Pressure Based Algorithms for Multi-Fluid Flow at All Speeds-Part II: Geometric Conservation Formulation

F. Moukalled[1] and M. Darwish[2]
American University of Beirut,
Faculty of Engineering & Architecture,
Mechanical Engineering Department,
P.O.Box 11-0236
Riad El Solh, Beirut 1107 2020
Lebanon

Abstract

This work is concerned with the implementation and testing, within a structured collocated finite-volume framework, of seven segregated algorithms for the prediction of multi-phase flow at all speeds. These algorithms belong to the Geometric Conservation Based Algorithms (GCBA) group in which the pressure correction equation is derived from the constraint equation on volume fractions (i.e. sum of volume fractions equals 1). The pressure correction schemes in these algorithms are based on SIMPLE, SIMPLEC, SIMPLEX, SIMPLEM, SIMPLEST, PISO, and PRIME. The performance and accuracy of these algorithms are assessed by solving, using the single grid method (SG), the prolongation grid method (PG), and the full non-linear multi-grid method (FMG), the following four two-dimensional two-phase flow problems: (i) turbulent upward bubbly flow in a pipe, (ii) turbulent air-particle flow in a pipe, (iii) compressible dusty flow over a flat plate, (iv) and transonic dusty flow in a converging-diverging nozzle. Results are displayed in the form of convergence history plots and tabulated CPU times. The main outcomes of this study are the clear demonstrations of: (i) the capability of all GCBA algorithms to deal with multi-fluid flow situations; (ii) the ability of the FMG method to tackle the added non-linearity of multi-fluid flows; (iii) and the capacity of the GCBA algorithms to predict multi-fluid flow at all speeds.

[1] Author to whom all correspondence should be addressed; email: memouk@aub.edu.lb; [2]email: Darwish@aub.edu.lb

Nomenclature

$A_P^{(k)}$,..	coefficients in the discretized equation for $\phi^{(k)}$.
$B_P^{(k)}$	source term in the discretized equation for $\phi^{(k)}$.
$C_\rho^{(k)}$	coefficient equals to $1/R^{(k)}T^{(k)}$.
$\mathbf{D}_P^{(k)}[\phi^{(k)}]$	the Matrix **D** operator.
$H_P[\phi^{(k)}]$	the H operator.
$\mathbf{H}_P[\mathbf{u}^{(k)}]$	the vector form of the *H* operator.
$\dot{M}^{(k)}$	mass source per unit volume.
P	pressure.
$r^{(k)}$	volume fraction of fluid/phase k.
$RESG_P$	residuals of the volume fraction's constraint equation.
$R^{(k)}$	gas constant for fluid/phase k.
$R_P^{(k)}$	coefficient equals $1/A_P^{(k)}$.
$\mathbf{S}_f$	surface vector.
t	time.
$U_f^{(k)}$	interface flux velocity $\left(\mathbf{v}_f^{(k)}.\mathbf{S}_f\right)$ of fluid/phase k.
$\mathbf{u}^{(k)}$	velocity vector of fluid/phase k.
$u^{(k)}$,$v^{(k)}$,..	velocity components of fluid/phase k.

Greek Symbols

$\rho^{(k)}$	density of fluid/phase k.
$\phi^{(k)}$	general scalar quantity associated with fluid/phase k.
$\Delta_P\left[\phi^{(k)}\right]$	the Δ operator $\left(\Delta_P\left[\phi^{(k)}\right]=\sum_{nb}\phi^{(k)}\right)$.
Ω	cell volume.
δt	time step.

Subscripts

nb	refers to the east, west, … face of a control volume.
NB	refers to the East, West, … neighbors of the main grid point.
P	refers to the P grid point.

Superscripts

C	refers to convection contribution.
D	refers to diffusion contribution.
(k)	refers to fluid/phase k.
$(k)^*$	refers to updated value at the current iteration.
$(k)^\circ$	refers to values of fluid/phase k from the previous iteration.
$(k)'$	refers to correction field of phase/fluid k.
Old	refers to values from the previous time step.

Introduction

In a companion paper [1], the Mass Conservation Based Algorithms (MCBA) derived by Darwish et al. [2] were verified. These algorithms are extensions into multi-fluid flow at all speeds of the pressure-based SIMPLE family originally developed for incompressible single fluid flow [3]. In this paper, the multi-fluid pressure-based Geometric Conservation Based Algorithms (GCBA), also derived in [2] based on the SIMPLE family, are implemented and tested. Since a review of the chronological developments in the SIMPLE algorithm [4] was given in [1], it is deemed unnecessary to be repeated here. Rather attention is directed towards highlighting the differences between the MCBA and the GCBA families.

As detailed in [1], the pressure correction equation in the MCBA family is derived from the overall mass conservation equation and the resulting pressure correction field is used to update the velocity, density, and pressure fields. No correction is applied to the volume fraction fields. However, the volume fraction constraint (i.e. the sum of volume fractions equals 1) is enforced at a later stage by either normalizing the volume fraction fields or calculating the last volume fraction field as $r^n = 1 - \sum_{k \neq n} r^k$.

In the GCBA family the pressure correction equation is derived from the geometric constraint $\left(\sum_k r^k = 1 \right)$ and the volume fraction fields are obtained, as before, by solving the volume fraction equations. However, the volume fraction constraint is enforced through the use of the pressure correction equation by correcting the volume fraction fields in addition to the velocity, pressure, and density fields.

From the above it is clear that the critical difference between the MCBA and GCBA families is in the derivation and role of the pressure correction equation. With this in mind, the objective of this paper is to implement and test seven multi-fluid algorithms from the GCBA

family with their pressure correction schemes based on SIMPLE[4,5], SIMPLEC[6], SIMPLEM[7], SIMPLEX[8], SIMPLEST[9], PISO[10], and PRIME[11].

Since the governing equations and discretization procedure for multi-fluid flow were introduced in [1], they are not repeated here and only a brief description of the GCBA approach, the multigrid strategy, and the GCBA multi-fluid pressure correction equation is given. Then, the capability of the GCBA algorithms to predict multi-fluid flow phenomena at all speeds demonstrated, and their performance characteristics (in terms of convergence history and speed) assessed by solving four two-phase flow problems encompassing dilute and dense gas-solid and bubbly flows in the subsonic, transonic, and supersonic regimes. In addition, the performance of these algorithms is evaluated using (i) a Single Grid approach (SG), (ii) a Prolongation Grid approach (PG), (iii) and a Full Multi-Grid (FMG) approach with a V cycle.

The Geometric Conservation Based Algorithms (GCBA)

The numbers of equations describing an n-fluid flow situation are: n momentum equations, n volume fraction (or mass conservation) equations, a geometric conservation equation, and for the case of compressible flow additional n auxiliary pressure-density relations. Moreover, the variables involved are the n velocity vectors, the n volume fractions, the pressure field, and for a compressible flow an additional n unknown density fields. The n-momentum equations are used to compute the n-velocity fields. The volume fractions are calculated from the volume fraction equations, which mean that the remaining equation i.e. the geometric conservation equation (the volume fractions sum to 1) has to be used in deriving the pressure correction equation. This results in the Geometric Conservation Based Algorithms (GCBA). The sequence of events in the GCBA is as follows:

- Solve the individual mass conservation equations for volume fractions.
- Solve the momentum equations for velocities.

- Solve the pressure correction equation.
- Correct velocity, volume fraction, density, and pressure fields.
- Solve the individual energy equations.
- Return to the first step and repeat until convergence.

The GCBA uses the momentum equations for a first estimate of velocities. However, the volume fractions are calculated without enforcing the geometric conservation equation. Hence, the mass conservation equations of all fluids are used to calculate the volume fractions. As such, the pressure correction equation should be based on the geometric conservation equation and used to restore the imbalance of volume fractions. The errors in the calculated volume fractions are expressed in terms of pressure correction (P'), which is also used to adjust the velocity and density fields as described below.

The Pressure Correction Equation

After solving the continuity equations for the volume fraction fields and the momentum equations for the velocity fields, the next step is to correct the various fields such that the volume fraction fields satisfy the compatibility equation and the velocity and pressure fields satisfy the continuity equations. For that purpose, a guess-and-correct scheme is adopted. Correction is obtained by solving a pressure correction equation derived from the geometric conservation equation. To start the derivation, it is noticed that initially the volume fraction fields denoted by $r^{(k)*}$, do not satisfy the compatibility equation and a discrepancy exists i.e.

$$\mathrm{RESG_P} = 1 - \sum_k \mathrm{r_P^{(k)*}} \tag{1}$$

A change to $\mathrm{r^{(k)*}}$ is sought that would restore the balance. The corrected r values denoted by $\mathrm{r^{(k)}}$ $\left(\mathrm{r^{(k)}} = \mathrm{r^{(k)*}} + \mathrm{r^{(k)'}}\right)$, are such that

$$\sum_k \left(\mathrm{r^{(k)'}}\right) = 1 - \sum_k \left(\mathrm{r^{(k)*}}\right) = \mathrm{RESG_P} \tag{2}$$

Corrections to the volume fraction, $r^{(k)'}$, will be associated with corrections to the velocity, density, and pressure fields, $\mathbf{u}^{(k)'}$, $\rho^{(k)'}$, and P', respectively. Thus, the corrected fields are given as:

$$r^{(k)} = r^{(k)^*} + r^{(k)'}, P = P^o + P', \mathbf{u}^{(k)} = \mathbf{u}^{(k)^*} + \mathbf{u}^{(k)'}, \rho^{(k)} = \rho^{(k)^o} + \rho^{(k)'} \tag{3}$$

The discretized form of the corrected continuity equation of phase (k) can be written as

$$\begin{aligned} &\frac{\left(r_P^{(k)*} + r_P^{(k)'}\right)\left(\rho_P^{(k)\circ} + \rho_P^{(k)'}\right) - \left(r_P^{(k)}\rho_P^{(k)}\right)^{Old}}{\delta t}\Omega_P \\ &+ \Delta_P\left(\left(r^{(k)*} + r^{(k)'}\right)\left(\rho^{(k)\circ} + \rho^{(k)'}\right)\left(\mathbf{u}^{(k)*} + \mathbf{u}^{(k)'}\right).\mathbf{S}\right) = \dot{M}_P^{(k)}\left(r_P^{(k)*} + r_P^{(k)'}\right)\Omega_P \end{aligned} \tag{4}$$

Neglecting second and third order terms (i.e. $r_P^{(k)'}\rho_P^{(k)'}$, $\rho_P^{(k)'}\mathbf{u}^{(k)'}$, $r_P^{(k)'}\mathbf{u}^{(k)'}$, and $r_P^{(k)'}\rho_P^{(k)'}\mathbf{u}^{(k)'}$), its expanded form reduces to:

$$\begin{aligned} &\frac{\left(r_P^{(k)*}\rho_P^{(k)'} + r_P^{(k)'}\rho_P^{(k)\circ}\right)}{\delta t}\Omega_P + \Delta_P\left[\left(r^{(k)*}\rho^{(k)\circ}\mathbf{u}^{(k)'}.\mathbf{S} + r^{(k)*}U^{(k)*}\rho_P^{(k)'} + \rho^{(k)\circ}U^{(k)*}r^{(k)'}\right)\right] \\ &- \dot{M}_P^{(k)}r_P^{(k)'}\Omega_P = -\frac{\left(r_P^{(k)*}\rho_P^{(k)\circ}\right) - \left(r_P^{(k)}\rho_P^{(k)}\right)^{Old}}{\delta t}\Omega_P - \Delta_P\left[\left(r^{(k)*}\rho^{(k)\circ}U^{(k)*}\right)\right] + \dot{M}_P^{(k)}r_P^{(k)*}\Omega_P \end{aligned} \tag{5}$$

From [1], the discretization procedure for the momentum equation yields an algebraic equation of the form:

$$\mathbf{u}_P^{(k)} = \mathbf{H}_P\left[\mathbf{u}^{(k)}\right] - r_P^{(k)}\mathbf{D}_P^{(k)}\nabla_P(P) \tag{6a}$$

where

$$H_P\left[\phi^{(k)}\right] = \frac{\sum_{NB} A_{NB}^{(k)}\phi_{NB}^{(k)} + B_P^{(k)}}{A_P^{(k)}} \tag{6b}$$

and

$$\mathbf{D}_P^{(k)} = \begin{bmatrix} \dfrac{\Omega}{A_P^{u^{(k)}}} & 0 \\ 0 & \dfrac{\Omega}{A_P^{v^{(k)}}} \end{bmatrix} \tag{6c}$$

Then, using (6a) the following equation for $\mathbf{u}_P^{(k)'}$, as a function of P', is obtained

$$\mathbf{u}_P^{(k)'} = \mathbf{H_P}[\mathbf{u}^{(k)'}] - r_P^{(k)*}\mathbf{D}_P^{(k)}\nabla P' - r_P^{(k)'}\mathbf{D}_P^{(k)}\nabla P^o - r_P^{(k)'}\mathbf{D}_P^{(k)}\nabla P' \tag{7}$$

Substituting Eq. (7) into Eq. (5), rearranging, and discretizing one gets

$$r_P^{(k)'} - H_P\left[r^{(k)'}\right] = -R_P^{(k)}\left(\begin{array}{l}\dfrac{r_P^{(k)*}\Omega_P}{\delta t}\rho_P^{(k)'} + \Delta_P\left[r^{(k)*}\rho^{(k)\circ}\left(\begin{array}{l}\mathbf{H}[\mathbf{u}^{(k)'}] - r^{(k)*}\mathbf{D}^{(k)}\nabla P' \\ - r^{(k)'}\mathbf{D}^{(k)}\nabla P'\end{array}\right).\mathbf{S} + r^{(k)*}U^{(k)*}\rho^{(k)'}\right] \\ + \dfrac{\left(r_P^{(k)*}\rho_P^{(k)\circ}\right) - \left(r_P^{(k)}\rho_P^{(k)}\right)^{Old}}{\delta t}\Omega_P + \Delta_P\left[\left(r^{(k)*}\rho^{(k)\circ}U^{(k)*}\right)\right] - \dot{M}_P^{(k)}r_P^{(k)*}\Omega_P\end{array}\right) \quad (8)$$

where $R_P^{(k)} = 1/A_P^{(k)}$.

Neglecting the correction to neighboring cells, equation (8) reduces to:

$$r_P^{(k)'} = -R_P^{(k)}\left(\begin{array}{l}\dfrac{r_P^{(k)*}\Omega_P}{\delta t}\rho_P^{(k)'} + \Delta_P\left[r^{(k)*}\rho^{(k)\circ}\left(\begin{array}{l}\mathbf{H}[\mathbf{u}^{(k)'}] - r^{(k)*}\mathbf{D}^{(k)}\nabla P' \\ - r^{(k)'}\mathbf{D}^{(k)}\nabla P'\end{array}\right).\mathbf{S} + r^{(k)*}U^{(k)*}\rho^{(k)'}\right] \\ + \dfrac{\left(r_P^{(k)*}\rho_P^{(k)\circ}\right) - \left(r_P^{(k)}\rho_P^{(k)}\right)^{Old}}{\delta t}\Omega_P + \Delta_P\left[\left(r^{(k)*}\rho^{(k)\circ}U^{(k)*}\right)\right] - \dot{M}_P^{(k)}r_P^{(k)*}\Omega_P\end{array}\right) \quad (9)$$

Substituting this equation into the geometric conservation equation and expressing density correction in terms of pressure correction (i.e. $\rho^{(k)'} = C_\rho^{(k)}P'$), the pressure correction equation is obtained as

$$\sum_k\left\{-R_P^{(k)}\left(\begin{array}{l}\dfrac{r_P^{(k)*}\Omega_P C_\rho^{(k)}}{\delta t}P_P' + \Delta_P\left[r^{(k)*}U^{(k)*}C_\rho^{(k)}P'\right] + \\ \Delta_P\left[r^{(k)*}\rho^{(k)\circ}\left(\mathbf{H}[\mathbf{u}^{(k)'}] - r^{(k)*}\mathbf{D}^{(k)}\nabla P' - r^{(k)'}\mathbf{D}^{(k)}\nabla P'\right).\mathbf{S}\right] \\ + \dfrac{\left(r_P^{(k)*}\rho_P^{(k)\circ}\right) - \left(r_P^{(k)}\rho_P^{(k)}\right)^{Old}}{\delta t}\Omega_P + \Delta_P\left[\left(r^{(k)*}\rho^{(k)\circ}U^{(k)*}\right)\right]\end{array}\right)\right\} = RESG_P \quad (10)$$

If the $\mathbf{H}\left[\mathbf{u}^{(k)'}\right]$ term in the above equation is retained, there will result a pressure correction equation relating the pressure correction value at a point to all values in the domain. To facilitate implementation and reduce cost, simplifying assumptions related to this term have been introduced. Depending on these assumptions, different algorithms are obtained. A summary of the various GCBA algorithms (GCBA-SIMPLE, GCBA-SIMPLEC, GCBA-PISO,…) used in this work was accorded a full length paper to which interested reader is referred [2]. Moreover, the discretization of the above equation yields

$$A_P^{P'}P_P' = \sum_{NB}A_{NB}^{P'}P_{NB}' + B_P^{P'} \quad (11)$$

After calculating the pressure correction field, $\mathbf{u}_P^{(k)'}, \rho_P^{(k)'}, \text{and } r_P^{(k)'}$ are obtained using the following equations

$$\begin{aligned}
\mathbf{u}_P^{(k)'} &= -r^{(k)}\mathbf{D}_P^{(k)}\nabla_P\left(P'\right) \\
\rho^{(k)'} &= C_\rho^{(k)}P' \\
r_P^{(k)'} &= -R_P^{(k)}\left(\frac{r_P^{(k)}\Omega_P}{\delta t}\rho_P^{(k)'} + \Delta_P\left[\left(r^{(k)}\rho^{(k)}\mathbf{u}^{(k)'}\right).\mathbf{S} + r^{(k)}U^{(k)}\rho^{(k)'}\right]\right)
\end{aligned} \tag{12}$$

The Multi-Grid and Prolongation grid Strategies

The multi-grid algorithm [12,13] adopted in this work can be summarized as follows. Starting with the fine mesh, the coarser grid cells are generated through agglomeration of four finer grid cells, two in each direction. On the other hand, a finer grid is obtained by subdividing the coarser grid control volume into four control volumes, again two in each direction. With the FMG cycle, the algorithm starts at the coarsest level, where the solution is first computed; this solution is interpolated onto the next finer mesh, where it is used as initial guess. This stage is called the prolongation stage. Then iterations are performed on the fine mesh and the solution is transferred back to the coarser mesh by applying a restriction operator. In order to obtain the same approximation on each level, a forcing term is added to the discrete conservation equations on the coarser grid. This term represents the truncation error on the coarse grid with respect to the fine grid. After performing a number of iterations on the coarse mesh, the solution is transferred back to the finer mesh in the form of a correction and a number of iterations are performed on the finer grid to smooth the fields. This process is continued until a converged solution on the fine mesh is obtained. Then the solution is extrapolated to correct the finer mesh fields, followed by a number of smoother iterations on the finer mesh and the process repeated until convergence is reached on the desired finest mesh. This strategy has been applied to both incompressible and compressible supersonic multi-fluid flows and good savings have been realized as will be shown in the results section.

In addition to the FMG strategy, the PG approach is also tested. This approach differs from the FMG method in that the solution moves in one direction from the coarse to the fine grid with the initial guess on level n+1 obtained by interpolation from the converged solution on level n. As such, the acceleration over the SG method obtained with this approach is an indication of the effect of initial guess on convergence.

Results and Discussion

To demonstrate the capability of the GCBA in predicting multiphase flow at all speeds and to assess their relative performance, solutions to the following four two-dimensional two-phase flow problems are presented: (i) turbulent upward bubbly flow in a pipe, (ii) turbulent air-particle flow in a pipe, (iii) compressible dusty flow over a flat plate, (iv) and transonic dusty flow in a converging-diverging nozzle. Results are presented in terms of CPU-time and residual history plots for a number of grids using the single grid, the prolongation grid, and the full non-linear multi-grid method. The residual of a variable $\phi^{(k)}$ at the end of an outer iteration is defined as:

$$RES_{\phi}^{(k)} = \sum_{c.v} \left| A_P \phi_P^{(k)} - \sum_{NB} A_{NB} \phi_{NB}^{(k)} - B_p^{(k)} \right| \tag{13}$$

For global mass conservation, the imbalance in mass is defined as:

$$RES_C = \sum_k \sum_{c.v.} \left| \frac{\left(r_P^{(k)} \rho_P^{(k)}\right) - \left(r_P^{(k)} \rho_P^{(k)}\right)^{Old}}{\delta t} \Omega - \Delta_P \left[r^{(k)} \rho^{(k)} \mathbf{u}^{(k)} . \mathbf{S} \right] - r^{(k)} \dot{M}^{(k)} \right| \tag{14}$$

All residuals are normalized by their respective inlet fluxes. Computations are terminated when the maximum normalized residuals of all variables drop below a very small number ε_s. Unless otherwise stated, the HR SMART scheme is used in all computations reported in this study. For a given problem, all results are generated starting from the same initial guess. Since a detailed comparison between current numerical results and available experimental/theoretical data was performed in [1], it is not repeated here.

Problem 1: Turbulent upward bubbly flow in a pipe

This problem is concerned with the prediction of radial phase distribution in turbulent upward air-water flow in a pipe [1,14-22] . The case considered reproduces numerically the experimental data reported by Seriwaza et al [14] for which the Reynolds number based on superficial liquid velocity and pipe diameter is $8x10^4$, the inlet superficial gas and liquid velocities are 0.077 and 1.36 m/s, respectively, and the inlet void fraction is $5.36x10^{-2}$ with no slip between the incoming phases. Moreover, the bubble diameter is taken as 3 mm [22], while the fluid properties are assigned the values $\rho^{(c)}$=1000 Kg/m^3, $\rho^{(d)}$=1.23 Kg/m^3, and $\nu_l^{(c)}= 10^{-6}$ m^2/s.

Calculations are performed using the SG, PG, and FMG strategies for all algorithms. Results are displayed in the form of (i) total mass residuals summed over both phases as a function of outer iterations (Fig. (1)), and (ii) normalized CPU times (Table 1) needed for the maximum normalized residuals of all variables and for all phases to drop below $\varepsilon_s=10^{-6}$.

As can be seen from Fig. 1, it is possible to obtain converged solutions to the desired level with all algorithms. With the exception of PISO (Fig. 1(a)), the convergence characteristics of all algorithms (Figs. 1(b)-(g)) are very similar. The PG method reduces, on average, the number of iterations in comparison with the SG method by about 20%. On the other hand, the FMG method results in a 50% reduction in the number of outer iterations. The use of 3 and 4 levels for both the PG and FMG methods does not seem to have any effect on convergence acceleration for all algorithms except PISO, for which the use of 4 levels with the FMG method increases the number of outer iterations considerably and results in a kind of oscillations (Fig. 1(a)). The convergence histories of all algorithms with the FMG method on 3 levels presented in Fig. 1(h) confirm once more the aforementioned observations.

Table 1 reports on the normalized CPU-times (i.e. CPU-time divided by the time needed by SIMPLE on the coarsest grid) required by the different algorithms using the various

methodologies to decrease the solution residuals to the desired level. For the SG method, the CPU-times on two different grids of sizes 48x16 and 96x32 C.V. are presented. The PG and FMG solutions are for a grid of 96x32 C.V. using 3 (96x32, 48x16, and 24x8 C.V.) and 4 (96x32, 48x16, 24x8, and 12x4 C.V.) grid levels.

As expected, the CPU effort increases for all algorithms and methodologies with increasing the grid size. With the SG method, the variations in the normalized CPU times among algorithms decrease from a maximum of 19 % to 6.7% with increasing the grid density. In both cases however, SIMPLE is the cheapest while PRIME followed by SIMPLEM are the most expensive. On the dense grid, the use of the PG method with 3 grid levels reduces, on average, the computational effort by about 14.62% as compared to the SG method. The effect of employing four grid levels is seen to be marginal. In both cases the CPU times of the various algorithms is within 7.5% from each others. The average decrease in computational time with the FMG method using three grid levels is 40.89% and 27.95% as compared to the SG and PG method, respectively. The performance of PISO on the 4 levels FMG method is highly unexpected necessitating higher computational effort than the SG method and may be caused by the additional explicitness introduced by the PRIME step. Moreover, with the FMG method, the least computational effort is obtained with SIMPLEC, which is slightly less expensive than SIMPLE. Excluding PISO, the most expensive algorithm with the FMG is PRIME, which requires about 27% more time than SIMPLE.

Problem 2: Turbulent air-particle flow in a vertical pipe

Here, the upward flow of a dilute gas-solid mixture in a vertical pipe is simulated [23-25].The experimental results of Tsuji et al [23] are replicated for the case of an air Reynolds number, based on the pipe diameter (of value 30.5 mm), of $3.3x10^4$ and a mean air inlet velocity of 15.6 m/s using particles of diameter 200 μm and density 1020 Kg/m^3. In the computations, the mass-loading ratio at inlet is considered to be 1 with no slip between the fluids [1].

The problem is solved using the SIMPLE, SIMPLEC, SIMPLEX, and SIMPLEST multi-fluid algorithms and the SG, PG, and FMG solution methods. As in the previous problem, results are displayed in the form of (i) total mass residuals summed over both phases as a function of outer iterations (Fig. (2)), and (ii) normalized CPU times (Table 2) needed for the maximum normalized residuals of all variables and for all phases to drop below $\varepsilon_s=10^{-6}$. For the SG method, the CPU-times on two different grids of sizes 48x20 and 96x40 C.V. are presented. The PG and FMG solutions are for a grid of size 96x40 C.V. using 3 (96x40, 48x20, and 24x10 C.V.) and 4 (96x40, 48x20, and 24x10, and 12x5 C.V.) grid levels.

Mass residual plots presented in Fig. 2 indicate a similar convergence behavior for all four algorithms with very close number of outer iterations to achieve the desired level of convergence (i.e. within 50 outer iterations). It is hard to see any noticeable difference between the 3 and 4 levels with both the PG and FMG methods. The decrease in the number of iterations with the PG method over the SG method is smaller than the decrease obtained with bubbly flows. This lower effectiveness of the PG method is due to the following reason. In solving the problem, it is noticed that the initial guess greatly affects the convergence history and time required to reduce residuals to the desired level. Except when solving on the finest mesh with the SG method, the initial guess used for the velocity field is $u^{(c)}=u^{(d)}=1$ m/s. The use of this initial value with the SG method on the finest mesh greatly increased the CPU effort needed over the one needed when starting with an initial field of $u^{(c)}=u^{(d)}=15.6$ m/s. To reduce cost, the latter initial guess is used. For this reason the mass residuals start from somehow a lower value than expected and the PG method appears to be less effective. The FMG method reduces the number of outer iterations by about 53% over the SG method, which indicates a good capability to deal with the added non-linearity of multiphase flows.

On the coarsest grid, the time required by the various algorithms using the SG method varies widely (Table 2). As compared to SIMPLE, the SIMPLEC, SIMPLEX, and SIMPLEM

algorithms require 26%, 28%, and 45% additional computational effort, respectively. As the grid size is increased to 96x40 C.V., the normalized CPU times required by the various algorithms increase, however the differences in CPU times among them decrease to a maximum of 11.9% with SIMPLEM. For the reasons stated above, the PG method on both 3 and 4 grid levels does not have any effects on the convergence rate. In fact for some of the algorithms it slightly increases the computational time. However, it brings the computational effort of the algorithms closer to each others with a spread of maximum 6.7%, again associated with SIMPLEM. On the other hand, the FMG method does reduce the computational cost. The average reduction over the SG method is 36.2% and 33.2% using 3 and 4 grid levels, respectively. The use of 4 grid levels increases the CPU time of all algorithms except SIMPLEM. However it reduces the variation in the CPU times among algorithms from 23.49% to 12.17%. The least computational effort is accomplished with SIMPLE while SIMPLEM is the most expensive with all methods (23.49% more expensive than SIMPLE with the FMG on 3 levels).

Problem 3: Compressible dilute air-particle flow over a flat plate

The problem deals with predicting the features of a two-fluid boundary layer [26-28]. In the simulation, the particle diameter, particle Reynolds number, material density, Prandtl number, and mass load ratio are set to: 10 μm, 10, 1766 kg/m^3, 0.75, and 1 respectively. The wall boundary is treated as a no-slip boundary for the gas phase (i.e. both components of the gas velocity are set to zero), and as a slip boundary condition for the particles phase (i.e. the normal fluxes are set to zero). Results obtained [1] are in excellent agreement with numerical solutions reported by Thevand et al. [28].

As in test 1, the problem is solved using all multi-fluid algorithms and the SG, PG, and FMG solution methods. Results are displayed in the form of total mass residuals (Fig. 3) and normalized CPU time (Table 3) with $\varepsilon_s=10^{-6}$. For the SG method, the CPU-times on two

different grids of sizes 52x24 and 104x48 C.V. are presented. The PG and FMG solutions are for a grid of size 104x48 C.V. using 3 (104x48, 52x24, and 26x12 C.V.) and 4 (104x48, 52x24, and 26x12, and 13x6 C.V.) grid levels.

Plots presented in Fig. 3 indicate that it is possible to get a converged solution to the desired level with all algorithms. As depicted in figure 3(a), PISO requires the least number of outer iterations. This, however, is not associated with the lowest computational effort due to the higher cost per iteration in comparison with other algorithms. The convergence characteristics of SIMPLE (Fig. 3(b)), SIMPLEC (Fig. 3(c)), SIMPLEM (Fig. 3(d)), and SIMPLEX (Fig. 3(g)) are very similar, requiring nearly the same number of outer iterations with the SG, PG, and FMG methods. The number of iterations required by PRIME (Fig. 3(f)) with the SG method is higher than SIMPLEST (Fig. 3(e)). With the FMG method however, the performance of the two algorithms is very close with that of PRIME being slightly better. In general, the use of the PG method reduces the number of outer iterations, as compared to the SG method, by over 40% with all algorithms whereas the use of the FMG method reduces it by over 64%. Fig. 3(h) indicates that when using the FMG method on 3 levels, PISO requires the lowest number of iterations followed by SIMPLEM, SIMPLEC, SIMPLEX, SIMPLE, PRIME, and SIMPLEST. Moreover, the number of iterations needed by SIMPLEST and PRIME is very close and it is nearly double that needed by SIMPLE. It should be clarified that the displayed numbers of iterations represent those needed for the mass residuals to be reduced to the desired level. The CPU-times however represent the computational effort needed to reduce the maximum residuals to the desired level. In some cases, even though the mass residuals may become below the desired value, other residuals could still be above that value. This is why, for example, the CPU time needed by SIMPLE is lower than that needed by SIMPLEC (Table 3) even though Fig. 3(h) indicates that the number of iterations needed by SIMPLEC to reduce the mass residuals to below ε_s is lower.

As depicted in Table 3, The SIMPLE algorithm is the cheapest to use with all methods and over all grids. Moreover, the CPU times consumed by the various algorithms using the SG method on the coarsest grid vary widely. Following SIMPLE, the SIMPLEC algorithm is the cheapest to use (requiring only 4% more CPU time than SIMPLE) and PRIME the most expensive, requiring 113% more CPU time than SIMPLE. On the 104X48 grid system, the relative performance of the algorithms remains unchanged however their performance relative to SIMPLE further deteriorates. In this case, SIMPLEC and PRIME require 6.91% and 171.1% more time than SIMPLE. The PG method reduces, on average, the CPU-time by 27.23% and 27.27% with 3 and 4 grid levels, respectively. Moreover, the additional CPU times required by the various algorithms over SIMPLE vary from 4.84% to 71.79% for the 3 grid levels and from 3.71% to 71.41% for the 4 grid levels with PRIME being consistently the most expensive followed by SIMPLEST. The FMG method reduces, on average, the CPU time as compared to the SG method by 65.87% and 65.45% over the 3 and 4 grid levels, respectively. This is equivalent to saying that the FMG method is about 3.5 times faster than the SG method. With the PG method, the use of 3 or 4 grid levels has marginal effect on solution acceleration for all algorithms. However, this is not the case with the FMG method where the use of 4 grid levels noticeably increases the computational cost of all algorithms except PRIME.

Problem 4: Inviscid transonic dusty flow in a converging-diverging nozzle

The last test considered deals with the prediction of supersonic dilute air-particle flow in an axi-symmetric converging-diverging rocket nozzle [29-34]. The physical quantities employed are similar to those used in [32]. The gas stagnation temperature and pressure at inlet to the nozzle are 555 ^{o}K and 10.34×10^{5} N/m^{2}, respectively. The specific heat for the gas and particles are 1.07×10^{3} J/Kg^{o}K and 1.38×10^{3} J/Kg^{o}K, respectively, and the particle density is 4004.62 kg/m^{3}. With a zero inflow velocity angle, the fluid is accelerated from subsonic to

supersonic speed in the nozzle. The inlet velocity and temperature of the particles are presumed to be the same as those of the gas phase. Results generated [1] are in excellent agreement with published results reported in [32] and others using different methodologies.

To compare the relative performance of the multi-fluid algorithms, the problem is solved via the PG method using the SIMPLE, SIMPLEC, SIMPLEM, and SIMPLEX algorithms over three different grids of sizes 47x20, 94x40, and 188x80 C.V. for a particle radius of size 1 μm. As before, results are displayed in the form of total mass residuals (Fig. 4) and normalized CPU times (Table 4) with ε_s set to 10^{-5}. As shown in Fig. 4, all algorithms require almost the same number of iterations with the exception of SIMPLE on the 94x40 grid, which requires a larger number of iterations than on the finest mesh. Excluding that case, the convergence histories of all algorithms are nearly identical. In terms of computational effort, the normalized CPU-times presented in Table 4 indicate that on the coarsest and finest meshes, SIMPLE is the most efficient algorithm (7% less expensive than SIMPLEC on the dense grid) and SIMPLEM the most expensive (43% more expensive than SIMPLE on the dense grid). On the other hand, SIMPLEX is 11% more expensive than SIMPLE on the fine mesh. On the 94x40 grid, SIMPLE is the most expensive (13.07 % more expensive than SIMPLEC) with the remaining algorithms retaining their relative performance.

Closing Remarks

Seven multiphase flow algorithms belonging to the pressure-based GCBA family were implemented and tested using a single grid, a prolongation grid, and a full non-linear multi-grid method. Solving a variety of two-dimensional two-phase flow problems assessed the performance and accuracy of these algorithms. Results obtained demonstrated the capability of all algorithms to deal with multi-fluid flow situations and to predict multi-fluid flow at all speeds. The PG and FMG methods accelerated the convergence rate for all algorithms. The

FMG method, however, was found to be more efficient and capable of tackling the added non-linearity of laminar and turbulent multi-fluid flows.

References

1. Moukalled, F. and Darwish, M.,"Pressure Based Algorithms for Multi-Fluid Flow at All Speeds-Part I: Mass Conservation Formulation," Numerical Heat Transfer, Part B, (in print).
2. Darwish, M., Moukalled, F., and Sekar, B." A Unified Formulation of the Segregated Class of Algorithms for Multi-Fluid Flow at All Speeds," Numerical Heat Transfer, Part B, vol.40, no. 2, pp. 99-137, 2001.
3. Moukalled, F. and Darwish, M.,"A Unified Formulation of the Segregated Class of Algorithms for Fluid Flow at All Speeds," Numerical Heat Transfer, Part B, vol. 37, No 1, pp. 103-139, 2000.
4. Patankar, S.V.,Numerical Heat Transfer and Fluid Flow, Hemisphere, N.Y., 1981.
5. Patankar , S.V. and Spalding, D.B., "A Calculation Procedure for Heat, Mass and Momentum Transfer in Three Dimensional Parabolic Flows", International Journal of Heat and Mass Transfer, vol. 15, pp. 1787, 1972.
6. Van Doormaal, J. P. and Raithby, G. D."Enhancement of the SIMPLE Method for Predicting Incompressible Fluid Flows," Numerical Heat Transfer, vol. 7, pp. 147-163, 1984.
7. Acharya, S. and Moukalled, F., "Improvements to Incompressible Flow Calculation on a Non-Staggered Curvilinear Grid," Numerical Heat Transfer, Part B, vol. 15, pp. 131-152, 1989.
8. Van Doormaal, J. P. and Raithby, G. D."An Evaluation of the Segregated Approach for Predicting Incompressible Fluid Flows," ASME Paper 85-HT-9, Presented at the National Heat Transfer Conference, Denver, Colorado, August 4-7, 1985.
9. Spalding D. B. 'Mathematical Modelling of Fluid Mechanics, Heat Transfer and Mass Transfer Processes,' Mech. Eng. Dept., Rept. HTS/80/1, Imperial College of Science,

Technology and Medecine, London, 1980.

10. Issa, R.I.,"Solution of the Implicit Discretized Fluid Flow Equations by Operator Splitting," Mechanical Engineering Report, FS/82/15, Imperial College, London, 1982.
11. Maliska, C.R. and Raithby, G.D.,"Calculating 3-D fluid Flows Using non-orthogonal Grid," Proc. Third International Conference on Numerical Methods in Laminar and Turbulent Flows, Seattle, pp. 656-666, 1983.
12. Ferziger, J.H. and Peric, M., Computational Methods for Fluid Dynamics, Springer-Verlag, Berlin Heidelberg, 1996.
13. Thompson, C.P. and Lezeau, P."Application of the Full Approximation Storage Method to the Numerical Simulation of Two-Dimensional Steady Incompressible Viscous Multifluid Flows," International Journal for Numerical Methods in Fluids, vol. 28, no. 8, pp. 1217-1239, 1998.
14. Serizawa, A., Kataoka, I., and Michiyoshi, I.," Phase Distribution in Bubbly Flow," Data set No. 24, Proceedings of the Second International Workshop on Two-Phase Flow Fundamentals, Rensselaer Polytechnic Institute, Troy, NY, 1986.
15. Wang, S.k., Lee, S-j, Jones, Jr., O.C., and Lahey, Jr., R.T.,"3-D Turbulence structure and Phase Distribution Measurements in Bubbly Two-Phase Flows," International Journal of Multiphase Flow, vol. 13, no. 3, 1987.
16. Antal, S.P., Lahey,R.T., and Flaherty, J.E.,"Analysis of Phase Distribution in Fully Developed Laminar Bubbly Two-Phase Flows," International Journal of Multiphase Flow, vol. 17, no. 5, pp. 635-652, 1991.
17. Lahey, R.T., Lopez de Bertodano, M., and Jones, O.C.,"Phase Distribution in Complex Geometry Ducts," Nuclear Engineering Design, vol. 141, p. 177, 1993.
18. Lopez de Bertodano, M., Lee, S-J., Lahey, Jr., R.T., and Drew, D.A.,"The Prediction of Two-Phase Turbulence and Phase Distribution Phenomena Using a Reynolds Stress

Model," Journal of Fluids Engineering, vol. 112, pp. 107-113, 1990.

19. Lopez de Bertodano, M., Lahey, Jr., R.T., and Jones, O.C.,"Development of a k-ε Model for Bubbly Two-Phase Flow," Journal of Fluids Engineering, vol. 116, pp. 128-134, 1994.
20. Lopez de Bertodano, M., Lahey, Jr., R.T., and Jones, O.C.,"Phase Distribution in Bubbly Two-Phase Flow in Vertical Ducts," International Journal of Multiphase Flow, vol. 20, no. 5, pp. 805-818, 1994.
21. Nakoryakov, V.E., Kashinsky, O.N., Randin, V.V., and Timkin, L.S.,"Gas-Liquid Bubbly Flow in Vertical Pipes," Journal of Fluids Engineering, vol. 118, pp. 377-382, 1996.
22. Boisson, N. and Malin, M.R.,"Numerical Prediction of Two-Phase Flow in Bubble Columns," International Journal for Numerical Methods in Fluids, vol. 23, pp. 1289-1310, 1996.
23. Tsuji, Y., Morikawa, Y., and Shiomi, H.,"LDV Measurements of an Air-Solid Two-Phase Flow in a Vertival Pipe," Journal of Fluid Mechanics, vo. 139, pp. 417-434, 1984.
24. Adeniji-Fashola, A. and Chen, C.P.,"Modeling of Confined Turbulent Fluid-Particle Flows Using Eulerian and Lagrangian Schemes,"International Journal of Heat and Mass transfer, vol.33, pp. 691-701, 1990.
25. Naik, S. and Bryden, I.G.,"Prediction of Turbulent Gas-Solids Flow in Curved Ducts Using The Eulerian-Lagrangian Method," International Journal for Numerical Methods in Fluids, vol. 31, pp. 579-600, 1999.
26. Osiptsov, A.N.,"Structure of the Laminar Boundary Layer of a Disperse Medium on a Flat Plate," Fluid Dynamics, vol. 15, pp. 512-517, 1980.
27. Wang, B.Y., and Glass, I.I.,"Compressible Laminar Boundary Layer Flows of a Dusty Gas Over a Semi-Infinite Flat Plate," Journal of Fluid Mechanics, vol. 186, pp. 223-241, 1988.

28. Thevand, N., Daniel, E., and Loraud, J.C.,"On high-Resolution Schemes for Solving Unsteady Compressible Two-Phase Dilute Viscous Flows," International Journal of Numerical Methods in Fluids, vol. 31, pp. 681-702, 1999.
29. Chang, I.S.,"One and Two-Phase Nozzle Flows," AIAA J., vol. 18, pp. 1455-1461, 1980.
30. Ishii, R., Umeda, Y., and Kawasaki, K.,"Nozzle Flows of Gas-Particle Mixtures," Physics of Fluids, vol. 30, No. 3, pp. 752-760, 1987.
31. Hwang, C.J. and Chang, G.C.,"Numerical Study of Gas-Particle Flow in a Solid Rocket Nozzle," AIAA Journal, vol. 26, no. 6, pp. 682-689, 1988.
32. Chang, H.T., Houng, L.W., and Chien, L.E.,"Application of Flux-Vector-Splitting Scheme to a Dilute Gas-Particle JPL Nozzle Flow," International Journal for Numerical Methods in Fluids, vol. 22, pp. 921-935, 1996.
33. Mehta, R.C. and Jayachandran, T.,"A Fast Algorithm to Solve Viscous Two-Phase Flow in an Axisymmetric Rocket Nozzle," International Journal for Numerical Methods in Fluids, vol. 26, pp. 501-517, 1998.
34. Cuffel, R.F., Back, L.H., and Massier, P.F.,"Transonic Flowfield in a Supersonic Nozzle with Small Throat Radius of Curvature," AIAA Journal, vol. 7, pp. 1364-1366, 1969.

Figure Captions

Fig. 1 (a)-(g) Convergence histories of the SG, PG, and FMG methods on the finest grid, and (h) convergence histories of the various algorithms on the finest mesh using the FMG method for turbulent upward bubbly flow in a pipe.

Fig. 2 Convergence histories of the (a) SIMPLE, (b) SIMPLEC, (c) SIMPLEST, and (d) SIMPLEX algorithms using the SG, PG, and FMG methods on the finest mesh for turbulent air-particle flow in a pipe.

Fig. 3 (a)-(g) Convergence histories of the SG, PG, and FMG methods on the finest grid, and (h) convergence histories of the various algorithms on the finest mesh using the FMG method for dusty gas flow over a flat plate.

Fig. 4 Convergence histories of the (a) SIMPLE, (b) SIMPLEC, (c) SIMPLEM, and (d) SIMPLEX algorithms using the SG method for dusty gas flow in a converging-diverging nozzle.

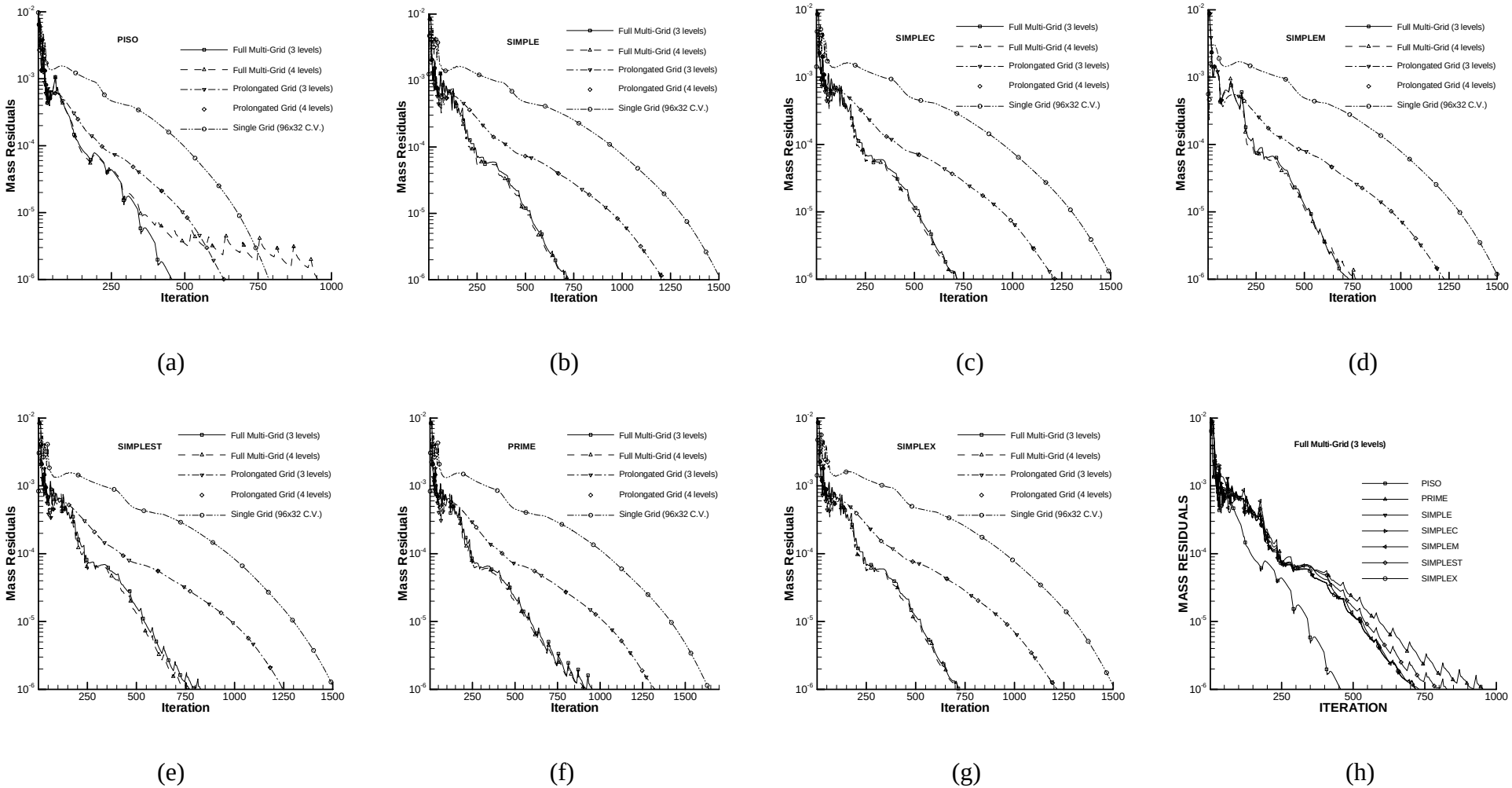
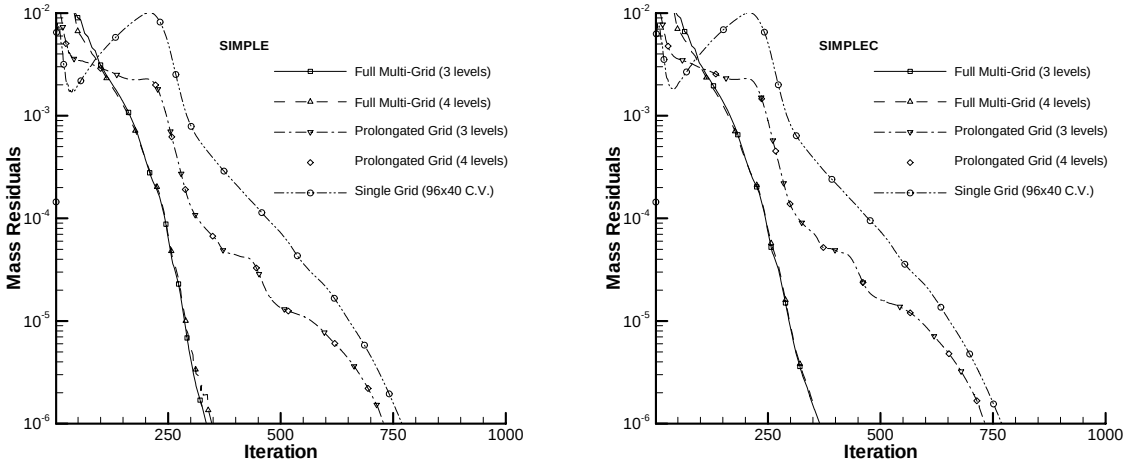


Fig. 1 (a)-(g) Convergence histories of the SG, PG, and FMG methods on the finest grid, and (h) convergence histories of the various algorithms on the finest mesh using the FMG method for turbulent upward bubbly flow in a pipe.



(a) (b)

(c) (d)

Fig. 2 Convergence histories of the (a) SIMPLE, (b) SIMPLEC, (c) SIMPLEM, and (d) SIMPLEX algorithms using the SG, PG, and FMG methods on the finest mesh for turbulent air-particle flow in a pipe.

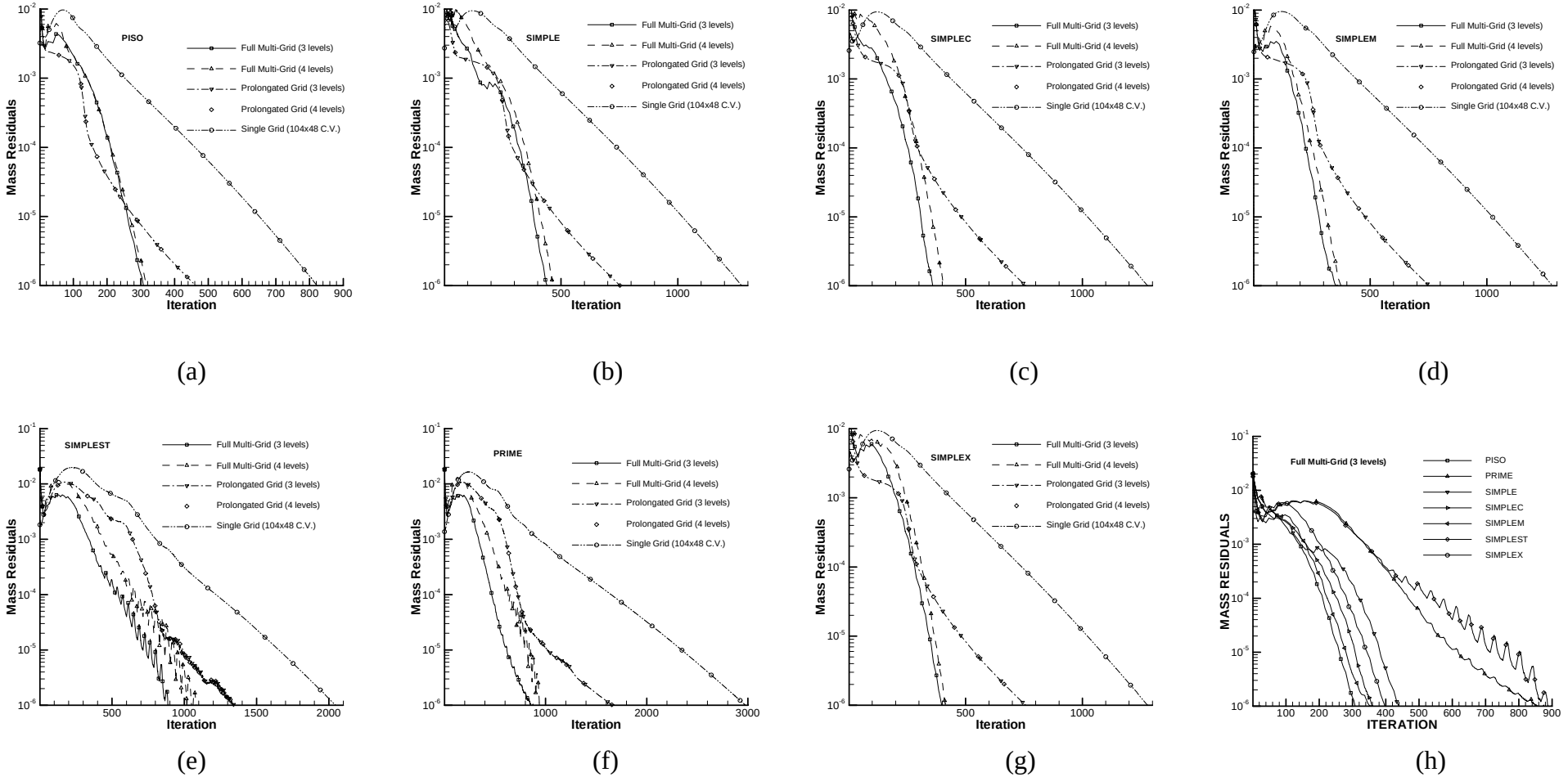
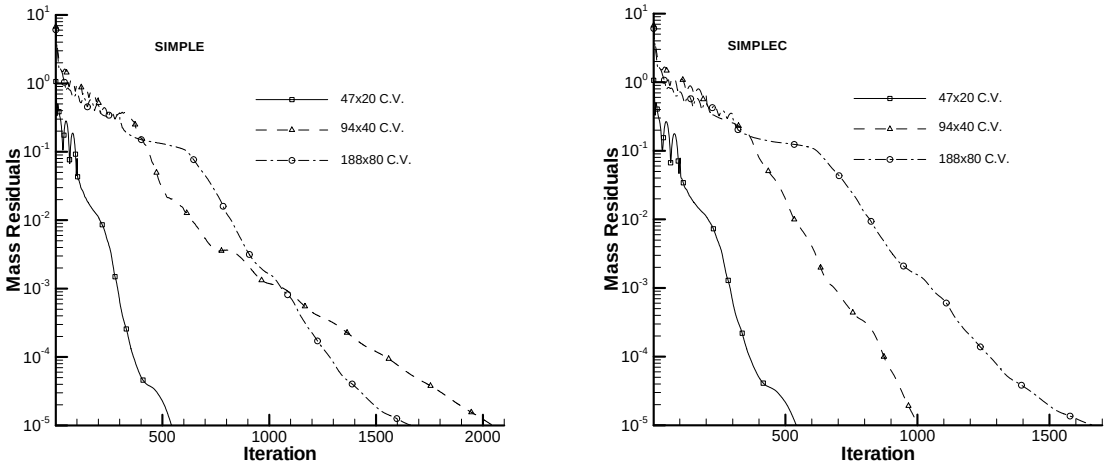


Fig. 3 (a)-(g) Convergence histories of the SG, PG, and FMG methods on the finest grid, and (h) convergence histories of the various algorithms on the finest mesh using the FMG method for dusty gas flow over a flat plate.



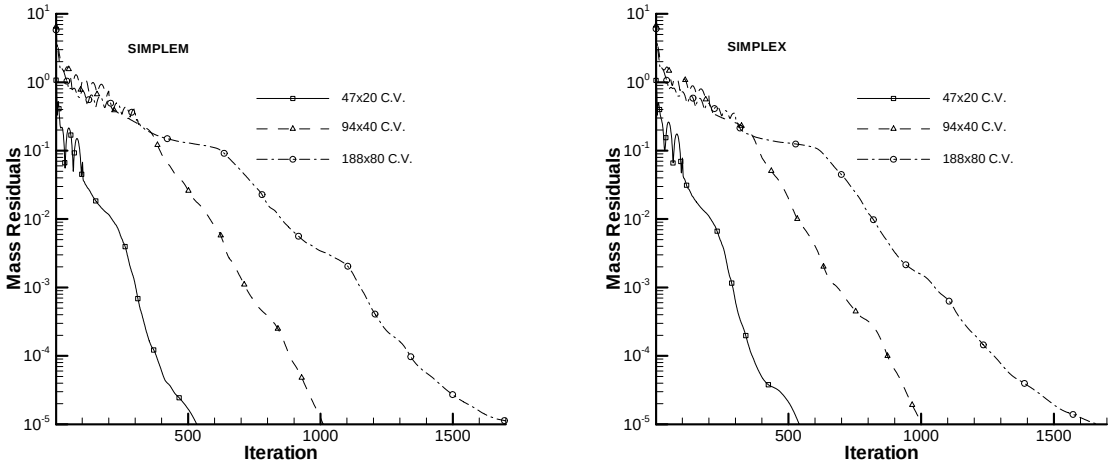


Fig. 4 Convergence histories of the (a) SIMPLE, (b) SIMPLEC, (c) SIMPLEM, and (d) SIMPLEX algorithms using the PG method for dusty gas flow in a converging-diverging nozzle.

Table 1 Normalized CPU-times for turbulent bubbly flow in a pipe.

		ALGORITHMS						
GRID	METHOD	SIMPLE	SIMPLEC	SIMPLEX	SIMPLEST	SIMPLEM	PISO	PRIME
48x16 C.V.	SG	1.00	1.00	1.02	1.04	1.09	1.08	1.19
96x32 C.V.	SG	40.09	40.42	40.98	40.66	43.07	41.82	43.59
	PG (3 levels)	34.24	34.41	35.02	36.82	37.12	34.48	36.05
	PG (4 levels)	34.34	34.25	34.67	36.88	36.93	34.50	36.07
	FMG (3 levels)	22.32	22.17	22.40	25.55	23.67	27.31	28.38
	FMG (4 levels)	22.53	22.51	22.78	23.81	25.55	55.68	27.89

Table 2 Normalized CPU-times for turbulent air-particle flow in a pipe.

		ALGORITHMS			
GRID	METHOD	SIMPLE	SIMPLEC	SIMPLEX	SIMPLEM
48x20 C.V.	SG	1.00	1.26	1.28	1.45
96x40 C.V.	SG	18.81	18.85	19.07	21.05
	PG (3 levels)	18.94	19.37	19.37	20.22
	PG (4 levels)	19.03	19.46	19.44	20.30
	FMG (3 levels)	11.11	12.34	12.44	13.72
	FMG (4 levels)	12.07	13.10	13.25	13.54

Table 3 Normalized CPU-times for Dusty flow over a flat plate.

		ALGORITHMS						
GRID	METHOD	SIMPLE	SIMPLEC	SIMPLEX	SIMPLEST	SIMPLEM	PISO	PRIME
52x24 C.V.	SG	1.00	1.04	1.11	1.52	1.31	1.12	2.13
104x48 C.V.	SG	16.92	18.09	18.94	26.00	21.48	19.46	45.87
	PG (3 levels)	14.25	14.94	15.51	18.85	17.48	15.84	24.48
	PG (4 levels)	14.27	14.80	15.50	18.90	17.56	15.79	24.46
	FMG (3 levels)	5.35	5.44	5.88	11.87	7.84	8.34	12.19
	FMG (4 levels)	6.12	6.14	6.66	12.98	6.66	9.08	9.98

Table 4 Normalized CPU-times for Dusty flow in a converging-diverging nozzle.

		ALGORITHMS			
GRID	METHOD	SIMPLE	SIMPLEC	SIMPLEX	SIMPLEM
47x20 C.V.	SG	1.00	1.02	1.06	1.13
94x40 C.V.	PG (2 levels)	13.08	9.24	9.47	11.37
188x80 C.V.	PG (3 levels)	82.43	88.49	91.35	117.83