_y^n & \\ & & \theta_z^n \end{bmatrix}$$

is the coefficient matrix. θ_x, θ_y, θ_z $\epsilon[0,1]$ may depend on the grid size, time step, and even velocity. $\mathbf{I}$ is the unit identity matrix operator, and $\mathbf{G}$ is the discrete gradient operator. We have trapezoidal updating of the pressure term when $\boldsymbol{\theta}^n = 1/2\,\mathrm{I}$, and fully implicit updating when $\boldsymbol{\theta}^n = \mathrm{I}$. The solution of Eqs. (2*a*) and (2*b*) is not easy to obtain because it involves simultaneous solution for the velocity and pressure.

The fundamental concept of the SIMPLE method is to derive a pressure-correction equation by enforcing mass continuity over each cell. Using Patankar's notation [10], and supposing a fully implicit technique is employed to update both the convective and diffusion terms, we now have the following approximation of the convective and diffusion terms:

$$(\boldsymbol{\nu} \cdot \nabla \boldsymbol{\nu})^{n+1/2} - \left(\nabla^2 \boldsymbol{\nu}/\mathrm{Re}\right)^{n+1/2} = \boldsymbol{A}_P^n \boldsymbol{\nu}_P^{n+1} - \sum_M \boldsymbol{A}_M^n \boldsymbol{\nu}_M^{n+1} \tag{3}$$

A linear treatment of $\boldsymbol{\nu}^{n+1} \cdot \nabla \boldsymbol{\nu}^{n+1} \approx \boldsymbol{\nu}^n \cdot \nabla \boldsymbol{\nu}^{n+1}$ has been applied for the discretization of the convective term in Eq. (3).

$$\mathbf{A}_P^n = \begin{bmatrix} A_{PX}^n & & \\ & A_{PY}^n & \\ & & A_{PZ}^n \end{bmatrix} \quad \text{and} \quad \mathbf{A}_M^n = \begin{bmatrix} A_{MX}^n & & \\ & A_{MY}^n & \\ & & A_{MZ}^n \end{bmatrix}$$

are the coefficient matrices with $A_{PX}^n = \sum_M A_{MX}^n$, $A_{PY}^n = \sum_M A_{MY}^n$, $A_{PZ}^n = \sum_M A_{MZ}^n$. These coefficient terms contain the contribution of the convective and diffusion terms, and may depend on grid size and velocity, but are independent with time step. Index M is a grid identifier referring to all nodes surrounding the pole nodes that are involved in the formulation of the finite-difference representation of spatial fluxes. Subscript P denotes the pole node. As an example, when the central difference (CD) and first upwind difference (FUD) schemes are employed for the convective term in a uniform grid system, we have $\mathbf{A}_P = 6(\mathrm{Re}\,h^2)^{-1}\mathrm{I}$ and $\mathbf{A}_P = \left[(|u| + |v| + |w|)h^{-1} + 6\left(\mathrm{Re}\,h^2\right)^{-1}\right]\mathrm{I}$, respectively. u, v, w are the velocity variables corresponding to *x, y, z* coordinates, h is the grid size. Using the discretization formula of Eq. (3), a fully discretized momentum equation can be acquired based on Eq. (2*a*) with $\boldsymbol{\theta}^n = 0$ as

$$\left(\frac{1}{\Delta t}\mathbf{I} + \mathbf{A}_P^n\right)\hat{\boldsymbol{\nu}}_P = \frac{\boldsymbol{\nu}_P^n}{\Delta t} + \sum_M \mathbf{A}_M^n \hat{\boldsymbol{\nu}}_M - \mathbf{G}_P(p^n) \tag{4}$$

Here $\hat{\boldsymbol{\nu}}_P$ and $\hat{\boldsymbol{\nu}}_M$ are the intermediate velocities on pole node and neighbor nodes, respectively. Since the $\left(\boldsymbol{\nu}^{n+1}, p^{n+1}\right)$ satisfies the momentum and continuity equations of (2*a*) and (2*b*) with $\boldsymbol{\theta}^n = \mathrm{I}$, using the same updating technique and linear treatment as for Eq. (4), we have

$$\left(\frac{1}{\Delta t}\mathbf{I} + \mathbf{A}_P^n\right)\boldsymbol{\nu}_P^{n+1} = \frac{\boldsymbol{\nu}_P^n}{\Delta t} + \sum_M \mathbf{A}_M^n \boldsymbol{\nu}_M^{n+1} - \mathbf{G}_P\left(p^{n+1}\right) \tag{5}$$

By subtracting Eq. (4) from Eq. (5), a velocity difference equation can be acquired:

$$\left(\frac{1}{\Delta t}\mathbf{I} + \mathbf{A}_P^n\right)\left(\boldsymbol{\nu}_P^{n+1} - \hat{\boldsymbol{\nu}}_P\right) = \sum_M \mathbf{A}_M^n\left(\boldsymbol{\nu}_M^{n+1} - \hat{\boldsymbol{\nu}}_M\right) - \mathbf{G}_P\left(p^{n+1} - p^n\right) \tag{6}$$

By neglecting the underlined term in Eq. (6), a velocity-correction equation can be written as

$$\boldsymbol{\nu}_P^{n+1} = \hat{\boldsymbol{\nu}}_P - \Delta t\big(\mathbf{I} + \mathbf{A}_P^n \Delta t\big)^{-1}\mathbf{G}_P\big(p^{n+1} - p^n\big) \tag{7}$$

Since $\nabla \cdot v_P^{n+1} = 0$, the pressure correction can be acquired through the solution of the following Poisson equation of pressure difference:

$$\Delta t\,\mathbf{D}\Big[\big(\mathbf{I} + \mathbf{A}_P^n/\Delta t\big)^{-1}\mathbf{G}_P\big(p^{n+1} - p^n\big)\Big] = \mathbf{D}\hat{\boldsymbol{\nu}}_P \tag{8}$$

D is the discrete divergence operator, and

$$\big(\mathbf{I} + \mathbf{A}_P^n \Delta t\big)^{-1} = \begin{bmatrix} 1/\big(1 + \mathbf{A}_{PX}^n \Delta t\big) & & \\ & 1/\big(1 + \mathbf{A}_{PY}^n \Delta t\big) & \\ & & 1/\big(1 + \mathbf{A}_{PZ}^n \Delta t\big) \end{bmatrix}$$

Equations (4), (7), and (8) form the concise formulations of standard SIMPLE method [10, 11], repeated here for convenience's sake:

$$\left(\frac{1}{\Delta t}\mathbf{I} + \mathbf{A}_P^n\right)\hat{\boldsymbol{\nu}}_P = \frac{\boldsymbol{\nu}_P^n}{\Delta t} + \sum_M \mathbf{A}_M^n \hat{\boldsymbol{\nu}}_M - \mathbf{G}_P(p^n) \tag{9a}$$

$$\Delta t\,\mathbf{D}\Big[\big(\mathbf{I} + \mathbf{A}_P^n \Delta t\big)^{-1}\mathbf{G}_P\big(p^{n+1} - p^n\big)\Big] = \mathbf{D}\hat{\boldsymbol{\nu}}_P \tag{9b}$$

$$\boldsymbol{\nu}_P^{n+1} = \hat{\boldsymbol{\nu}}_P - \Delta t\big(\mathbf{I} + \mathbf{A}_P^n \Delta t\big)^{-1}\mathbf{G}_P\big(p^{n+1} - p^n\big) \tag{9c}$$

For the SIMPLE method, the predictor step will get the intermediate velocity by solving the momentum equation. The pressure correction from the Poisson equation (9*b*) will be utilized to correct the velocity by (9*c*).

Now we analyze the temporal accuracy of the standard SIMPLE method based on the above concise formulations of Eqs. (9). By employing $\boldsymbol{\theta}_P^n = \big(\mathbf{I} + \mathbf{A}_P^n \Delta t\big)^{-1}$ in Eq. (2*a*), we have

$$\frac{\boldsymbol{\nu}_P^{n+1} - \boldsymbol{\nu}_P^n}{\Delta t} + \boldsymbol{\theta}_P^n \mathbf{G}_P\big(p^{n+1}\big) = \left(\frac{\nabla^2 \boldsymbol{\nu}}{\mathrm{Re}}\right)_P^{n+1/2} - (\boldsymbol{\nu} \cdot \nabla \boldsymbol{\nu})_P^{n+1/2} - \big(\boldsymbol{I} - \boldsymbol{\theta}_P^n\big)\boldsymbol{G}_P(p^n) \tag{10a}$$

$$\nabla \cdot \boldsymbol{\nu}_P^{n+1} = 0 \tag{10b}$$

The SIMPLE method of Eqs. (9) is employed to approximate Eqs. (10). From Eq. (9c), the intermediate velocities $\hat{\boldsymbol{\nu}}_P$ and $\hat{\boldsymbol{\nu}}_M$ in the SIMPLE method can be obtained as

$$\hat{\boldsymbol{\nu}}_P = \boldsymbol{\nu}_P^{n+1} + \Delta t\big(\mathbf{I} + \mathbf{A}_P^n \Delta t\big)^{-1}\mathbf{G}_P\big(p^{n+1} - p^n\big) = \boldsymbol{\nu}_P^{n+1} + \Delta t\,\boldsymbol{\theta}_P^n \mathbf{G}_P\big(p^{n+1} - p^n\big) \tag{11}$$

$$\hat{\boldsymbol{\nu}}_M = \boldsymbol{\nu}_M^{n+1} + \Delta t\big[\mathbf{I} + \big(\mathbf{A}_P^n\big)_M \Delta t\big]^{-1}\mathbf{G}_M\big(p^{n+1} - p^n\big) = \boldsymbol{\nu}_M^{n+1} + \Delta t\,\boldsymbol{\theta}_M^n \mathbf{G}_M\big(p^{n+1} - p^n\big) \tag{12}$$

$\left(\mathbf{A}_P^n\right)_M$ is the discretization coefficient at the pole point, but the pole point is M, not P. $\mathbf{A}_P^n$ is the discretization coefficient at the pole point of P. Substitute Eqs. (11) and (12) into the predictor-step equation of Eq. (9a), we have

$$\left(\frac{1}{\Delta t}\mathbf{I}+\mathbf{A}_P^n\right)\left[\boldsymbol{v}_P^{n+1}+\Delta t\,\boldsymbol{\theta}_P^n\mathbf{G}_P\left(p^{n+1}-p^n\right)\right]=\frac{\boldsymbol{v}_P^n}{\Delta t}$$
$$+\sum_M \mathbf{A}_M^n\left[\boldsymbol{v}_M^{n+1}+\Delta t\,\boldsymbol{\theta}_M^n\mathbf{G}_M\left(p^{n+1}-p^n\right)\right]-\mathbf{G}_P(p^n) \tag{13}$$

Equation (13) is the approximating equation of Eq. (10a). Reducing Eq. (10a) from its approximating equation (13), we can get the error term as

$$\mathbf{Err}=\left\{\left(\mathbf{A}_P^n\boldsymbol{v}_P^{n+1}-\sum_M \mathbf{A}_M^n\boldsymbol{v}_M^{n+1}\right)-\left[\left(\frac{\nabla^2\boldsymbol{v}}{\mathrm{Re}}\right)_P^{n+1/2}-(\boldsymbol{v}\cdot\nabla\boldsymbol{v})_P^{n+1/2}\right]\right\}$$
$$+\Delta t\left[\mathbf{A}_P^n\boldsymbol{\theta}_P^n\mathbf{G}_P\left(p^{n+1}-p^n\right)-\sum_M \mathbf{A}_M^n\boldsymbol{\theta}_M^n\mathbf{G}_M\left(p^{n+1}-p^n\right)\right] \tag{14}$$

The error term includes two parts. One is due to the special discretization of convective and diffusion terms,

$$\mathbf{Err\,U}=\left(\mathbf{A}_P^n\boldsymbol{v}_P^{n+1}-\sum_M \mathbf{A}_M^n\boldsymbol{v}_M^{n+1}\right)-\left[\left(\frac{\nabla^2\boldsymbol{v}}{\mathrm{Re}}\right)_P^{n+1/2}-(\boldsymbol{v}\cdot\nabla\boldsymbol{v})_P^{n+1/2}\right] \tag{15}$$

And the other is due to the SIMPLE method itself, which can be expressed as

$$\begin{aligned}\mathbf{Err\,P}&=\Delta t\left[\mathbf{A}_P^n\boldsymbol{\theta}_P^n\mathbf{G}_P\left(p^{n+1}-p^n\right)-\sum_M \mathbf{A}_M^n\boldsymbol{\theta}_M^n\mathbf{G}_M\left(p^{n+1}-p^n\right)\right]\\&=(\Delta t)^2\left[\mathbf{A}_P^n\boldsymbol{\theta}_P^n\mathbf{G}_P\left(\delta p^{n+1}\right)-\sum_M \mathbf{A}_M^n\boldsymbol{\theta}_M^n\mathbf{G}_M\left(\Delta p^{n+1}\right)\right]\end{aligned} \tag{16}$$

Here $\delta p^{n+1}=\left(p^{n+1}-p^n\right)/\Delta t$. Since $\left|\boldsymbol{\theta}_P^n\right|=\left|\left(\mathrm{I}+\mathbf{A}_P^n\,\Delta t\right)^{-1}\right|\leq 1$, and $\left|\boldsymbol{\theta}_M^n\right|=|(\mathrm{I}+\left(\mathbf{A}_P^n\right)_M\Delta t)^{-1}|\leq 1$, we have

$$|\mathbf{Err\,P}|=\left|(\Delta t)^2\left[\mathbf{A}_P^n\boldsymbol{\theta}_P^n\mathbf{G}_P\left(\delta p^{n+1}\right)-\sum_M \mathbf{A}_M^n\boldsymbol{\theta}_M^n\mathbf{G}_M\left(\delta p^{n+1}\right)\right]\right|$$
$$\leq(\Delta t)^2\left[\left|\mathbf{A}_P^n\right|\left|\mathbf{G}_P\left(\delta p^{n+1}\right)\right|+\sum_M\left|\mathbf{A}_M^n\right|\left|\mathbf{G}_M\left(\delta p^{n+1}\right)\right|\right] \tag{17}$$

Since coefficient matrices are independent of time step, we have $|\mathbf{Err\,P}|=\mathbf{O}[(\Delta t)^2]$. $|\mathbf{Err\,U}|=\mathbf{O}(\Delta t)$ when a first-order fully explicit or fully implicit Euler scheme is employed to update the temporal term. $|\mathbf{Err\,U}|$ can also be $\mathbf{O}[(\Delta t)^2]$ if we employ a second-order temporal accuracy technique to update the temporal term or the convective and diffusion terms, such as the second-order semi-implicit

Crank-Nicholson technique. Also, when a third-order temporal updating technique is employed, we have $|\mathbf{Err\,U}| = \mathbf{O}(\Delta t)^3$. No matter what kind of temporal accuracy scheme is employed for the updating of temporal term, $|\mathbf{Err\,P}| = \mathbf{O}[(\Delta t)^2]$. Now we say that SIMPLE method has second-order temporal accuracy because $|\mathbf{Err\,P}| = \mathbf{O}[(\Delta t)^2]$ from Eq. (17). In other words, the SIMPLE method can obtain second-order-temporal-accuracy results of an unsteady problem if the temporal term is updated using a second-order technique. However, the SIMPLE method has at most second-order temporal accuracy even if a third-order-temporal-accuracy updating scheme is employed to update the temporal term. The first projection method of Chorin [2] has only first-order temporal accuracy because the result is first-order even if a second-order technique is used to update both the diffusion and convective terms. The projection methods of Bell, Collela, and Glaz [3], and Kim and Moin [5] are second-order accuracy since they can obtain second-order results when the temporal term is second-order updated. This section's accuracy analysis shows that the SIMPLE method can demonstrate the same accuracy as second-order projection methods.

3. SIMPLEC AND A FOUR-STEP SIMPLE METHOD FOR UNSTEADY FLOWS

We have shown that the SIMPLE method can demonstrate second-order accuracy as a second-order projection method. In fact, a classical second-order projection method can be acquired in the following way. In Eq. (6), instead of neglecting the underlined term, we approximate $\left(\boldsymbol{v}_M^{n+1} - \hat{\boldsymbol{v}}_M\right)$ by $\left(\boldsymbol{v}_P^{n+1} - \hat{\boldsymbol{v}}_P\right)$, and considering the fact that $\left[\mathbf{A}_P^n\right] = \sum_M \left[\mathbf{A}_M^n\right]$, we have the following velocity-correction equation and pressure difference Poisson equation:

$$\boldsymbol{v}_P^{n+1} = \hat{\boldsymbol{v}}_P - \Delta t\, \mathbf{G}_P\left(p^{n+1} - p^n\right) \tag{18}$$

Since $\nabla \cdot \boldsymbol{v}_P^{n+1} = 0$, we can obtain the pressure difference Poisson equation as

$$\Delta t\, \mathbf{D}\left[\mathbf{G}_P\left(p^{n+1} - p^n\right)\right] = \mathbf{D}\hat{\boldsymbol{v}}_P \tag{19}$$

Equations (4), (18), and (19) form the traditional second-order three-step projection method:

$$\left(\frac{1}{\Delta t}\mathbf{I} + \mathbf{A}_P^n\right)\hat{\boldsymbol{v}}_P = \frac{\boldsymbol{v}_P^n}{\Delta t} + \sum_M \mathbf{A}_M^n \hat{\boldsymbol{v}}_M - \mathbf{G}_P(p^n) \tag{20a}$$

$$\Delta t\, \mathbf{D}\left[\mathbf{G}_P\left(p^{n+1} - p^n\right)\right] = \mathbf{D}\hat{\boldsymbol{v}}_P \tag{20b}$$

$$\boldsymbol{v}_P^{n+1} = \hat{\boldsymbol{v}}_P - \Delta t\, \mathbf{G}_P\left(p^{n+1} - p^n\right) \tag{20c}$$

This construction idea for the traditional projection method of Eqs. (20) has been employed in [12] to form the SIMPLEC method.

In SIMPLE-type methods for steady flows, usually the underrelaxation method is employed to solve both the momentum equation and the pressure Poisson equation. Employing an underrelaxation technique for the predictor step of Eq. (9*a*) in the standard SIMPLE method, we have

$$\left[(\Delta t)^{-1}\mathbf{I} + \boldsymbol{\lambda}^{-1}\mathbf{A}_P^n\right]\hat{\boldsymbol{v}}_P = (\Delta t)^{-1}\boldsymbol{v}_P^n + \left(\boldsymbol{\lambda}^{-1} - \mathbf{I}\right)\mathbf{A}_P^n\hat{\boldsymbol{v}}_P^n + \sum_M \mathbf{A}_M^n\hat{\boldsymbol{v}}_M - \mathbf{G}_P(p^n) \quad (21a)$$

$$\Delta t\,\mathbf{D}\left[\left(\mathbf{I} + \Delta t\lambda^{-1}\mathbf{A}_P^n\right)^{-1}\mathbf{G}_P\left(p^{n+1} - p^n\right)\right] = \mathbf{D}\hat{\mathrm{v}}_P \quad (21b)$$

$$\mathrm{v}_P^{n+1} = \hat{\mathrm{v}}_P - \Delta t\left(\mathbf{I} + \Delta t\,\boldsymbol{\lambda}^{-1}\mathbf{A}_P^n\right)^{-1}\mathbf{G}_P\left(p^{n+1} - p^n\right) \quad (21c)$$

Here

$$\boldsymbol{\lambda} = \begin{bmatrix} \lambda_x & & \\ & \lambda_y & \\ & & \lambda_z \end{bmatrix}$$

is the underrelaxation factor matrix. $\lambda_x, \lambda_y, \lambda_z$ are the underrelaxation factors for the momentum equations in x, y, z coordinates, respectively, which can be used to improve the solution stability of the momentum equation (21a). In [20], the solution stability criterion was acquired for one- and multidimensional steady convective diffusion equations using the underrelaxation technique.

One important point needs to be noted when using an underrelaxation method for the calculation of unsteady flows. A new variable velocity, $\hat{\boldsymbol{v}}_P^n$, in Eq. (21a), is introduced to improve the numerical stability of the momentum equation at the predictor step, which is not equal to $\boldsymbol{v}_P^n$ at the n time level and is also different with $\hat{\boldsymbol{v}}_P$. $\boldsymbol{v}_P^n$ represents the velocity value at the previous time step, which is divergence-free. $\hat{\boldsymbol{v}}_P$ is the intermediate velocity used to obtain the pressure difference by solving Eq. (21b). This intermediate velocity is further used to obtain the corrected velocity using Eq. (21c). $\hat{\boldsymbol{v}}_P^n$ can be called subintermediate velocity, which is renewed by $\hat{\boldsymbol{v}}_P$ during the subiteration for the acquisition of intermediate velocity of $\hat{\boldsymbol{v}}_P$ from the predictor step. When the convergent solution of momentum at the predictor step of Eq. (21a) is acquired, we should have $\hat{\boldsymbol{v}}_P = \hat{\boldsymbol{v}}_P^n$. However, before we obtain the convergent solution, the two velocities are different. For unsteady flows, both the subintermediate velocity $\hat{\boldsymbol{v}}_P^n$ and the intermediate velocity $\hat{\boldsymbol{v}}_P$ are not equal to $\boldsymbol{v}_P^n$, and are not divergence-free. For steady flow, when convergent solution is acquired, the three velocities should be the same.

Approximating $\left(\mathrm{v}_M^{n+1} - \hat{\boldsymbol{v}}_M\right)$ using $\left(\mathrm{v}_P^{n+1} - \hat{\boldsymbol{v}}_P\right)$ in a velocity difference equation, which is similar to Eq. (6), the SIMPLEC method can be acquired based on the above formula as

$$\left[(\Delta t)^{-1}\mathbf{I} + \boldsymbol{\lambda}^{-1}\mathbf{A}_P^n\right]\hat{\boldsymbol{v}}_P = (\Delta t)^{-1}\boldsymbol{v}_P^n + \left(\boldsymbol{\lambda}^{-1} - \mathbf{I}\right)\mathbf{A}_P^n\hat{\boldsymbol{v}}_P^n + \sum_M \mathbf{A}_M^n\hat{\boldsymbol{v}}_M - \mathbf{G}_P(p^n) \quad (22a)$$

$$\Delta t\,\mathbf{D}\left\{\left[\mathbf{I} + \Delta t\left(\boldsymbol{\lambda}^{-1} - \mathbf{I}\right)\mathbf{A}_P^n\right]^{-1}\mathbf{G}_P\left(p^{n+1} - p^n\right)\right\} = \mathbf{D}\hat{\boldsymbol{v}}_P \quad (22b)$$

$$\boldsymbol{v}_P^{n+1} = \hat{\boldsymbol{v}}_P - \Delta t\left[\mathbf{I} + \Delta t\left(\boldsymbol{\lambda}^{-1} - \mathbf{I}\right)\mathbf{A}_P^n\right]^{-1}\mathbf{G}_P\left(p^{n+1} - p^n\right) \quad (22c)$$

When $\boldsymbol{\lambda} = \mathbf{I}$, the SIMPLEC method of Eqs. (22) is in fact a traditional second-order three-step projection method in Eqs. (20). It is not difficult to conclude that the above methods of Eqs. (21) and Eqs. (22) are also temporally second-order, approximating to Eqs. (10) with $\boldsymbol{\theta}_P^n = \left(\mathbf{I} + \Delta t\,\boldsymbol{\lambda}^{-1}\mathbf{A}_P^n\right)^{-1}$ and $\boldsymbol{\theta}_P^n = \left[\mathbf{I} + \Delta t\left(\boldsymbol{\lambda}^{-1} - \mathbf{I}\right]\mathbf{A}_P^n\right]^{-1}$, respectively. This also means that the underrelaxation factor does not affect the temporal accuracy of the SIMPLE-type method for unsteady flows.

In the above equations, Δt is a time step for the calculation of unsteady flows. This term can also be regarded as an underrelaxation term for steady-flow calculation. For steady flow, we do not need to stress the difference between the subintermediate velocity $\hat{\boldsymbol{v}}_P^n$ and the intermediate velocity $\boldsymbol{v}_P^n$. Consider the simulation of a steady flow using the standard SIMPLE method of Eqs. (9), which can be written as

$$\left(\boldsymbol{\lambda}^{-1}\mathbf{A}_P^n\right)\hat{\boldsymbol{v}}_P = \left(\boldsymbol{\lambda}^{-1} - \mathbf{I}\right)\mathbf{A}_P^n\boldsymbol{v}_P^n + \sum_M \mathbf{A}_M^n\hat{\boldsymbol{v}}_M - \mathbf{G}_P(p^n) \tag{23a}$$

$$\mathbf{D}\left[\boldsymbol{\lambda}\left(\mathbf{A}_P^n\right)^{-1}\mathbf{G}_P\left(p^{n+1} - p^n\right)\right] = \mathbf{D}\hat{\boldsymbol{v}}_P \tag{23b}$$

$$\boldsymbol{v}_P^{n+1} = \hat{\boldsymbol{v}}_P - \boldsymbol{\lambda}\left(\mathbf{A}_P^n\right)^{-1}\mathbf{G}_P\left(p^{n+1} - p^n\right) \tag{23c}$$

with

$$\lambda_X = \frac{A_{PX}^n\,\Delta t}{A_{PX}^n\,\Delta t + 1} \qquad \lambda_Y = \frac{A_{PY}^n\,\Delta t}{A_{PY}^n\,\Delta t + 1} \qquad \lambda_Z = \frac{A_{PZ}^n\,\Delta t}{A_{PZ}^n\,\Delta t + 1}$$

Based on the relationship between Eqs. (9) and (23), apparently the temporal term can play a role of underrelaxation to improve the solution stability of the discretized momentum equation.

The three-step SIMPLE-type methods of Eqs. (9), (20), (21), and (22) can also be written as a four-step fractional step formula. For example, in the standard three-step SIMPLE method of Eqs. (9), Eq. (9c) can be rewritten as

$$\boldsymbol{v}_P^{n+1} = \left[\hat{\boldsymbol{v}}_P + \Delta t\left(\mathbf{I} + \mathbf{A}_P^n\,\Delta t\right)^{-1}\mathbf{G}_P(p^n)\right] - \Delta t\left(\mathbf{I} + \mathbf{A}_P^n\,\Delta t\right)^{-1}\mathbf{G}_P\left(p^{n+1}\right) \tag{24}$$

The term in the left brackets owns the velocity scale, which can be defined as a second intermediate velocity $\tilde{\boldsymbol{v}}_P = \hat{\boldsymbol{v}}_P + \Delta t\left(\mathbf{I} + \mathbf{A}_P^n\,\Delta t\right)^{-1}\mathbf{G}_P(p^n)$. Now we have the four-step standard SIMPLE method as

$$\left(\frac{1}{\Delta t}\mathbf{I} + \mathbf{A}_P^n\right)\hat{\boldsymbol{v}}_P = \frac{\mathbf{v}_P^n}{\Delta t} + \sum_M \mathbf{A}_M^n\hat{\boldsymbol{v}}_M - \mathbf{G}_P(p^n) \tag{25a}$$

$$\tilde{\boldsymbol{v}}_P = \hat{\boldsymbol{v}}_P + \Delta t\left(\mathbf{I} + \mathbf{A}_P^n\,\Delta t\right)^{-1}\mathbf{G}_P(p^n) \tag{25b}$$

$$\Delta t\,\mathbf{D}\left[\left(\mathbf{I} + \mathbf{A}_P^n\,\Delta t\right)^{-1}\mathbf{G}_P\left(p^{n+1}\right)\right] = \mathbf{D}\tilde{\boldsymbol{v}}_P \tag{25c}$$

$$\boldsymbol{v}_P^{n+1} = \tilde{\boldsymbol{v}}_P - \Delta t\left(\mathbf{I} + \mathbf{A}_P^n\,\Delta t\right)^{-1}\mathbf{G}_P\left(p^{n+1}\right) \tag{25d}$$

The solution procedure of the four-step SIMPLE method (25) can be summarized as follows. The first intermediate velocity $\hat{\boldsymbol{v}}_P$ is obtained by solving the momentum equation of Eq. (25a) at the first predictor step. Then the second intermediate velocity $\tilde{\boldsymbol{v}}_P$ is acquired by Eq. (25b) at the second predictor step. This second intermediate velocity is not divergence-free, and is used to get the pressure term at the new time level by solving the pressure Poisson equation (25*c*). The gradient of the pressure value from (25*c*) is further used to get the corrected velocity at the new time level. This velocity should be divergence-free if the convergent pressure value is obtained from Eq. (25*c*). It has second-order-temporal-accuracy as the three-step SIMPLE method, if the convective and diffusion terms are updated using second-order-temporal-accuracy schemes. Based on this procedure, we can see the difference between three-step and four-step SIMPLE methods. For the three-step SIMPLE method, a Poisson equation is solved to get the pressure difference, which is further used to correct the velocity. For the four-step SIMPLE method, a Poisson equation is solved to get the pressure itself, which is used to correct velocity. The four-step projection method was first presented in [4], and has been analyzed in detail by Ni et al. [6, 8]. The CLEAR algorithm [21] can be considered as a four-step SIMPLE method for steady flows, and the efficiency of CLEAR for steady flows is shown in [22].

4. HIGH-ORDER SIMPLE METHOD FOR UNSTEADY FLOWS

In Sections 2 and 3, we concluded that SIMPLE-type methods have second-order temporal accuracy for unsteady flows if second-order temporal updating techniques are employed for the temporal term or both the convective and diffusion terms in momentum equations at the predictor step. However, due to the linear treatment of the convective terms in SIMPLE-type methods, the coefficients $\mathbf{A}_P^n$ and $\mathbf{A}_M^n$ are dependent on the velocity for any implicit updating of the convective term. Iteration is needed to obtain the intermediate velocity or the velocity at the next time step. Here we consider two iteration algorithms for the calculations of unsteady flows using SIMPLE-type methods. Algorithm A is an iteration method, in which the iteration needs to be done between the predictor and corrected steps. At every iteration step, the pressure Poisson equation needs to be solved. Algorithm B is not an iteration method, in which the convergent intermediate velocity is acquired from the predictor step. This convergent intermediate velocity is not divergence-free and is used to get pressure difference. The divergence-free velocity is obtained using the intermediate velocity and pressure difference. In algorithm B, the pressure Poisson equation needs to be solved only once for a time step. Since the solution of the pressure Poisson equation costs more time than the solution of the momentum equation, algorithm B is more economical than algorithm A.

Based on the standard SIMPLE method of Eqs. (9), we have the SIMPLE A algorithm as

$$\left\{\begin{array}{l}\mathbf{Do\,i} = 1, \mathbf{L} \\ \left\{\begin{array}{l}\mathbf{A}_P^{n+(i-1)/L} = \mathbf{A}_P\left(\boldsymbol{v}^{n+(i-1)/L}\right) \qquad \mathbf{A}_M^{n+(i-1)/L} = \mathbf{A}_M\left(\boldsymbol{v}^{n+(i-1)/L}\right) \\ \left(\frac{1}{\Delta t}\mathbf{I} + \mathbf{A}_P^{n+(i-1)/L}\right)\hat{\boldsymbol{v}}_P^{n+(i/L)} = \frac{\boldsymbol{v}_P^n}{\Delta t} + \sum_M \mathbf{A}_M^{n+(i-1)/L}\hat{\boldsymbol{v}}_M^{n+(i/L)} - \mathbf{G}p^{n+(i-1)/L} \\ \Delta t\,\mathbf{D}\left[\left(\mathbf{I} + \mathbf{A}_P^{n+(i-1)/L}\Delta t\right)^{-1}\mathbf{G}\left(p^{n+(i/L)} - p^{n+(i-1)/L}\right)\right] = \mathbf{D}\hat{\boldsymbol{v}}_P^{n+(i/L)} \\ \boldsymbol{v}_P^{n+(i/L)} = \hat{\boldsymbol{v}}_P^{n+(i/L)} - \Delta t\left(\mathbf{I} + \mathbf{A}_P^{n+(i-1)/L}\Delta t\right)^{-1}\mathbf{G}\left(p^{n+(i/L)} - p^{n+(i-1)/L}\right)\end{array}\right. \\ \mathbf{EndDo}\end{array}\right. \tag{26}$$

and the SIMPLE B algorithm as

$$\begin{cases} \hat{\boldsymbol{v}}^n = \boldsymbol{v}^n \\ \textbf{Do i} = 1, \textbf{L} \\ \begin{cases} \hat{\mathbf{A}}_P^{n+(i-1)/L} = \mathbf{A}_P\left(\hat{\boldsymbol{v}}^{n+(i-1)/L}\right) \qquad \hat{\mathbf{A}}_M^{n+(i-1)/L} = \mathbf{A}_M\left(\hat{\boldsymbol{v}}^{n+(i-1)/L}\right) \\ \left(\frac{1}{\Delta t}\mathbf{I} + \hat{\mathbf{A}}_P^{n+(i-1)/L}\right)\hat{\boldsymbol{v}}_P^{n+(i/L)} = \frac{\boldsymbol{v}_P^n}{\Delta t} + \sum_M \hat{\mathbf{A}}_M^{n+(i-1)/L}\hat{\boldsymbol{v}}_M^{n+(i/L)} - \mathbf{G}p^n \end{cases} \\ \textbf{EndDo} \\ \hat{\mathbf{A}}_P^{n+1} = \mathbf{A}_P\left(\hat{\boldsymbol{v}}^{n+1}\right) \qquad \hat{\mathbf{A}}_M^{n+1} = \mathbf{A}_M\left(\hat{\boldsymbol{v}}^{n+1}\right) \\ \Delta t\,\mathbf{D}\left[\left(\mathbf{I} + \hat{\mathbf{A}}_P^{n+1}\,\Delta t\right)^{-1}\mathbf{G}\left(p^{n+1} - p^n\right)\right] = \mathrm{D}\hat{\boldsymbol{v}}_P^{n+1} \\ \boldsymbol{v}_P^{n+1} = \hat{\boldsymbol{v}}_P^{n+1} - \Delta t\left(\mathbf{I} + \hat{\mathbf{A}}_P^{n+1}\,\Delta t\right)^{-1}\mathbf{G}\left(p^{n+1} - p^n\right) \end{cases} \tag{27}$$

In algorithm (26), L steps of iterations are needed to get the velocity and pressure $\left(\boldsymbol{v}^{n+1}, p^{n+1}\right)$ from $(\boldsymbol{v}^n, p^n)$. The pressure difference Poisson equation needs to be solved at every iteration step of this algorithm. Algorithm (26) is not economical. However, most applications of the SIMPLE-type method for unsteady flows employ this kind of algorithm. For SIMPLE B of algorithm (27), L steps of iterations are done only for momentum equations to obtain the intermediate velocity $\hat{\boldsymbol{v}}_P^{n+1}$, which is then used to acquire the pressure and velocity corrections. Apparently, algorithm (27) will greatly save computational time compared with algorithm (26). According to the analysis in Sections 2 and 3, both algorithms should be able to obtain second-order-temporal-accuracy results, supposing that second-order updating techniques are employed for both the convective and diffusion terms. However, in (26) and (27), the first-order fully implicit technique is employed, so these have only first-order temporal accuracy. In the following, three versions of high-order SIMPLE algorithms for unsteady flows will be presented by employing high-order updating techniques for the convective and diffusion terms.

(I) Crank-Nicholson Updating for Both Convective and Diffusion Terms

The Crank-Nicholson scheme is a second-order semi-implicit technique. It is often employed to update the diffusion term for classical projection methods. Now, we employ it for the updating of convective and diffusion terms in the standard SIMPLE method of Eqs. (9). We have $\left(\nabla^2\boldsymbol{v}/\mathrm{Re}\right)^{n+1/2} = \frac{1}{2}\left[\left(\nabla^2\boldsymbol{v}/\mathrm{Re}\right)^{n+1/2} + \left(\nabla^2\boldsymbol{v}/\mathrm{Re}\right)^{n+1/2}\right]$ and $(\boldsymbol{v}\cdot\nabla\boldsymbol{v})^{n+1/2} = \frac{1}{2}\left[(\boldsymbol{v}\cdot\nabla\boldsymbol{v})^{n+1} + (\boldsymbol{v}\cdot\nabla\boldsymbol{v})^n\right]$. Using linear treatment of the convective term, $(\boldsymbol{v}\cdot\nabla\boldsymbol{v})^{n+1/2} = \frac{1}{2}\left(\boldsymbol{v}^n\cdot\nabla\boldsymbol{v}^{n+1} + \boldsymbol{v}^n\cdot\nabla\boldsymbol{v}^n\right)$, we have

Algorithm A:

$$\begin{cases} \mathbf{A}_P^n = \mathbf{A}_P(\mathbf{v}^n) \qquad \mathbf{A}_M^n = \mathbf{A}_M(\mathbf{v}^n) \\ \mathbf{Do\ i} = 1, \mathbf{L} \\ \begin{cases} \mathbf{A}_P^{n+(i-1)/L} = \mathbf{A}_P\left(\mathbf{v}^{n+(i-1)/L}\right) \qquad \mathbf{A}_M^{n+(i-1)/L} = \mathbf{A}_M\left(\mathbf{v}^{n+(i-1)/L}\right) \\ \left(\frac{1}{\Delta t}\mathbf{I} + \frac{1}{2}\mathbf{A}_P^{n+(i-1)/L}\right)\hat{\mathbf{v}}_P^{n+(i/L)} = \left[\left(\frac{\mathbf{I}}{\Delta t} - \frac{1}{2}\mathbf{A}_P^n\right)\mathbf{v}_P^n + \frac{1}{2}\sum_M \mathbf{A}_M^n \mathbf{v}_M^n\right] \\ \qquad\qquad + \frac{1}{2}\sum_M \mathbf{A}_M^{n+(i-1)/L}\hat{\mathbf{v}}_M^{n+(i/L)} - \mathbf{G}p^{n+(i-1)/L} \\ \Delta t\,\mathbf{D}\left[\left(\mathbf{I} + \frac{\Delta t}{2}\mathbf{A}_P^{n+(i-1)/L}\right)^{-1}\mathbf{G}\left(p^{n+(i/L)} - p^{n+(i-1)/L}\right)\right] = \mathbf{D}\hat{\mathbf{v}}_P^{n+(i/L)} \\ \mathbf{v}_P^{n+(i/L)} = \hat{\mathbf{v}}_P^{n+(i/L)} - \Delta t\left(\mathbf{I} + \frac{\Delta t}{2}\mathbf{A}_P^{n+(i-1)/L}\right)^{-1}\mathbf{G}\left(p^{n+(i/L)} - p^{n+(i-1)/L}\right) \end{cases} \\ \mathbf{EndDo} \end{cases} \tag{28}$$

Algorithm B:

$$\begin{cases} \hat{\mathbf{v}}^n = \mathbf{v}^n \\ \mathbf{Do\ i} = 1, \mathbf{L} \\ \begin{cases} \hat{\mathbf{A}}_P^{n+(i-1)/L} = \mathbf{A}_P\left(\hat{\mathbf{v}}^{n+(i-1)/L}\right) \qquad \hat{\mathbf{A}}_M^{n+(i-1)/L} = \mathbf{A}_M\left(\hat{\mathbf{v}}^{n+(i-1)/L}\right) \\ \left(\frac{1}{\Delta t}\mathbf{I} + \frac{1}{2}\hat{\mathbf{A}}_P^{n+(i-1)/L}\right)\hat{\mathbf{v}}_P^{n+(i/L)} = \left\{\left(\frac{\mathbf{I}}{\Delta t} - \frac{1}{2}\mathbf{A}_P^n\right)\mathbf{v}_P^n + \frac{1}{2}\sum_M \mathbf{A}_M^n \mathbf{v}_M^n\right\} \\ \qquad\qquad + \frac{1}{2}\sum_M \hat{\mathbf{A}}_M^{n+(i-1)/L}\hat{\mathbf{v}}_M^{n+(i/L)} - \mathbf{G}p^n \end{cases} \\ \mathbf{EndDo} \\ \hat{\mathbf{A}}_P^{n+1} = \mathbf{A}_P\left(\hat{\mathbf{v}}^{n+1}\right) \qquad \hat{\mathbf{A}}_M^{n+1} = \mathbf{A}_M\left(\hat{\mathbf{v}}^{n+1}\right) \\ \Delta t\,\mathbf{D}\left[\left(\mathbf{I} + \frac{\Delta t}{2}\hat{\mathbf{A}}_P^{n+1}\right)^{-1}\mathbf{G}\left(p^{n+1} - p^n\right)\right] = \mathbf{D}\hat{\mathbf{v}}_P^{n+1} \\ \mathbf{v}_P^{n+1} = \hat{\mathbf{v}}_P^{n+1} - \Delta t\left(\mathbf{I} + \frac{\Delta t}{2}\hat{\mathbf{A}}_P^{n+1}\right)^{-1}\mathbf{G}\left(p^{n+1} - p^n\right) \end{cases} \tag{29}$$

(II) Crank-Nicholson Updating for Diffusion Term, and Explicit Updating for Convective Term

The Crank-Nicholson scheme can be employed to update the diffusion term to improve stability, while a high-order explicit technique, such as the Adams-Bashforth scheme, can be employed to update the convective term for convenience. Hence, we have $(\mathbf{v} \cdot \nabla \mathbf{v})^{n+1/2} = N\left(\mathbf{v}^n, \mathbf{v}^{n-1}\right)$ and $\left(\nabla^2 \mathbf{v}/\mathrm{Re}\right)^{n+1/2} = \frac{1}{2}\left[\left(\nabla^2 \mathbf{v}/\mathrm{Re}\right)^{n+1} +\right.$

$(\nabla^2 \boldsymbol{v}/\mathrm{Re})^n$]. Since explicit techniques are employed to update the convective term, that means $\mathbf{A}_P$ and $\mathbf{A}_M$ include only the contributions from the discretization of the diffusion term, and they depend only on the grid size. Hence, we have Algorithm A as

$$\begin{cases} \textbf{Calculate } \mathbf{A}_P, \mathbf{A}_M \\ \textbf{Do i} = 1, \mathbf{L} \\ \begin{cases} \left(\frac{1}{\Delta t}\mathbf{I} + \frac{1}{2}\mathbf{A}_P\right)\hat{\boldsymbol{v}}_P^{n+(i/L)} = \left[\left(\frac{\mathbf{I}}{\Delta t} - \frac{1}{2}\mathbf{A}_P\right)\boldsymbol{v}_P^n + \frac{1}{2}\sum_M \mathbf{A}_M \boldsymbol{v}_M^n + N\left(\boldsymbol{v}^n, \boldsymbol{v}^{n-1}\right)\right] \\ \qquad\qquad + \frac{1}{2}\sum_M \mathbf{A}_M \hat{\boldsymbol{v}}_M^{n+(i/L)} - \mathbf{G}p^{n+(i-1)/L} \\ \Delta t\, \mathbf{D}\left[\left(\mathbf{I} + \frac{\Delta t}{2}\mathbf{A}_P\right)^{-1}\mathbf{G}\left(p^{n+(i/L)} - p^{n+(i-1)/L}\right)\right] = \mathbf{D}\hat{\boldsymbol{v}}_P^{n+(i/L)} \\ \boldsymbol{v}_P^{n+(i/L)} = \hat{\boldsymbol{v}}_P^{n+(i/L)} - \Delta t\left(\mathbf{I} + \frac{\Delta t}{2}\mathbf{A}_P\right)^{-1}\mathbf{G}\left(p^{n+(i/L)} - p^{n+(i-1)/L}\right) \end{cases} \\ \textbf{EndDo} \end{cases} \tag{30}$$

Also, algorithm B can be acquired for this updating:

$$\begin{cases} \textbf{Calculate } \mathbf{A}_P, \mathbf{A}_M \\ \textbf{Do i} = 1, \mathbf{L} \\ \begin{cases} \left(\frac{1}{\Delta t}\mathbf{I} + \frac{1}{2}\mathbf{A}_P\right)\hat{\boldsymbol{v}}_P^{n+(i/L)} = \frac{1}{2}\sum_M \mathbf{A}_M \hat{\boldsymbol{v}}_M^{n+(i/L)} - \mathbf{G}p^n \\ \qquad\qquad + \left[\left(\frac{\mathbf{I}}{\Delta t} - \frac{1}{2}\mathbf{A}_P\right)\boldsymbol{v}_P^n + \frac{1}{2}\sum_M \mathbf{A}_M \boldsymbol{v}_M^n + N\left(\boldsymbol{v}^n, \boldsymbol{v}^{n-1}\right)\right] \end{cases} \\ \textbf{EndDo} \\ \Delta t\, \mathbf{D}\left[\left(\mathbf{I} + \frac{\Delta t}{2}\mathbf{A}_P\right)^{-1}\mathbf{G}\left(p^{n+1} - p^n\right)\right] = \mathbf{D}\hat{\boldsymbol{v}}_P^{n+1} \\ \boldsymbol{v}_P^{n+1} = \hat{\boldsymbol{v}}_P^{n+1} - \Delta t\left(\mathbf{I} + \frac{\Delta t}{2}\mathbf{A}_P\right)^{-1}\mathbf{G}\left(p^{n+1} - p^n\right) \end{cases} \tag{31}$$

Since $\mathbf{A}_P^n$ and $\mathbf{A}_M^n$ depend only on grid size, we can choose $L = 1$ in the above algorithms (30) and (31) if a direct numerical method, such as the tri-diagonal matrix algorithm (TDMA) for a spatial three-point discretization scheme, can be employed for the solution of the momentum equation. For this case, the above two algorithms are really identical. This will greatly save computational time. Although, the Adams-Bashforth scheme is not very stable, the three-stage Runge-Kutta technique employed in [6–8] can be used to improve the stability for the above algorithms.

(III) Second-Order Temporal Discretization for the Temporal Term in the Momentum Equations

Now we present the third second-order SIMPLE method, in which the temporal term is updated using a second-order temporal updating technique as

$$\frac{\partial \boldsymbol{v}}{\partial t} = \frac{1}{\Delta t}\left(\frac{3}{2}\boldsymbol{v}_P^{n+1} - 2\boldsymbol{v}_P^n + \boldsymbol{v}_P^{n-1}\right)$$

The diffusion term is updated using a fully implicit technique as $\left(\nabla^2 \boldsymbol{v}/\mathrm{Re}\right)^{n+1/2} = \left(\nabla^2 \boldsymbol{v}/\mathrm{Re}\right)^{n+1}$ to improve numerical stability. The convective term can be updated using a fully implicit technique such as $(\boldsymbol{v}\cdot\nabla\boldsymbol{v})^{n+1/2} = (\boldsymbol{v}\cdot\nabla\boldsymbol{v})^{n+1}$ to improve numerical stability, or a fully explicit updating as $(\boldsymbol{v}\cdot\nabla\boldsymbol{v})^{n+1/2} = (\boldsymbol{v}\cdot\nabla\boldsymbol{v})^{n}$ for convenience.

We therefore have algorithm A as

$$\begin{cases} \mathbf{A}_P^n = \mathbf{A}_P(\boldsymbol{v}^n), \qquad \mathbf{A}_M^n = \mathbf{A}_M(\boldsymbol{v}^n) \\ \textbf{Do i} = 1, \mathbf{L} \\ \begin{cases} \mathbf{A}_P^{n+(i-1)/L} = \mathbf{A}_P\left(\boldsymbol{v}^{n+(i-1)/L}\right) \qquad \mathbf{A}_M^{n+(i-1)/L} = \mathbf{A}_M\left(\boldsymbol{v}^{n+(i-1)/L}\right) \\ \left(\frac{3}{2}\frac{\mathbf{I}}{\Delta t} + \mathbf{A}_P^{n+(i-1)/L}\right)\hat{\boldsymbol{v}}_P^{n+(i/L)} = \left[\frac{\mathbf{I}}{2\Delta t}\left(4\boldsymbol{v}_P^n - \boldsymbol{v}_P^{n-1}\right) + N(\boldsymbol{v}^n)\right] \\ \qquad + \sum_M \mathbf{A}_M^{n+(i-1)/L}\hat{\boldsymbol{v}}_M^{n+(i/L)} - \mathbf{G}p^{n+(i-1)/L} \\ \Delta t\,\mathbf{D}\left[\left(\frac{3}{2}\mathbf{I} + \mathbf{A}_P^{n+(i-1)/L}\right)^{-1}\mathbf{G}\left(p^{n+(i/L)} - p^{n+(i-1)/L}\right)\right] = \mathbf{D}\hat{\boldsymbol{v}}_P^{n+(i/L)} \\ \boldsymbol{v}_P^{n+(i/L)} = \hat{\boldsymbol{v}}_P^{n+(i/L)} - \Delta t\left(\frac{3}{2}\mathbf{I} + \mathbf{A}_P^{n+(i-1)/L}\right)^{-1}\mathbf{G}\left(p^{n+(i/L)} - p^{n+(i-1)/L}\right) \end{cases} \\ \textbf{EndDo} \end{cases} \tag{32}$$

and algorithm B as

$$\begin{cases} \hat{\boldsymbol{v}}^n = \boldsymbol{v}^n \\ \textbf{Do i} = 1, \mathbf{L} \\ \begin{cases} \hat{\mathbf{A}}_P^{n+(i-1)/L} = \mathbf{A}_P\left(\hat{\boldsymbol{v}}^{n+(i-1)/L}\right) \qquad \hat{\mathbf{A}}_M^{n+(i-1)/L} = \mathbf{A}_M\left(\hat{\boldsymbol{v}}^{n+(i-1)/L}\right) \\ \left(\frac{3}{2}\frac{\mathbf{I}}{\Delta t} + \hat{\mathbf{A}}_P^{n+(i-1)/L}\right)\hat{\boldsymbol{v}}_P^{n+(i/L)} = \left[\frac{I}{2\Delta t}\left(4\boldsymbol{v}_P^n - \boldsymbol{v}_P^{n-1}\right) + N(\boldsymbol{v}^n)\right] \\ \qquad + \sum_M \hat{\mathbf{A}}_M^{n+(i-1)/L}\hat{\boldsymbol{v}}_M^{n+(i/L)} - \mathbf{G}p^n \end{cases} \\ \textbf{EndDo} \\ \hat{\mathbf{A}}_P^{n+1} = \mathbf{A}_P\left(\hat{\boldsymbol{v}}^{n+1}\right) \qquad \hat{\mathbf{A}}_M^{n+1} = \mathbf{A}_M\left(\hat{\boldsymbol{v}}^{n+1}\right) \\ \Delta t\,\mathbf{D}\left[\left(\frac{3}{2}\mathbf{I} + \mathbf{A}_P^{n+1}\,\Delta t\right)^{-1}\mathbf{G}\left(p^{n+1} - p^n\right)\right] = \mathbf{D}\hat{\boldsymbol{v}}_P^{n+1} \\ \boldsymbol{v}_P^{n+1} = \hat{\boldsymbol{v}}_P^{n+1} - \Delta t\left(\frac{3}{2}\mathbf{I} + \mathbf{A}_P^{n+1}\Delta t\right)^{-1}\mathbf{G}\left(p^{n+1} - p^n\right) \end{cases} \tag{33}$$

In the above algorithms, if the convective term is updated using a fully explicit technique, $\mathbf{A}_P^n$ and $\mathbf{A}_M^n$ depend only on the grid size, and the algorithms of (32) and (33) will be identical if a direct numerical method, such as TDMA for a spatial three-point discretization scheme, can be employed to solve the discretized momentum equation. For this case, $L = 1$. However, supposing that the convective term is updated using a fully implicit technique, $N(\mathbf{v}^n) = 0$ and the coefficient matrices also depend on velocity and the computational efficiency of algorithms (32) and (33) will be different.

5. NUMERICAL VALIDATION

2-D Vortex Flows

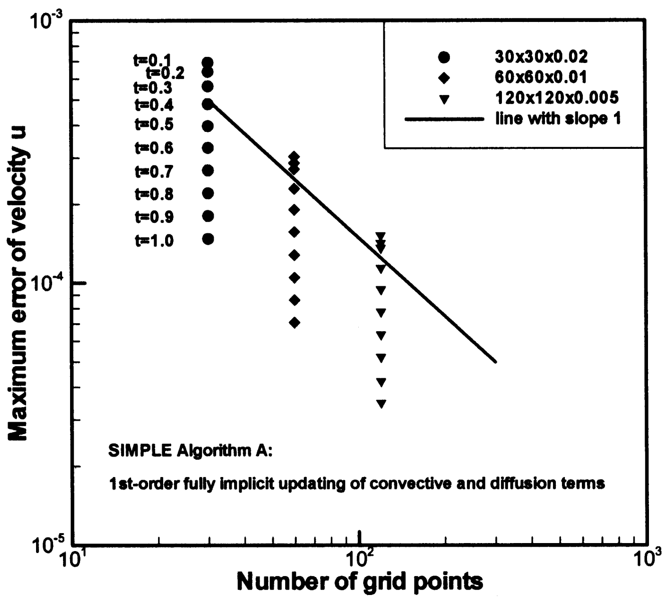
The proposed SIMPLE-type methods are tested in computing the following 2-D unsteady flow of decaying vertices [5], which has the following exact solution:

$$u(x, y, t) = -\cos(x)\,\sin(y)e^{-2t} \tag{34a}$$

$$v(x, y, t) = \sin(x)\,\cos(y)e^{-2t} \tag{34b}$$

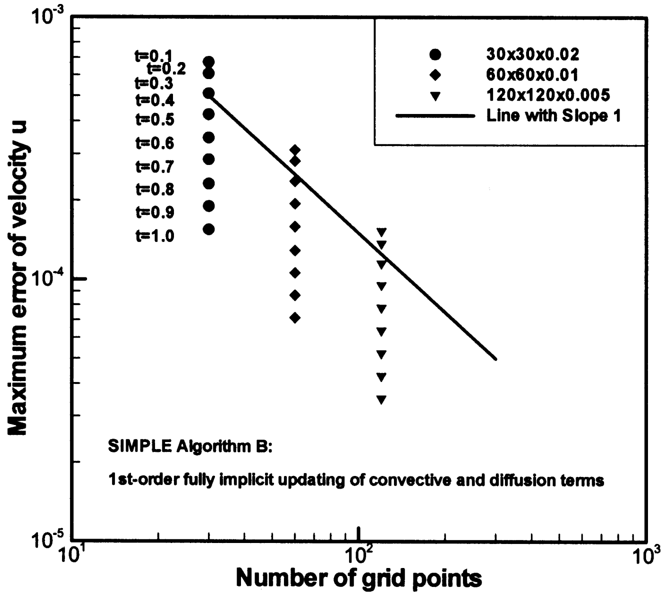
$$p(x, y, t) = -\tfrac{1}{4}[\cos(2x) + \cos(2y)]e^{-4t} \tag{34c}$$

Computations are carried out in the domain $0 \le x, y \le \pi$. In the computation, the uniform collocated meshes were used in the computation. On the boundaries, the

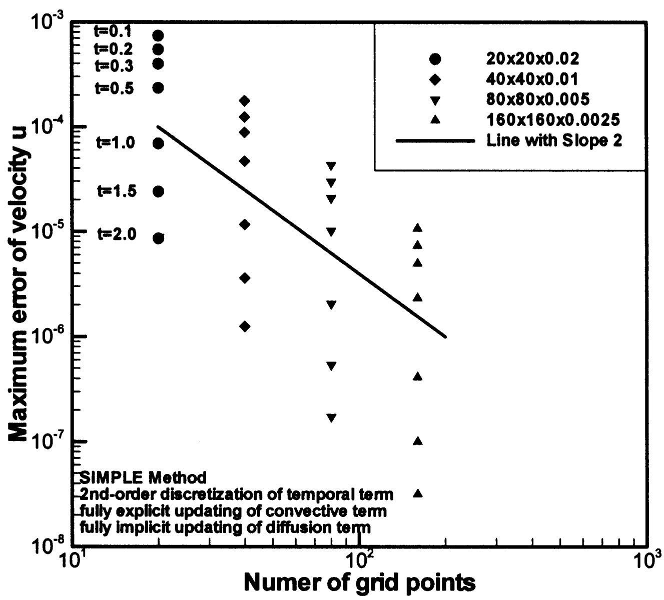


(*a*)

Figure 1. Maximum error of velocity u over the refined meshes: (*a*) by SIMPLE method A in Eq. (26); (*b*) by SIMPLE method B in Eq. (27); (*c*) by high-order SIMPLE method in Eqs. (32) and (33).



(*b*)



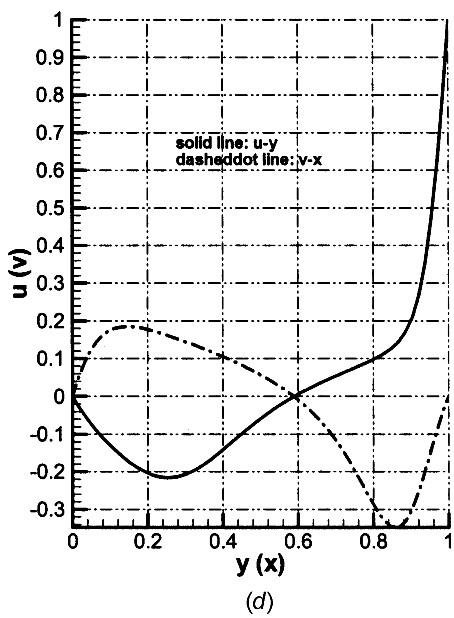
(*c*)

Figure 1. Continued.

exact solution is imposed. The same Courant-Fridrich-Levy (CFL) number is used for all of the tested algorithms, which means the time-step size is proportional to the grid size. Figure 1 plots the maximum errors in u, at different time levels as a function of mesh refinement. The y axis represents the maximum errors in u, while the x axis represents the mesh points. Similar results are obtained for v. Figures 1a and 1b are the results from SIMPLE methods A and B of Eqs. (26) and (27). From the figures, we can see that the slopes are equal to 1 and the temporal accuracy of Eqs. (26) and (27) is first-order. This is due to the employment of first-order fully implicit updating techniques for both convective and diffusion terms. The error is due mainly to the first-order error of Eq. (15). Using the high-order SIMPLE method III, in which the temporal term is discretized using a second-order temporal discretization scheme and the diffusion term is updated using a fully implicit scheme, Figure 1c shows that the slope is greater than 2 and the method has second-order temporal accuracy. Since the convective term is updated using a fully explicit scheme, Eqs. (32) of algorithm A and Eqs. (33) of algorithms B are identical. This validation shows that SIMPLE methods are in fact second-order temporal accuracy if a second-order updating is employed for the convective and diffusion terms.

3-D Lid-driven Flows in a Square Cavity

The lid-driven flow in a cubic cavity ($1 \times 1 \times 1$) has been widely used for validation and comparison purpose. The computation for $\mathrm{Re} = 400$ using the SIMPLEC method of Eqs. (21) with underrelaxation factor $\boldsymbol{\lambda} = \mathbf{I}$ is conducted on a $64 \times 64 \times 64$ uniform collocated grid system. The diffusion term is updated using the Crank-Nicholson scheme and the convective term is updated using the three-step Runge-Kutta technique. The results are shown in Figure 2. In Figure 2 the velocity profiles of the u component on the vertical centerline and the v component on the horizontal centerline of the plane $z = 0.5$ are shown. Both the positions and values of the extremes velocity are in good agreement with the result in [23]. The velocity vector



(d)

Figure 2. Velocity profiles (u component on vertical centerline, v component on horizontal center line).

plots projected onto three orthogonal midplanes are also displayed in Figure 3. The plots in y–z (b) and x–z(c) clearly demonstrate that the flow is completely 3-D even at this low Re. The flow structure of the velocity field and the position of the transversal vortex core of present calculation are quantitatively consistent with the plots of [23]. The velocity-vector plot of Figure 3 also clearly demonstrates the formation of steady transversal vortices, which are present in both the x–z and y–z planes. The strength of these transversal rolls is small compared with that of the main one. The above results are acquired using a time step of 0.05, which corresponds to an underrelaxation factor of

$$\lambda_X = \lambda_Y = \lambda_Z = \frac{A^n_{PX}\,\Delta t}{A^n_{PX}\,\Delta t + 1} = \frac{0.05{*}64^2/400}{0.05{*}64^2/400 + 1} = 0.338 \tag{35}$$

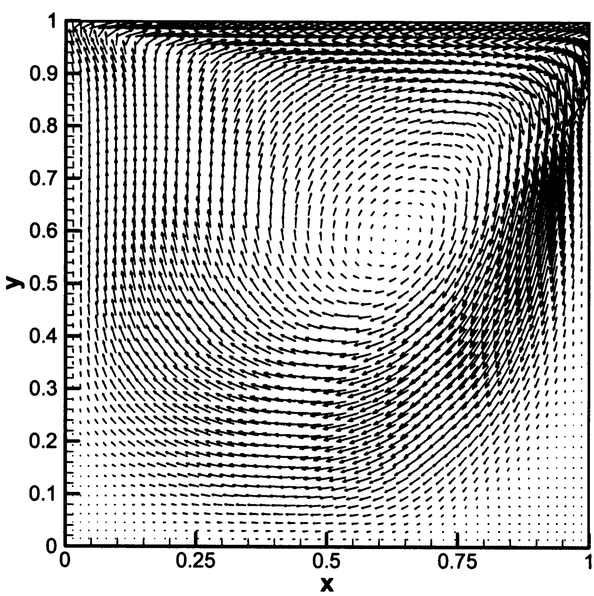
In the above formula,

$$A^n_{PX} = \frac{1}{\mathrm{Re}}\frac{1}{(\Delta x)^2}$$

since the Crank-Nicholson scheme is employed for the updating of the diffusion term and an explicit scheme is used for the convective term. This example validates the analysis of Eqs. (23).

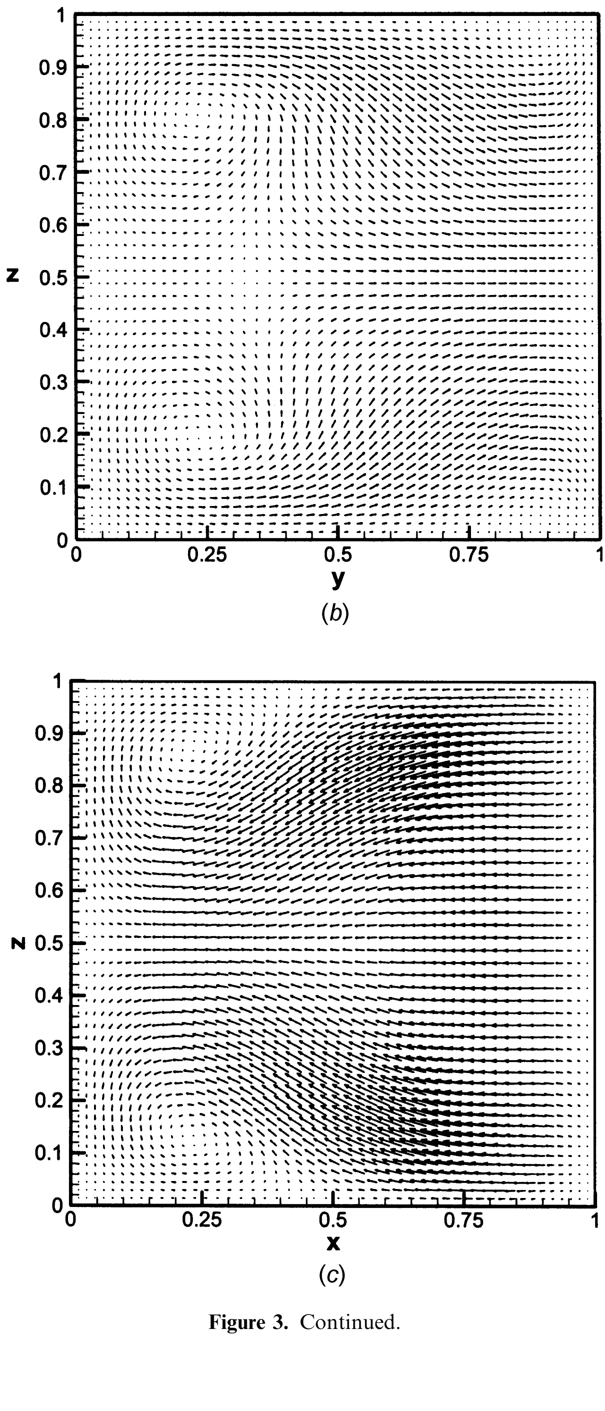
6. SUMMARY

1. The detailed temporal accuracy of SIMPLE-type methods is demonstrated. Error formulations are given, which include two parts. One is due to the



(a)

Figure 3. 3-D lid-driven cavity flow for Re = 400: (a) $u - v$ vector plot at midplane of $z = 0.5$; (b) $v - w$ vector plot at midplane of $x = 0.5$; (c) $u - w$ vector plot at midplane of $y = 0.5$.



(*b*)

(*c*)

Figure 3. Continued.

temporal updating technique employed for the temporal term, the other is due to the SIMPLE-type method itself.

2. SIMPLE-like methods have second-order temporal accuracy for unsteady flows, supposing that second-order-temporal-accuracy schemes are employed to update the temporal terms.
3. The standard second-order projection method can be constructed using the idea for the construction of SIMPLEC method.
4. A four-step standard SIMPLE method has been given, in which pressure is acquired from the pressure Poisson equation to correct the velocity, whereas in the original three-step SIMPLE method, a pressure difference from a Poisson equation is used to correct velocity.
5. Several high-order-accuracy SIMPLE-like methods are presented for the calculation of unsteady flows. Two algorithms of SIMPLE-like methods are described and discussed. Algorithm A is a kind of iteration method, in which the solution of the pressure Poisson equation costs more time. Algorithm B is a noniteration method, which is more economical than algorithm A.

REFERENCES

1. F. H. Harlow and J. E. Welch, Numerical Calculation of Time-Dependent Viscous Incompressible Flow of Fluid with Free Surface, *Phys. Fluids*, vol. 8, no. 12, pp. 2182–2189, 1965.
2. A. J. Chorin, Numerical Solution of Navier–Stokes Equations, *Math. Comput.*, vol. 22, pp. 745, 1968.
3. J. B. Bell, P. Colella, and H. M. Glaz, A Second-Order Projection Method for the Incompressible Navier-Stokes Equations, *J. Comput. Phys.*, vol. 85, p. 257, 1989.
4. H. Choi and P. Moin, Effects of the Computational Time Step on Numerical Solutions of Turbulent Flow, *J. Comput. Phys.*, vol. 113, pp. 1–4, 1994.
5. J. Kim and P. Moin, Application of Fractional-Step Method to Incompressible Navier-Stokes Equations, *J. Comput. Phys.*, vol. 59, p. 108, 1985.
6. M.-J. Ni, S. Komori, and N. Morley, General Second-Order Projection Formulas for Unsteady Flows, *Numer. Heat Transfer B*, vol. 44, pp. 533–551, 2003.
7. M. M. Rai and P. Moin, Direct Simulation of Turbulent Flow Using Finite Difference Schemes, *J. Comput. Phys.*, vol. 96, p. 15, 1991.
8. M.-J. Ni, M. Abdou, and S. Komori, A Variable-Density Projection Method for Interfacial Flows, *Numer. Heat Transfer B*, vol. 44, pp. 553–574, 2003.
9. 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*, vol. 15, pp. 1787–1806, 1972.
10. S. V. Patankar, *Numerical Heat Transfer and Fluid Flow*, McGraw-Hill, New York, 1980.
11. S. V. Patankar, A Calculation Procedure for Two-Dimensional Elliptic Situations, *Numer. Heat Transfer*, vol. 4, pp. 409–425, 1981
12. J. P. Van Doormaal and G. D. Raithby, Enhancement of the SIMPLE Method for Predicting Incompressible Fluid Flows, *Numer. Heat Transfer*, vol. 7, pp. 147–163, 1984.
13. W. Shyy, *Computational Modeling for Fluid Flow and Interfacial Transport*, Elsevier, Amsterdam, 1994.
14. W. Q. Tao, *Recent Advances in Computational Heat Transfer*, Science Press, China, 2000.
15. L. Chen, S. V. Garimella, J. A. Reizes, and E. Leonardi, The Development of a Bubble Rising in a Viscous Liquid, *J. Fluid Mech.*, vol. 387, pp. 61–96, 1999.
16. H. S. Udaykumar, H.-C. Kan, W. Shyy, and R. Tran-Son-Tay, Multiphase Dynamics in Arbitrary Geometries on Fixed Cartesian Grids, *J. Comput. Phys.*, vol. 137, pp. 366–405, 1997.

17. H. Suzuki and H. Usui, Direct Numerical Simulation of a Turbulent Duct Flow by High-Ordered Scheme for SIMPLE, *13th Japanese Symp. on Computational Fluid Dynamics* (JSCFD), B04–3, 1999.
18. R. I. Issa, Solution of the Implicitly Discretized Fluid Flow Equations by Operator-Splitting, *J. Comput. Phys.*, vol. 62, pp. 40–65, 1985.
19. A. A. Mohamad, Benchmark Solution for Unsteady State CFD Problems, *Numer. Heat Transfer A.*, vol. 34, pp. 653–672, 1998.
20. M.-J. Ni, W.-Q. Tao, and S.-J. Wang, Stability Analysis for Discretized Steady Convective-Diffusion Equation, *Numer. Heat Transfer B*, vol. 35, pp. 369–388, 1999.
21. W.-Q. Tao, Z. G. Qu, and Y. L. He, A Novel Segregated Algorithm for Incompressible Fluid Flow and Heat Transfer Problems—CLEAR (Coupled and Linked Equations Algorithm Revised), Part I: Mathematical Formulation and Solution Procedure, *Numer. Heat Transfer B*, vol. 45, pp. 1–17, 2004.
22. W.-Q. Tao, Z. G. Qu, and Y. L. He, A Novel Segregated Algorithm for Incompressible Fluid Flow and Heat Transfer Problems—CLEAR (Coupled and Linked Equations Algorithm Revised) Part II: Application Examples, *Numer. Heat Transfer B*, vol. 45, pp. 19–48, 2004.
23. G. Guj and F. Stella, A Vorticity Method for the Numerical Solution of 3D Incompressible Flows, *J. Comput. Phys.*, vol. 106, pp. 286–298, 1993.