Numerical Heat Transfer: An International Journal of Computation and Methodology
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/unht19

COMPARISON OF THE PISO, SIMPLER, AND SIMPLEC ALGORITHMS FOR THE TREATMENT OF THE PRESSURE-VELOCITY COUPLING IN STEADY FLOW PROBLEMS

D. S. Jang [a] , R. Jetli [a] & S. Acharya [a]
[a] Mechanical Engineering Department, Louisiana State University, Baton Rouge, Louisiana, 70803
Version of record first published: 02 Mar 2007.

To cite this article: D. S. Jang , R. Jetli & S. Acharya (1986): COMPARISON OF THE PISO, SIMPLER, AND SIMPLEC ALGORITHMS FOR THE TREATMENT OF THE PRESSURE-VELOCITY COUPLING IN STEADY FLOW PROBLEMS, Numerical Heat Transfer: An International Journal of Computation and Methodology, 10:3, 209-228

To link to this article: http://dx.doi.org/10.1080/10407788608913517

Numerical Heat Transfer, vol. 10, pp. 209–228, 1986

COMPARISON OF THE PISO, SIMPLER, AND SIMPLEC ALGORITHMS FOR THE TREATMENT OF THE PRESSURE–VELOCITY COUPLING IN STEADY FLOW PROBLEMS

D. S. Jang, R. Jetli, and S. Acharya
Mechanical Engineering Department, Louisiana State University, Baton Rouge, Louisiana 70803

The performance of the PISO, SIMPLER, and SIMPLEC algorithms for the treatment of the pressure-velocity coupling in steady flow problems is examined by comparing the computational effort required to obtain the same level of convergence in four test problems. For problems in which the momentum equation is not coupled to a scalar variable the PISO algorithm is superior, but when the scalar variable is strongly coupled to the momentum equation SIMPLER and SIMPLEC exhibit better behavior and reasonable solutions with the PISO algorithm are obtained only for small time steps. Clear superiority of SIMPLER over SIMPLEC or vice versa is not observed, although the iterative formulations of these two algorithms exhibit more robust behavior than the corresponding time-marching formulations.

INTRODUCTION

Since its conception in 1972, the SIMPLE (semi-implicit method for the pressure linked equations) algorithm of Patankar and Spalding [1] has been extensively used in resolving the pressure-velocity coupling in incompressible flow problems. Over the years, a number of modifications to the SIMPLE algorithm have been proposed [2–4] and shown to have better convergence properties. The SIMPLER (SIMPLE revised) algorithm of Patankar [3] and SIMPLEC (SIMPLE consistent) algorithm of Van Doormaal and Raithby [4] are examples of such modifications and appear to exhibit better behavior than SIMPLE and its other variants. More recently, Issa and co-workers [5, 6] have proposed the PISO (pressure-implicit with splitting of operators) algorithm, which is a noniterative time-marching procedure and has been shown [6] to be superior to the SIMPLE method. However, no direct comparison between the PISO algorithm and the SIMPLER and SIMPLEC methods has been made. This is the objective of the present paper.

The primary interest in this paper is in steady-state results. Results with the PISO algorithm have been calculated, as proposed in [5], by solving the time-dependent equations. Steady-state solutions with the SIMPLER algorithm have generally been obtained by starting out with the steady-state equations and then solving them by an iterative procedure requiring under-relaxation of the finite-difference equations. This approach has been adopted here in obtaining the steady-state results

Support for this work was provided in part by DOW Chemical Co., Plaquemine, Louisiana. This support is gratefully acknowledged.

NOMENCLATURE

A	control volume face area
a	coefficient in discretization equation
b	source term in discretization equation, mass source in pressure correction equation
d	coefficient of pressure difference term
k	turbulence kinetic energy
i	stoichiometric ratio, mass of oxygen per unit mass of fuel consumed
L_f	length of furnace
m	mass fraction
p	pressure
p'	pressure correction
r	radial coordinate
R_{eff}	effective radius for swirl number calculation
R_f	furnace radius
Ra	Rayleigh number
S	source term, swirl number
S_c	constant part of linearized source term
S_p	ϕ-dependent part in linearized source term expression
t	time
U	maximum x-direction velocity used in definition of dimensionless time
u	x-direction velocity
$\mathbf{u}$	velocity vector
V	maximum y-direction velocity used in definition of dimensionless time
v	y-direction velocity
ΔV	volume of control volume
W	molecular weight
w	θ-direction velocity
x, y	coordinates
Γ	diffusion coefficient
Δr	minimum distance between adjacent grid points in r direction
Δt	time step
Δx	minimum distance between adjacent grid points in x direction
ϵ	turbulence dissipation rate
θ	angular coordinate, dimensionless temperature
λ	relaxation factor
μ_t	turbulent viscosity
ρ	density
ϕ	dependent variable

Subscripts

E	neighbor grid point in positive x direction
e	control volume face between P and E
fu	fuel
fx	mixture
N	neighbor grid point in positive y direction
n	control volume face between P and N
nb	neighboring grid point
ox	oxygen
P	central grid point under consideration
pr	product
S	neighbor grid point in negative y direction, swirl number
s	control volume face between P and S
W	neighbor grid point in negative x direction
w	control volume face between P and W

Superscripts

o, n	old value of variable
*	previous-iteration value of variable or guessed value, value in the predictor level
, *	value in corrector level
$\hat{}$, $\tilde{}$	pseudovelocity
$\bar{}$	main coefficient of discretised velocity equations
$'$	pressure or velocity correction

with the SIMPLER and SIMPLEC algorithms. In addition, to compare the iterative (with under-relaxation) and time-marching procedures for calculating steady-state solutions, representative steady-state results with the time-marching formulations of the SIMPLER and SIMPLEC methods have been obtained. Differences between the two approaches are noted and pointed out.

SOLUTION ALGORITHMS

Since the SIMPLE and SIMPLER algorithms are variants of the SIMPLE method, the SIMPLE algorithm will be described first, followed by descriptions of the SIMPLEC, SIMPLER, and PISO algorithms. In describing these algorithms it is presumed that the reader is familiar with the basic ideas and notations in the well-known SIMPLE method of Patankar and Spalding [1, 7].

The differential equation expressing the conservation of momentum in two dimensions can be written as

$$\frac{\partial}{\partial t}(\rho\phi) + \nabla\cdot(\rho\mathbf{u}\phi) = -\nabla p + \nabla\cdot(\Gamma\,\nabla\phi) + (S_c + S_p\phi) \tag{1}$$

where ϕ represents either u or v, Γ is the diffusion coefficient, and $S_c + S_p\phi$ is the linearized source term. In the Patankar-Spalding method [1, 7] the domain is subdivided into a number of control volumes, each associated with a grid point. The scalar variables and pressure are stored at the grid points while the velocities are stored at the staggered locations as shown in Fig. 1. The finite-difference equations are obtained by integrating the momentum equations over each staggered control volume and expressing the variations of ϕ in each coordinate direction by suitable profile approximations. This results in the discretization equations for u_e and v_n

$$\left(a_e + \frac{\rho\,\Delta V}{\Delta t}\right)u_e = \Sigma a_{nb}u_{nb} + A_e(p_P - p_E) + S_c\,\Delta V + \frac{\rho\,\Delta V}{\Delta t}u_e^o \tag{2}$$

$$\left(a_n + \frac{\rho\,\Delta V}{\Delta t}\right)v_n = \Sigma a_{nb}v_{nb} + A_n(p_P - p_N) + S_c\,\Delta V + \frac{\rho\,\Delta V}{\Delta t}v_n^o \tag{3}$$

where the summation on the right side of Eqs. (2) and (3) is done along the four neighbors (indicated by subscripts *nb*) and the superscript o denotes the previous time step values. The expressions for the coefficients a_e, a_n, and a_{nb} depend on the choice of the profile approximations in each coordinate direction.

Equations (2) and (3) represent the finite-difference analog of the unsteady momentum equations, and the steady-state results can be obtained by marching in time. Since only steady-state solutions are of interest, it is not necessary to obtain converged results of the system of algebraic equations at each time step. Therefore, at each time step the coefficients are calculated only once, based on the previous time step solution, and a partially converged result of the system of equations is obtained. At successive time steps, the results approach the steady-state values. This approach eliminates the need to store the dependent variable values at the present and previous time steps and is termed the time-marching procedure in this paper.

An alternative approach, and one that is commonly used, is to start with the steady-state form of Eqs. (2) and (3) and solve these equations in an iterative framework using under-relaxation. The under-relaxed discretization equations have the following form:

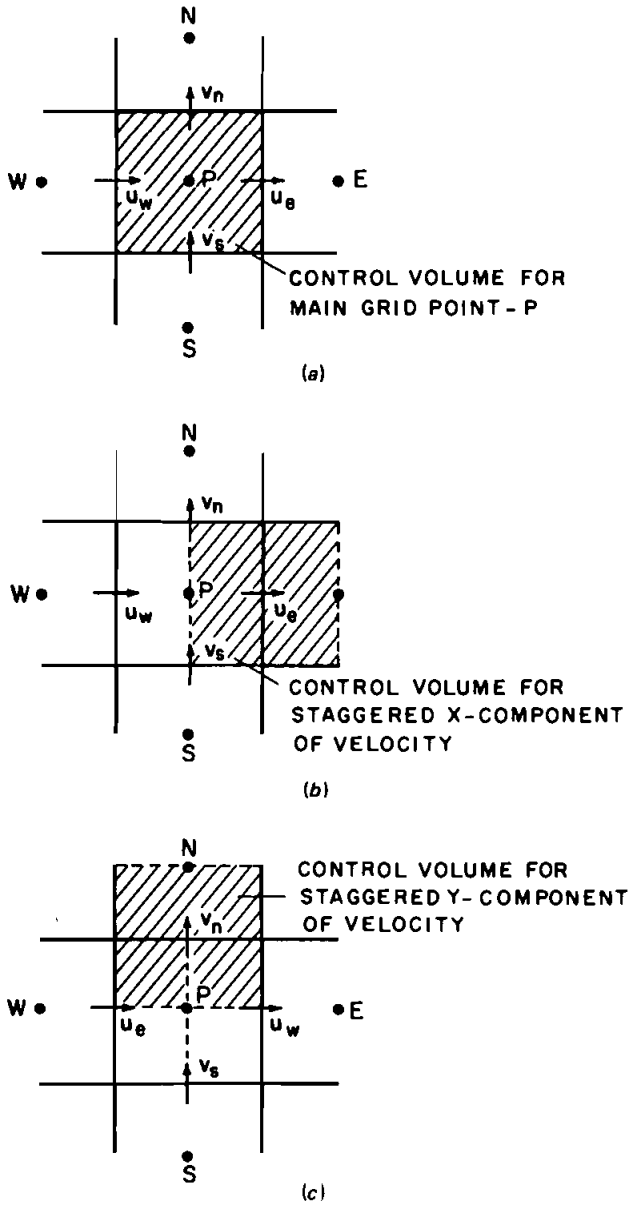


Fig. 1 Schematic of main and staggered control volumes.

$$a_e\left(1+\frac{1-\lambda}{\lambda}\right)u_e = \Sigma a_{nb}u_{nb} + A_e(p_P - p_E) + S_c\,\Delta V + \frac{1-\lambda}{\lambda}\,a_e u_e^* \tag{4}$$

$$a_n\left(1+\frac{1-\lambda}{\lambda}\right)v_n = \Sigma a_{nb}v_{nb} + A_n(p_P - p_N) + S_c\,\Delta V + \frac{1-\lambda}{\lambda}\,a_n v_n^* \tag{5}$$

where λ is the under-relaxation factor and u_e^*, v_n^* indicate results from the previous iteration. Successive solution of Eqs. (4) and (5), with the coefficients updated at

each iteration, results in the steady-state solution. This approach is called the iterative procedure in this paper.

In comparing the time-marching formulation [Eqs. (2) and (3)] with the iterative one [Eqs. (4) and (5)], it can be seen that the unsteady term in Eq. (1) and the under-relaxation practice have similar effects on the finite-difference equations. In fact, Eqs. (4) and (5) would have the same form as Eqs. (2) and (3) if λ was expressed by

$$\frac{1-\lambda}{\lambda} a_e = \frac{\rho \Delta V}{\Delta t} \qquad \text{or} \qquad \lambda = \frac{a_e}{a_e + \rho \Delta V/\Delta t} \tag{6}$$

It should be noted that λ in Eq. (6) depends on a_e, which in turn depends on the velocities. Therefore, λ in the time-marching formulation varies spatially and evolves with the solution. In constrast, in the iterative formulation λ is a constant. As mentioned earlier, the two formulations will be compared with each other. In the discussion that follows, the discretized momentum equations [Eqs. (2), (3) or (4), (5)] are first reexpressed (for convenience) as

$$\tilde{a}_e u_e = \Sigma a_{nb} u_{nb} + A_e(p_P - p_E) + b_e \tag{7}$$

$$\tilde{a}_n v_n = \Sigma a_{nb} v_{nb} + A_n(p_P - p_N) + b_n \tag{8}$$

SIMPLE Algorithm

For a guessed pressure field p^*, the velocities u^* and v^* satisfy

$$\tilde{a}_e u_e^* = \Sigma a_{nb} u_{nb}^* + A_e(p_P^* - p_E^*) + b_e \tag{9}$$

$$\tilde{a}_n v_n^* = \Sigma a_{nb} v_{nb}^* + A_n(p_P^* - p_N^*) + b_n \tag{10}$$

If Eqs. (9) and (10) are subtracted from Eqs. (7) and (8), the fully implicit velocity correction (u', v') equations are obtained as

$$\tilde{a}_e u_e' = \Sigma a_{nb} u_{nb}' + A_e(p_P' - p_E') \tag{11}$$

$$\tilde{a}_n v_n' = \Sigma a_{nb} v_{nb}' + A_n(p_P' - P_N') \tag{12}$$

where

$$u' = u - u^* \qquad v' = v - v^* \qquad p' = p - p^* \tag{13}$$

In the SIMPLE algorithm, the velocity correction equations are obtained by dropping the $\Sigma a_{nb} u_{nb}'$ and $\Sigma a_{nb} v_{nb}'$ terms on the right side of the Eqs. (11) and (12), resulting in

$$u_e = u_e^* + d_e(p_P' - p_E') \qquad v_n = v_n^* + d_n(p_P' - p_N') \tag{14}$$

where

$$d_e = \frac{A_e}{\tilde{a}_e} \qquad d_n = \frac{A_n}{\tilde{a}_n} \tag{15}$$

To obtain the pressure correction equation, Eq. (14) is substituted into the discretized continuity equation for the control volume around the main grid point, i.e.,

$$(\rho_p - \rho_p^o)\frac{\Delta V}{\Delta t} + \rho_e u_e A_e - \rho_w u_w A_w + \rho_n v_n A_n - \rho_s v_s A_s = 0 \tag{16}$$

The resulting pressure correction equation has the following form (see Fig. 1 for notation):

$$a_P p'_P = a_E p'_E + a_W p'_W + a_N p'_N + a_S p'_S + b \tag{17}$$

where

$$a_E = \rho_e A_e d_e \qquad a_W = \rho_w A_w d_w \qquad a_N = \rho_n A_n d_n \qquad a_S = \rho_s A_s d_s \tag{18}$$

$$a_P = a_E + a_W + a_N + a_S \tag{19}$$

$$b = (\rho_p^o - \rho_p)\frac{\Delta V}{\Delta t} + \rho_w u_w^* A_w - \rho_e u_e^* A_e + \rho_s v_s^* A_s - \rho_n v_n^* A_n \tag{20}$$

and u^* denotes the previous iteration value.

Before describing the SIMPLE algorithm, it should be pointed out that equations of the pentadiagonal system such as Eqs. (9), (10), and (17) are solved by a line-by-line Thomas algorithm in which repeated sweeps of the algorithm are performed in each coordinate direction until the correct solution to the system of equations is obtained. Since at any step the coefficients are tentative (being nonlinear), a very accurate solution to the system of equations is unnecessary and, in the interest of computational economy, is undesirable. Therefore, in most of the cases reported here, only a few (one or two) sweeps of the Thomas algorithm are performed in each coordinate direction before updating the coefficients for the next step. In this regard the present approach may differ from others in which, at each time step or iteration, the Thomas algorithm sweeps are repeated until a prespecified residual criterion is satisfied.

The SIMPLE algorithm consists of the following steps:

1. Guess the pressure field p^*.
2. Solve the momentum equations [Eqs. (9) and (10)] to obtain u^* and v^*.
3. Solve the p' equation [Eq. (17)] and update the pressure by $p = p^* + \lambda_p p'$, where λ_p is an under-relaxation factor.
4. Update velocities u and v by the velocity correction equation [Eq. (14)].
5. Repeat steps 2 and 4 until convergence is reached (each step is a time step in the time-marching formulation and an iteration in the iterative formulation).

SIMPLEC Algorithm

The SIMPLEC algorithm follows the same general steps as the SIMPLE algorithm; the primary difference is in the formulation of the velocity correction equa-

tion. In this approach $u'_e\Sigma a_{nb}$ is subtracted from both sides of Eqs. (11) and the term $\Sigma a_{nb}(u'_{nb} - u'_e)$ on the right side of the equation is neglected. A similar operation is performed on the v' equation. The resulting velocity correction equations have the same form as Eq. (14) but with d_e and d_n redefined as

$$d_e = \frac{A_e}{\tilde{a}_e - \Sigma a_{nb}} \qquad d_n = \frac{A_n}{\tilde{a}_n - \Sigma a_{nb}} \tag{21}$$

The pressure correction equation is exactly as described earlier [Eq. (17)] with the d's in Eq. (18) computed from equations such as Eq. (21) rather than Eq. (15).

The basic algorithm of the SIMPLEC method is identical to the steps in the SIMPLE method but with λ_p in the pressure equation set equal to 1.

SIMPLER Algorithm

In developing the SIMPLER procedure it was recognized that the pressure correction equation in the SIMPLE algorithm, in view of the approximation that $\Sigma a_{nb}u'_{nb} = \Sigma a_{nb}v'_{nb} \cong 0$, was responsible for the slow convergence of the pressure field. Thus a more suitable equation for updating the pressure field was derived. This was done by first defining pseudo-velocities $\hat{u}_e$ and $\hat{v}_n$ as

$$\hat{u}_e = \frac{\Sigma a_{nb}u_{nb} + b_e}{\tilde{a}_e} \qquad \hat{v}_n = \frac{\Sigma a_{nb}v_{nb} + b_n}{\tilde{a}_n} \tag{22}$$

Thus, Eqs. (7) and (8) are reexpressed as

$$u_e = \hat{u}_e + d_e(p_P - p_E) \qquad v_n = \hat{v}_n + d_n(p_P - p_N) \tag{23}$$

where d_e and d_n are defined by Eq. (15). Substituting Eq. (23) into Eq. (16) leads to the pressure equation:

$$a_P p_P = a_E p_E + a_W p_W + a_N p_N + a_S p_S + b \tag{24}$$

where a_E, a_W, a_N, and a_S are given by Eq. (18), a_P by Eq. (19), and b by Eq. (20) with u^* and v^* replaced by $\hat{u}$ and $\hat{v}$. Equation (24) is a better approximation than the corresponding p' equation [Eq. (17)] in the SIMPLE method.

The SIMPLER algorithm can be described by the following sequence of operations:

1. Using the currently available velocity field, calculate $\hat{u}$, $\hat{v}$ from Eq. (22).
2. Calculate coefficients of the pressure equation [Eq. (24)] and solve for the updated pressure field.
3. Using this pressure field, solve the momentum equation to calculate u^* and v^*.
4. Solve the p' equation [Eq. (17)].
5. Correct the velocity field by the velocity correction equation [Eq. (14)].
6. Repeat steps 1–5 until steady-state or converged solutions are obtained.

PISO Algorithm

The PISO algorithm, as proposed by Issa [5], is a time-marching procedure in which during each time step there is a predictor step and one or more corrector steps. Issa [5] recommends that the equations be first recast in an incremental form to minimize the computing effort and reduce storage requirements. For convenience in notation, the converged solution at the previous time step is denoted by a superscript n while the solutions at the present time step are denoted by a superscript * at the predictor level, ** at the first corrector level, and *** at the second corrector level.

Equation (7) can be expressed implicitly as

$$\tilde{a}_e u_e^* = \Sigma a_{nb} u_{nb}^* + A_e(p_P^n - p_E^n) + b_e^n \tag{25}$$

and then updated by

$$\tilde{a}_e u_e^{**} = \Sigma a_{nb} u_{nb}^* + A_e(p_P^* - p_E^*) + b_e^n \tag{26}$$

Subtracting Eq. (25) from Eq. (26) gives the increment equation

$$u_e^{**} = \hat{u}_e + d_e(p_P^* - p_E^*) \tag{27}$$

where

$$\hat{u}_e = u_e^* - d_e(p_P^n - p_E^n) \tag{28}$$

and d_e is given by Eq. (15). Similar equations can be derived for u_w^{**}, v_n^{**}, and v_s^{**}. Equation (25) is the velocity predictor equation while Eq. (27) represents the first corrector step for velocity. Substitution of Eq. (27) into the continuity equation (16) leads to the pressure equation, which is of the same form as Eq. (24) with the $\hat{u}$ and $\hat{v}$ in the coefficients given by equations like (28) instead of Eq. (22). This equation is the predictor equation for pressure.

The next set of corrector equations can be derived by expressing Eq. (7) as

$$\tilde{a}_e u_e^{***} = \Sigma a_{nb} u_{nb}^{**} + A_e(p_P^{**} - p_E^{**}) + b_e^n \tag{29}$$

and subtracting Eq. (26) from the above equation. This results in a second corrector equation for velocity

$$u_e^{***} = \hat{\hat{u}}_e + d_e(p_P^{**} - p_E^{**}) \tag{30}$$

where

$$\hat{\hat{u}}_e = u_e^{**} + [\Sigma a_{nb}(u_{nb}^{**} - u_{nb}^*)]/\tilde{a}_e - d_e(p_P^* - p_E^*) \tag{31}$$

Similar equations can be derived for u_w^{***}, v_n^{***}, and v_s^{***}.

To derive the pressure corrector equation, equations of the form of (30) are substituted into the discretized continuity equation, and again a system of equations

represented by Eq. (24) is obtained. This equation represents the pressure corrector equation.

The PISO algorithm can be described by the following steps:

1. Using the previous time step solution (u^n, v^n, p^n), calculate the coefficients of the momentum equations [such as Eq. (25)] and solve this system of implicit equations. This represents the predictor step for velocity.
2. Using this predicted velocity, calculate coefficients of pressure equation (24) and solve for the pressure field. This is the predictor step for the pressure.
3. Correct (first corrector step for velocity) the velocity field, using explicit-type equations such as Eq. (27).
4. Using the corrected velocities, calculate the coefficients of the pressure equation in the corrector step and solve the implicit system of equations to obtain an updated pressure field.
5. Using the corrected pressure field, reevaluate the velocity field (second corrector step) using explicit-type equations such as Eq. (30).
6. March to the next time step.

In solving turbulent flow problems with the two-equation turbulence model [8], the equations for the kinetic energy of the turbulence k and its dissipation rate ϵ are strongly coupled with each other and with the momentum equations. Thus to preserve the convergence properties of the PISO algorithm, a predictor-corrector operation in each time step is proposed by Issa [5]. This splitting operation is essentially similar to that described earlier for the momentum equations with an implicit predictor step and an explicit-type corrector step. For other scalar variables, which are linked to the momentum equation through the density or the source term, a two-stage PISO scheme has been employed in which only a predictor step following the first corrector step of the momentum equation is used to determine the value of the scalar variable at each time step.

RESULTS AND DISCUSSION

Results are presented for both the time-marching and iterative formulations with the intent of comparing the relative performance of the two approaches. Four test problems are studied. Test problem 1 deals with the sudden expansion of laminar flow in a plane geometry. In test problem 2, isothermal turbulent flow in an axisymmetric geometry with expansion at the inlet and contraction at the exit is examined. Test problem 3 considers reacting, swirling flow in an axisymmetric furnace configuration. The last test problem deals with laminar free convection in an externally heated square enclosure with perfectly conducting horizontal end walls. Two grid sizes are used in obtaining the results: a coarse 7×10 or 7×12 grid and a finer 20×20 grid. Calculations on a very fine grid (80×80) have not been made in view of the associated computational expense, and therefore results in this paper may not necessarily be valid for very fine grids. In discussing the results, the relative performance of the various algorithms will be examined and the time-marching and iterative formulations compared. Results are obtained over a suitable range of relaxation factors λ in the iterative formulation and over a range of dimensionless time

steps $\Delta tU/\Delta x$ or $\Delta tV/\Delta r$ in the time-marching formulation. All results are driven to the same level of convergence.

Test Problem 1: Sudden Expansion of Flow in a Plane Channel

A schematic of the test problem configuration is shown in Fig. 2. The equations of momentum and continuity are solved in the Cartesian coordinate system with symmetry conditions imposed along the vertical centerline and zero-gradient conditions at the outflow boundary.

In comparing the PISO results with the iterative SIMPLER formulation, the computing effort with the PISO algorithm is clearly less for both a coarse (7 × 12) grid and a finer (20 × 20) grid and there is a greater saving of computational effort at the finer grid size (Fig. 3). PISO is also superior to the iterative SIMPLEC formulation on the 20 × 20 grid, while the two algorithms exhibit comparable behavior on the coarse 7 × 12 grid with SIMPLEC requiring slightly shorter computing times. The SIMPLEC results, for both grid sizes, generally require less computing time than the SIMPLER solutions. When the iterative and time-marching SIMPLER and SIMPLEC formulations are compared, the SIMPLEC time-marching algorithm exhibits the best behavior. Converged results with the SIMPLER time-marching method are obtained only for small time steps. It should be mentioned that in making these comparisons, the general trends of the plotted lines are compared rather than the CPU seconds for a particular value along the abscissa (λ or dimensionless time). Also, in evaluating the performance of the various algorithms, both the computational effort required and the range of λ or Δt over which the algorithm has small

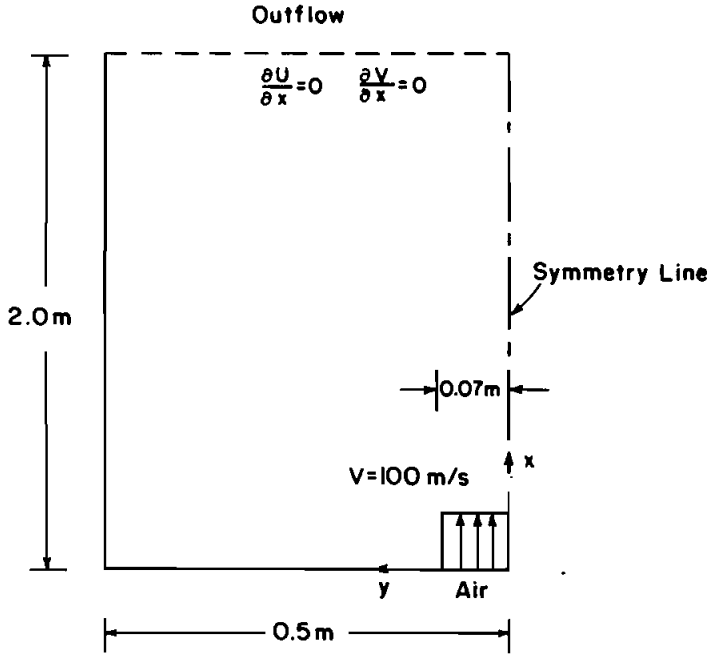


Fig. 2 Sudden expansion of flow in a plane channel.

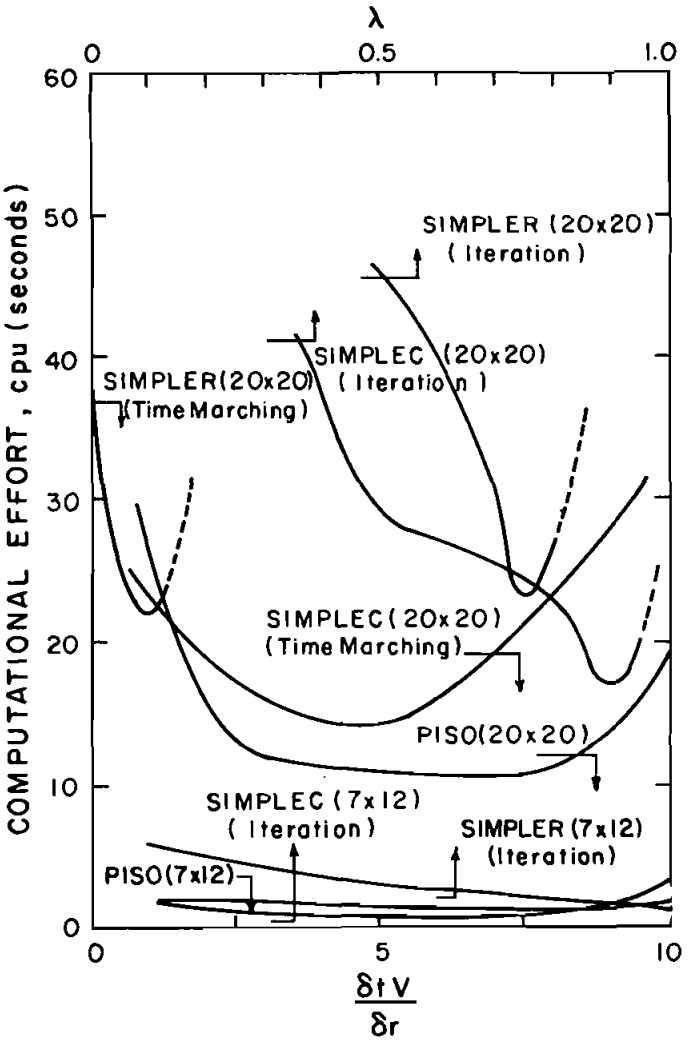


Fig. 3 Comparison of computational effort for the sudden expansion flow.

computational requirements are considered. In this regard, methods that exhibit a sharp dip in CPU seconds followed by a quick increase are not robust and stable methods.

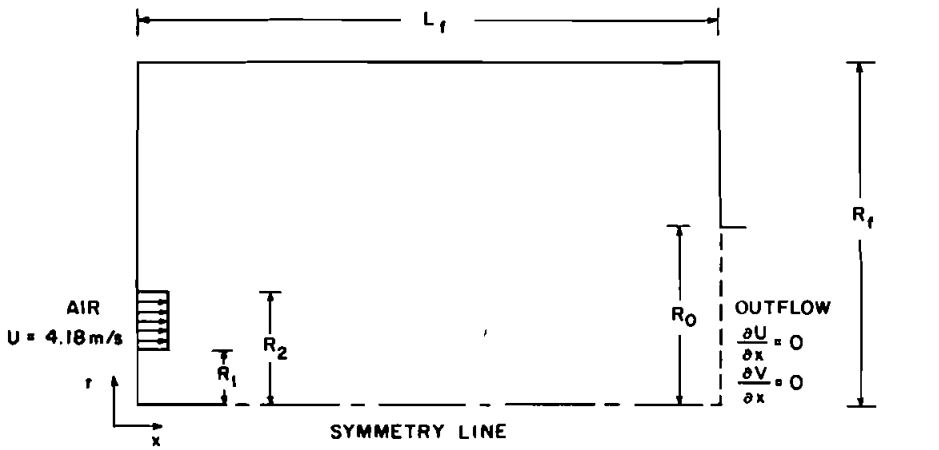
To explain some of the observed behavior it should be noted that in the SIMPLER approach the velocities are updated [Eq. (14)] on the basis of pressure corrections obtained by neglecting $\Sigma a_{nb}u'_{nb}$ and $\Sigma a_{nb}v'_{nb}$ terms in the u' and v' equations. Since these terms could be significant, the SIMPLEC algorithm improves on the velocity correction formula by neglecting $\Sigma a_{nb}(u'_{nb} - u'_e)$ and $\Sigma a_{nb}(v'_{nb} - v'_n)$ instead. However, unlike the SIMPLER algorithm, no separate equation for pressure is solved, and therefore neglect of the velocity correction terms in the u' and v' equations is reflected in the pressure equation ($p = p^* + p'$). Thus, it may be surmised that in

the SIMPLEC method the velocity corrections are closer to the true value while the pressure corrections are less accurate than the corresponding values in the SIMPLER algorithm. Keeping this in mind and noting that in the SIMPLER method an additional system of equation represented by Eq. (24) must be solved, distinct superiority of one method over another in all possible cases is unlikely.

PISO, on the other hand, appears to combine the advantages of the SIMPLER and SIMPLEC algorithms. The pressure predictor equation in PISO is similar to that in the SIMPLER method [Eq. (24)], while the velocity correction equations [Eq. (27) and (30)] are better approximations than those in the SIMPLER and SIMPLEC methods since no terms are neglected (as in the SIMPLER and even SIMPLEC approximation). Therefore, the solution is likely to converge faster even if PISO, due to a larger sequence of operations per step, requires more computational effort in each time step.

Test Problem 2: Turbulent Flow in an Axisymmetric Geometry

The second test problem again deals with sudden expansion of flow but for turbulent flow conditions in the axisymmetric geometry shown in Fig. 4. A two-equation model for the turbulence is used [8] in which differential equations for two additional scalar variables, the kinetic energy of turbulence k and its dissipation rate ϵ, are solved. As mentioned earlier, the splitting operations recommended by Issa [5] for the k and ϵ equations are adopted here. To account for the near-wall viscous effects, the conventional wall function method [8] is used in this work.



R_1	27	(mm)
R_2	55	
R_0	90	
R_f	300	
L_f	900	

Fig. 4 Isothermal turbulent flow in an axisymmetric geometry.

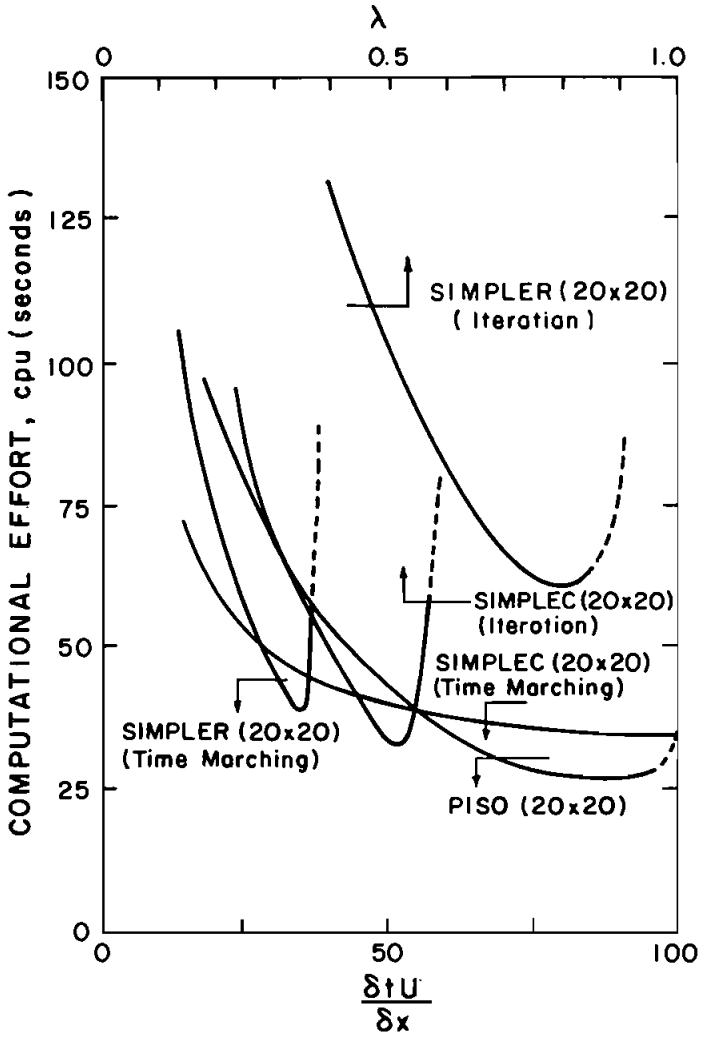


Fig. 5 Comparison of computational effort for the isothermal turbulent flow.

Figure 5 presents the computing effort requirements of the various algorithms and shows trends essentially similar to those observed in test problem 1. The PISO and SIMPLEC time-marching algorithms exhibit the best behavior in terms of the computing effort required, while the SIMPLER time-marching formulation again exhibits reasonable behavior only for small time steps and over a small Δt range.

While the comments pertaining to test problem 1 hold here, it should be remarked that in the PISO algorithm the appropriate splitting of the scalar variable equations (for k and ϵ) into implicit predictor and explicit corrector steps is important in maintaining a low computational effort.

Test Problem 3: Swirling, Reacting Flow in an Axisymmetric Furnace

The geometry of interest is shown in Fig. 6. Methane is injected through the inner nozzle and swirling primary air enters the furnace through the surrounding nozzle. Film cooling air is injected through a port immediately below the outer furnace wall. The inlet velocity of methane is adjusted to satisfy overall stoichiometry at inlet. The inlet tangential velocity of the primary air is characterized by the swirl number S, defined as

$$S = \left. \frac{\int \rho u w r^2 \, dr}{\int \rho u^2 r R_{\text{eff}} \, dr} \right|_{\text{inlet}} \tag{32}$$

The furnace walls are assumed to be adiabatic and impermeable. The diffusion flame and infinite chemistry assumptions (i.e., fuel and oxygen cannot coexist at the same place and at the same time) are invoked. The flow is assumed to be laminar and steady.

With these assumptions, a scalar variable m_{fx} can be defined as

$$m_{fx} = m_{fu} - \frac{m_{ox}}{i} \tag{33}$$

where m_{fx} satisfies the convection-diffusion equation with zero source term. The density ρ is calculated using the ideal gas law, i.e.,

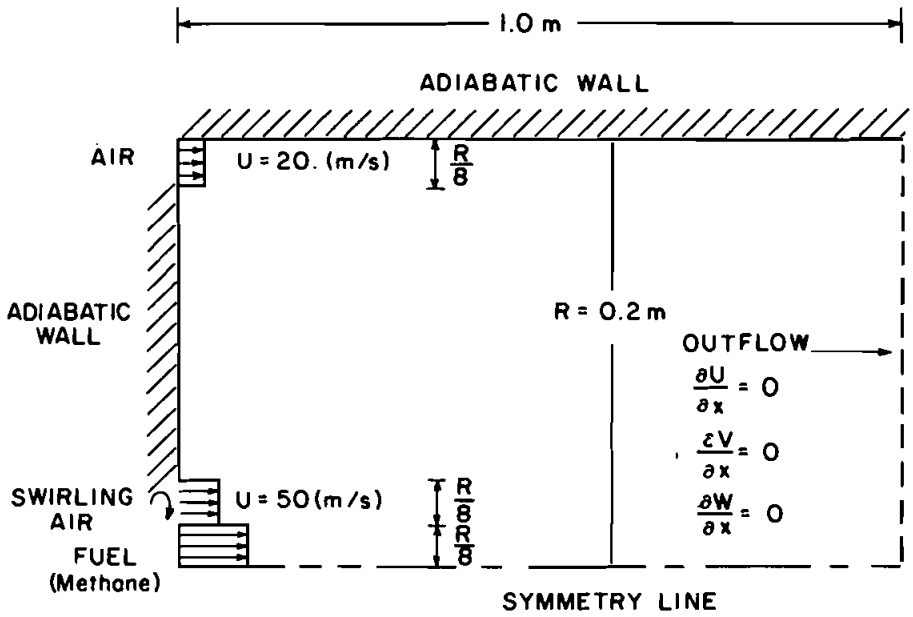


Fig. 6 Reacting swirling flow in an axisymmetric furnace.

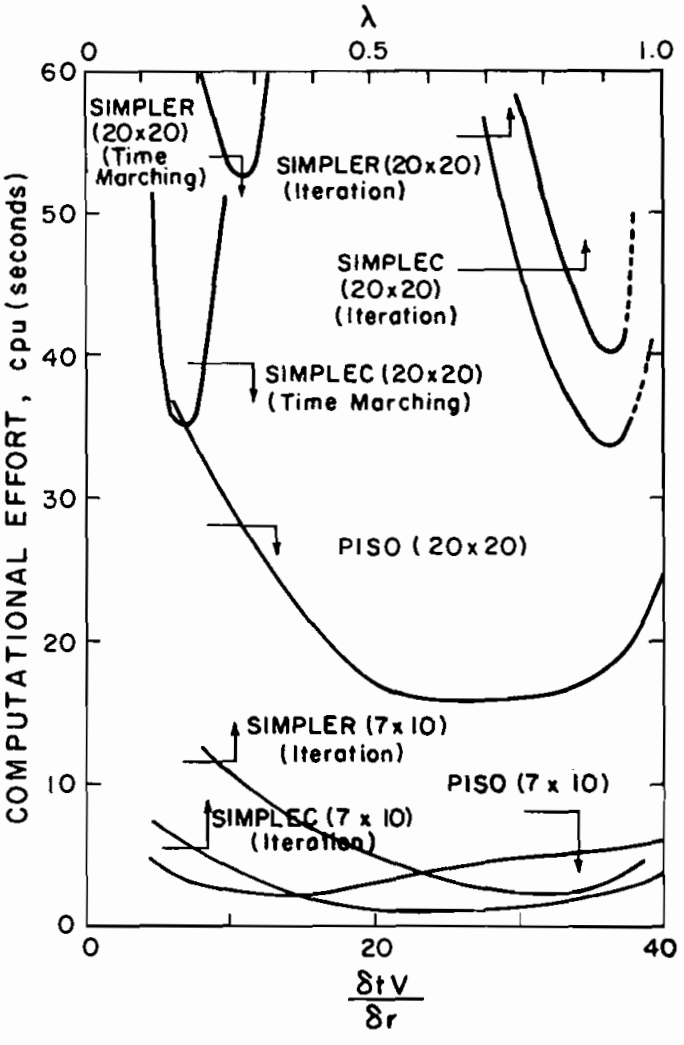


Fig. 7 Comparison of computational effort for reacting swirling flow in a furnace, swirl number $S = 0.2$.

$$\rho = \frac{p/RT}{(m_{fu}/W_{fu}) + (m_{ox}/W_{ox}) + (m_{pr}/W_{pr})} \tag{34}$$

The momentum equations are coupled to the equations for the mass fractions and enthalpy (or temperature) through Eq. (34).

In view of the axisymmetric nature of the problem, $\partial p/\partial\theta = 0$. Thus, the tangential velocity equation has the same form as a scalar variable with the coupling between this variable and the momentum equations explicitly appearing as a source term $\rho w^2/r$ in the radial momentum equation. The strength of this coupling is dictated by the magnitude of the swirl number.

In Fig. 7 results are presented for a low swirl number, $S = 0.2$, and indicate

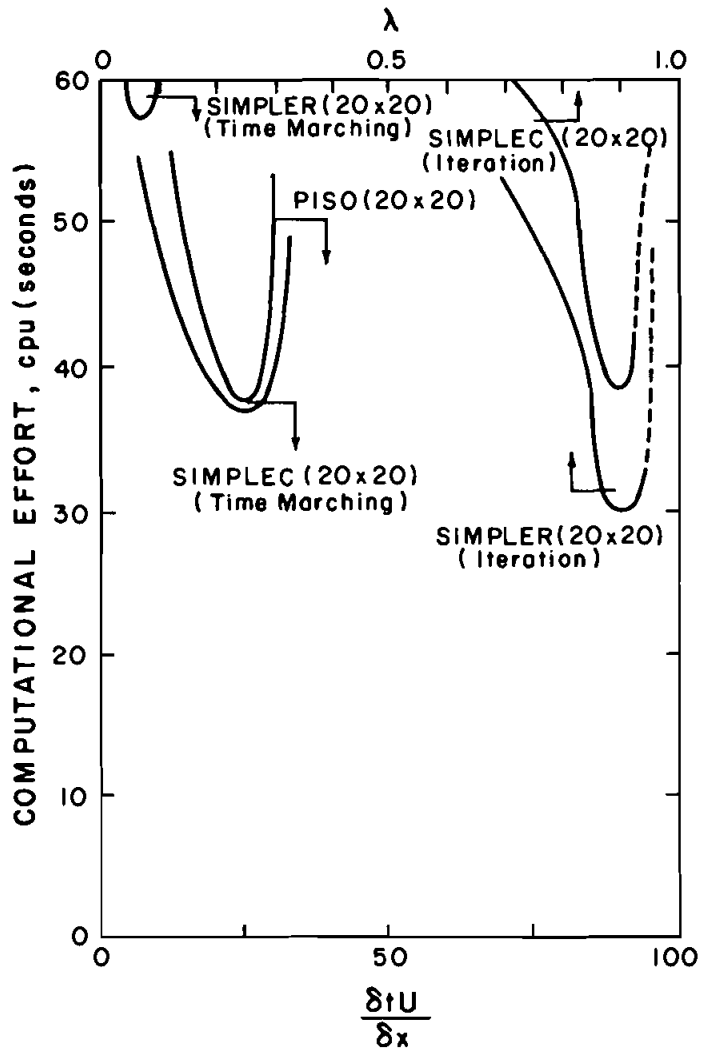


Fig. 8 Comparison of computational effort for reacting swirling flow in a furnace, swirl number $S = 20$.

clearly the superiority of the PISO algorithm for grid sizes of 20 × 20 and 7 × 10. Thus, at low swirl numbers, when the direct coupling between w and the radial momentum equation is weak, PISO as in the earlier two test problems continues to exhibit good convergence properties despite a single predictor step for the scalar variables (w, T, m_{fx}) as recommended by Issa [5] in the two-stage PISO scheme. The coupling between the scalar variables and the momentum equations through Eq. (34) does not appear to have any adverse effect on the favorable performance of the PISO algorithm.

It should be noted that, unlike the earlier two test problems, the SIMPLEC algorithm on the 20 × 20 grid requires greater computational effort than the SIM-

PLER method in the iterative formulation and vice versa in the time-marching formulation. Both algorithms exhibit poor convergence properties compared to the PISO method. While the smaller CPU time with the SIMPLER iterative algorithm (compared to the SIMPLEC method) is in contrast with the observations in the earlier test problems, it is in keeping with the previous argument that consistent superiority of one method over another is unlikely to be observed, since the p-updating scheme in SIMPLER is more accurate but the u and v correction procedure involves a greater approximation than the corresponding operations in the SIMPLEC algorithm.

Figure 8 presents the results for the same problem but at a very high swirl number ($S = 20.0$), where there is a strong coupling between the variable w and the other momentum equations through the source term. The nature of the coupling is different from that at the low swirl number, where the primary coupling is through the density term, and that in test problem 2, where the coupling of k and ϵ to the momentum equations is through the turbulent viscosity μ_t term. Unlike the low swirl number case (Fig. 7), PISO does not exhibit very good convergence behavior; it requires greater computing effort to obtain steady-state solutions than the iterative SIMPLER algorithm and nearly as much effort as the SIMPLEC algorithm. This seems to indicate that when the momentum equations are strongly coupled to a scalar variable (or tangential velocity) the PISO algorithm does not show robust convergence behavior; reasonable solutions are obtained only for small time steps and a small range of Δt. To explain this behavior it should be noted that the velocity correction steps in the PISO algorithm are explicit-type equations and therefore not unconditionally stable. In view of the explicit nature of these equations, it is likely

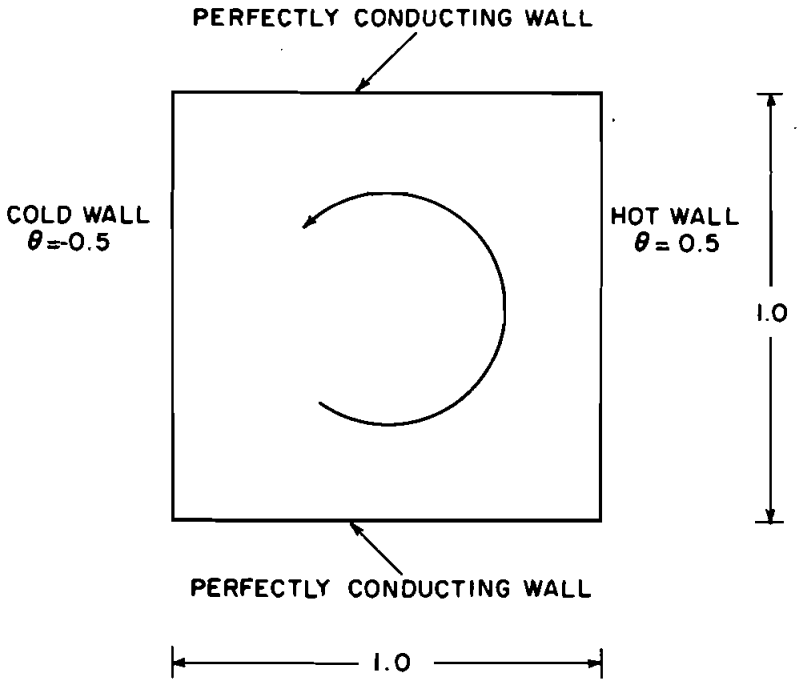


Fig. 9 Free convection in an enclosure.

that PISO tends to become unstable as Δt is increased, leading to a greater CPU time requirement. This view is supported by the fact that to obtain converged solutions with PISO it is necessary to solve the implicit system of pressure equations as accurately as possible (both at the predictor and the corrector steps), otherwise errors in the p solution grow rapidly with time steps. Since the solution to the system of equations is obtained by repeated sweeps of the Thomas algorithm, it is necessary to increase the number of sweeps of the algorithm while solving the p equation. Failure to do so leads to poor results. These conclusions are corroborated in the next test problem, which deals with enclosure convection where the coupling between the scalar variable (temperature) and the momentum equations is solely responsible for the fluid motion.

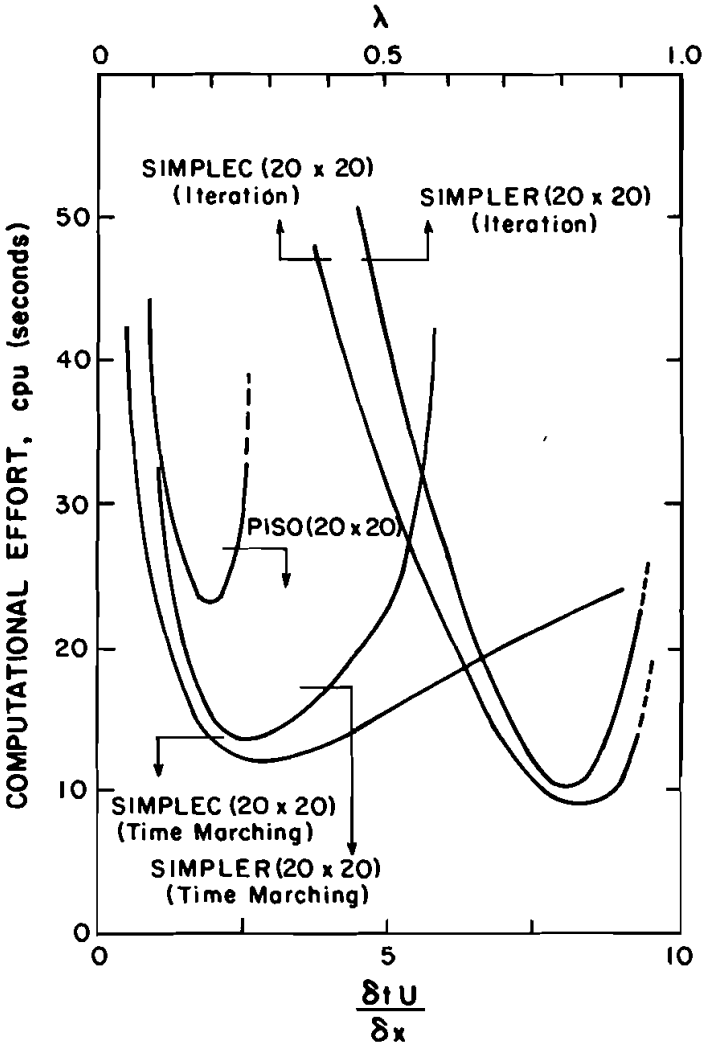


Fig. 10 Comparison of computational effort for free convection in an enclosure.

Test Problem 4: Free Convection in an Enclosure

We consider a square enclosure with isothermal vertical walls and perfectly conducting horizontal walls, as shown in Fig. 9. The flow is assumed to be laminar, two-dimensional, and stationary. The working fluid is taken to be air and is assumed to satisfy the Boussinesq approximation. Results presented are for a Rayleigh number (Ra) of 10^4.

The dimensionless y-momentum equation in this problem contains $Ra\theta$ as a source term, where θ is the dimensionless temperature. As mentioned earlier, this coupling alone is responsible for the convective motion. Results in Fig. 10 indicate, as expected (in view of the explicit-type corrector steps and the inherent stability restrictions associated with them), the greater computing effort necessary with the two-stage PISO algorithm than with the SIMPLER and SIMPLEC methods. The convergence behavior of the SIMPLEC algorithm is slightly better than that of the SIMPLER method.

CONCLUDING REMARKS

A comparative assessment has been made of the convergence behavior of the PISO, SIMPLER, and SIMPLEC algorithms. The following conclusions may be drawn.

1. For problems in which the momentum equation is not coupled to a scalar variable, the PISO algorithm clearly demonstrates robust convergence behavior and, in general, requires less computational effort than the SIMPLER and SIMPLEC methods.
2. For the turbulent flow problem, in which the k-ϵ model for turbulence is invoked, it is necessary to have a two-step scheme (a predictor and a corrector step) for the scalar k and ϵ equations to ensure superior performance by the PISO method.
3. For scalar variables that are weakly linked to the momentum equations, say through the density term, the two-stage PISO algorithm exhibits good convergence behavior compared to the other methods. However, for scalar variables that are strongly linked to the momentum equation, say through the source term as temperature, tangential velocity, etc., PISO does not exhibit robust behavior and gives acceptable results only for small time steps. Furthermore, to obtain converged solutions it is necessary at each time step of the PISO algorithm to obtain accurate solutions of the implicit system of pressure equations. The iterative formulations of the SIMPLER and SIMPLEC methods generally exhibit better behavior in such problems than the two-stage PISO method.
4. It is difficult to ascertain clear superiority of SIMPLER over SIMPLEC or vice versa. In general, the iterative formulations of these methods are more robust than the corresponding time-marching formulations, although exceptions exist. The time-marching SIMPLER algorithm, in particular, exhibits rather inconsistent behavior.

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*, vol. 15, pp. 1787–1806, 1972.
2. G. D. Raithby and G. E. Schneider, Numerical Solution of Problems in Incompressible Fluid Flow; Treatment of the Velocity-Pressure Coupling, *Numer. Heat Transfer*, vol. 2, no. 2, pp. 417–440, 1979.
3. S. V. Patankar, A Calculation Procedure for Two Dimensional Elliptic Situations, *Numer. Heat Transfer*, vol. 14, pp. 409–425, 1985.
4. J. P. Van Doormaal and G. D. Raithby, Enhancements of the SIMPLE Method for Predicting Incompressible Fluid Flows, *Numer. Heat Transfer*, vol. 7, pp. 147–163, 1984.
5. R. I. Issa, Solution of the Implicit Discretized Fluid Flow Equations by Operator Splitting, Mechanical Engineering Rept. FS/82/15, Imperical College, London, 1982.
6. R. I. Issa, A. D. Gosman, and A. P. Watkins, The Computation of Compressible and Incompressible Recirculating Flows by a Non-Iterative Implicit Scheme, *J. Comp. Phys.*, vol. 62, pp. 66–82, 1986.
7. S. V. Patankar, *Numerical Heat Transfer and Fluid Flow*, Hemisphere, Washington, D.C., 1980.
8. B. E. Launder and D. B. Spalding, *Mathematical Models of Turbulence*, Academic Press, New York, 1972.

Received December 23, 1985
Accepted May 29, 1986

Requests for reprints should be sent to S. Acharya.