AN IMPROVED VARIABLE DENSITY PRESSURE PROJECTION SOLVER FOR ADAPTIVE MESHES

AUSTEN DUFFY*, ALAN KUHNLE , AND MARK SUSSMAN †

Abstract. Tatebe [17] introduced the multigrid preconditioned conjugate gradient method (MGPCG) for solving the sparse matrix system that results from discretizing an elliptic equation with discontinuous coefficients and source terms. Tatebe's work was restricted to elliptic problems on a single structured grid. We present a MGPCG algorithm for an adaptive hierarchy of grids, and we show analytically that our method is guaranteed to converge. Using our new MGPCG algorithm to solve the projection equation, we report results for incompressible two-phase flow problems, and for incompressible flow in complex geometries. It is found that the new method is about twice as fast as the MG method proposed in [13, 14], about three times as fast as the PCG method proposed in [8] and that the new method scales well with increasing AMR levels.

Key words. Adaptive Mesh Refinement, Multigrid, Pressure Projection Solver, MGPCG

AMS subject classifications. 65Z05, 76T10,

1. Introduction.

The numerical simulation of incompressible fluid flows consisting of multiple phases with large density variations (such as in liquid-gas scenarios) can present significant challenges, particularly in the solution of the pressure projection step. The phase density ρ is represented in liquid and gas phases by $\rho = \rho_L H(\phi) + \rho_G(1 - H(\phi))$, where $H(\phi)$ is a Heaviside function equal to 1 in liquid and 0 in gas. The rate of convergence of the iterative solution of the pressure Poisson equation

$$\nabla \cdot \frac{1}{\rho} \nabla p = F \tag{1.1}$$

is retarded if $\frac{\rho_L}{\rho_G} >> 1$ because the resulting discretization matrix is poorly conditioned. Consider the 1D discretization of eqn. 1.1

$$\frac{1}{\rho_{i+\frac{1}{2}}} \frac{p_{i+1} - p_i}{\Delta x} - \frac{1}{\rho_{i-\frac{1}{2}}} \frac{p_i - p_{i-1}}{\Delta x} = \Delta x F_i, \tag{1.2}$$

where $\rho_{i+\frac{1}{2}} = 1$ in a first phase and α in a second phase, then the condition number of the corresponding discretization matrix becomes larger as the density ratio increases as seen in table 1.1. Though we have seen that the condition number is sensitive to the density ratio, we have found that the condition number is not necessarily sensitive to the problem geometry as seen in figure 1.2, where all illustrated geometries produce a discretization matrix whose condition number is of the same order of magnitude.

Significant steps in reducing the solution time of the pressure projection step for these problems have come from separate sources. The development of adaptive mesh refinement [4, 3, 15, 6] (AMR) has allowed for a reduction in the number of required

*Florida State University, Department of Mathematics, 208 Love Building, 1017 Academic way, Tallahassee, Florida, USA 32306-4510(aduffy@math.fsu.edu)

†Florida State University, Department of Mathematics, 208 Love Building, 1017 Academic way, Tallahassee, Florida, USA 32306-4510 (sussman@math.fsu.edu)

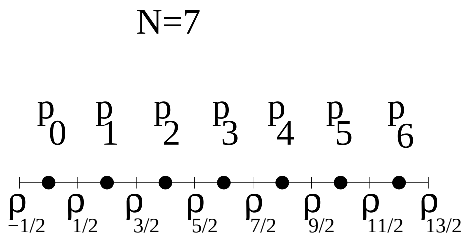

FIG. 1.1. *Example 1D discretization*

TABLE 1.1
In multiphase flows, the condition number of the discretization matrix for eqn. 1.1 grows with the density ratio. In this table, the condition number is calculated for the discretization matrix of a 1D two phase flow in the domain $[0,1]$ *following eqn. 1.2, and with the phase interface occurring at* $x = 0.25$. *The example flow has a density of 1 in the first phase on the interval* $[0, 0.25)$ *and* α *in a second phase on the interval* $(0.25, 1]$. *Values represent a discretization with 256 grid points and were calculated in MATLAB using the built in condition number function cond().*

α	1	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}
Density Ratio	1	10	10^2	10^3	10^4	10^5
Condition #	205	1.2 x 10^3	1.2 x 10^4	1.2 x 10^5	1.2 x 10^6	1.2 x 10^7

computational grid points when used in conjunction with a method such as multigrid (MG). Unstructured grid coarsening strategies have been implemented to allow for MG solutions of the Poisson equation in unstructured applications [18]. Alternatively, on uniform grids, MG has been used to form the preconditioning matrix for the preconditioned conjugate gradient (PCG) method. The multigrid preconditioned conjugate gradient method (MGPCG) was first introduced by Tatebe [17] who showed that poorly conditioned problems could be solved 12 times faster using MGPCG instead of MG, and 5 times faster than incomplete LU preconditioned CG (ILU-PCG). In our previous work [14] we developed a pressure solver on an adaptive hierarchy of grids using a combination of MG and MGPCG. The MGPCG "smoother" was applied a single level at a time. We have discovered though, that the performance of the solver developed in [14] is sensitive to the number of smoothing steps of MGPCG. In [14] a fixed number of pre-smoothing and post-smoothing MGPCG iterations were done, but we have found that the convergence rate of the overall MG AMR method was sensitive to the number of smoothing steps. If the number of smoothing steps increases, the cpu time increases, but if the number of smoothing steps decreases, the method may not converge at all. In [13] we had modified the algorithm in [14] by taking a number of MGPCG steps necessary in order to insure that the residual on a given level met a prescribed tolerance. The approach in [13] is more robust, but less efficient. Moreover, the performance of the method in [14, 13] is sensitive to the blocking factor.

1.1. Background and Related Work. Traditionally, the dominant methods for solving large systems of linear equations have been MG and PCG, each possessing its own benefits and drawbacks. Tatebe [17] united the two by using MG as the preconditioner for PCG which proved to be superior to both for problems with poorly conditioned matrices, noting that the strengths of each method complimented the weaknesses of the other. Block structured adaptive techniques for solving compressible hyperbolic problems emerged over twenty years ago [4, 3], and since have been

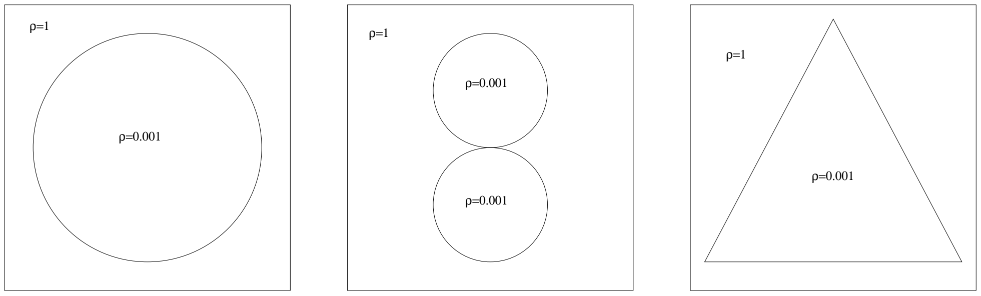

FIG. 1.2. *The condition number of the discretization matrix is not as sensitive to the problem geometry as it is to the density ratio. The corresponding condition numbers for these figures are 6,132,300 (left), 1,861,000 (middle) and 2,548,900 (right) using a 2D version of discretization eqn. 1.2 on a 64 x 64 grid.*

extended to incompressible multiphase flow problems using level set methods [15]. The quad tree method for handling adaptive meshes was introduced in [20] and, along with the octree method that followed, has been used in numerous works involving adaptive meshes [2, 1, 11]. Popinet [11] described a technique for numerically solving the time dependent incompressible Euler equations in complex geometries that differed from the classical block structured AMR methods. Using a combination of a quad/octree discretization first described in [2], an approximate projection method, a MG Poisson solver and a VOF embedded description for solid boundaries, [11] showed that the mesh adaptation procedure could be achieved with minimal overhead. Though in a later paper [12] Popinet did perform two phase flow calculations with this method, the MG solver could be considered non-optimal because Tatebe showed for a single grid that MGPCG was much faster than MG. The MG solver will perform poorly for many two phase flow problems, as pointed out by Tatebe [17], because the resulting matrix system is poorly conditioned (see table 1.1). We note that MGPCG can not be used in [11, 12] because their discretization matrix is not symmetric. [8, 9] continued the work of [11] by developing a symmetric discretization matrix allowing for the use of PCG which is guaranteed to converge, but still, Tatebe demonstrated that PCG is less efficient than MGPCG for even a uniform grid [17]. In [7], the method described in [8, 9] was accelerated by only using coarse grids in the projection step, but this technique can be avoided by instead using the faster MGPCG AMR method as described here.

2. Mathematical Formulation.

We will consider the following incompressible Navier-Stokes equations for multiphase flow [16, 19]:

$$\rho(\phi)(\vec{u}_t + \vec{u} \cdot \nabla \vec{u}) = -\nabla p + \nabla \cdot \mu(\phi) \nabla D - \gamma \kappa(\phi) \nabla H(\phi) + \rho \vec{g}.$$

$$\nabla \cdot \vec{u} = 0.$$

where $\rho(\phi)$ is the density, $\mu(\phi)$ is the viscosity, $\vec{u}$ is the velocity, D is the rate of deformation tensor, p is the pressure, $\kappa(\phi)$ is the curvature, $\vec{g}$ is the gravitational force, and $H(\phi)$ is the Heaviside function.

In the level set approach, the set of points,

$$\Gamma = \{(x, y, z) | \phi(x, y, z, t) = 0\},$$

represents the air/water interface Γ at time t. The equation governing the level set function is given by,

$$\phi_t + \vec{u} \cdot \nabla\phi = 0.$$

In order to increase the efficiency for simulating multi-phase flows, we have developed adaptive mesh refinement (AMR) level set methodology in order to dynamically place grid points in the vicinity of the air/water interface[15, 14, 5, 13].

The key component of this work considers the solution of the pressure projection equation,

$$\nabla \cdot \frac{1}{\rho}\nabla p = \nabla \cdot \vec{u}^*, \tag{2.1}$$

where $\vec{u}^*$ is an intermediate velocity field resulting from a standard "projection method" splitting scheme, and the density ρ is represented by both liquid and gas phases

$$\rho = \rho^L H(\theta) + \rho^G(1 - H(\theta)).$$

The discretization utilizes the level set function such that ρ is a function of ϕ and

$$\rho_{i+1/2,j}(\phi) = \rho^L\theta_{i+1/2,j} + \rho^G(1 - \theta_{i+1/2,j})$$

where $\theta_{i+1/2,j}$ is a height fraction,

$$\theta_{i+1/2,j} = \frac{\phi^+_{i,j} + \phi^+_{i,j+1}}{|\phi_{i,j}| + |\phi_{i,j+1}|}$$

and $\phi^+ = max(\phi, 0)$. The spatial discretization of eqn. 2.1 utilizes cell centered values for p with corresponding values at the cell faces for ρ such as in the example 1D discretization 1.2. We note that the use of first order boundary conditions results in a symmetric discretization matrix [9, 8]. The remainder of this section is devoted to solving eqn. 2.1.

2.1. An Improved Projection Algorithm for Adaptive Meshes. The first algorithm we will discuss here is the currently used MG algorithm for adaptive grids [13, 14] in order to demonstrate why improvement is needed. For the solution of the pressure projection step for incompressible flows in two phases, Tatebe's [17] MGPCG method is used as a smoother for each MG level, as described in either algorithm 1 [14] or 2 [13]. The problem with algorithm 1 or 2 is that the MGPCG smoother requires multiple levels of coarsening to occur in order to achieve optimal performance, but on an adaptive mesh these levels may not be available as demonstrated by the 8x2 level 1 grids in figure 2.2. Here, only one coarsening level can be achieved. The MGPCG smoother also requires one to compute on additional levels below the desired level as seen in fig. 2.1. Also, algorithm 1 is sensitive to the number of MGPCG smoothing steps as too many smoothing steps may lead to unnecessary increases in computation time, and too few might lead the MG AMR algorithm diverging. We improved the robustness of algorithm 1 [13] by selecting the number of smoothing steps to ensure the residual on a given level was less than a prescribed tolerance. Algorithm 2 is more robust than algorithm 1, but less efficient. To improve either algorithm 1 or 2, we replace the MG algorithm of [13, 14] with a MGPCG method, and use one of three possible MG preconditioning methods; a) MG with block ILU smoother, b) MG with Gauss-Seidel red black (GSRB) smoother, and c) MG with incomplete Cholesky red black (ICRB) smoother. This results in algorithm 3.

We note that the symmetric GSRB smoother that we have employed is not typical. The "GSRB" symmetric smoother employed by Tatebe used a red-black ordering down the MG v-cycle, and a black-red ordering up the v-cycle. Our GSRB smoother differs by using a red-black-black-red (RBBR) ordering at each step of the v-cycle. The ICRB smoother is an adaptation of the parallel incomplete Cholesky factorization of Ortega [10]. We remark that by choosing our smoother to be symmetric, it is unecessary for us to solve the coarsest level exactly (although we recommend it).

Algorithm 1 MG Algorithm for AMR [14]

```
Given $x^0$, $r = b - Ax^0$, $x = x^0$, $\delta x = 0$
Repeat until $||r|| < \epsilon$
1. Call relax($\delta x, r, \ell^{max}$) on finest level
2. Let $x = x + \delta x$, $r = r - A(\delta x)$

Recursive Routine relax($sol, rhs, \ell$)
if Coarsest Level then
  Solve exactly using MGPCG
else
  (a) Presmoothing Step
  for $i = 1$ to presmooth do
    Smooth using MGPCG on level $\ell$
  end for
  (b) Restriction Step
      (i) restrict($r$) to covered level $\ell - 1$ cells and exposed level $\ell - 1$ cells
  neighboring a covered cell.
      (ii) $cor = 0$
  (c) Relaxation on Next Coarser Level
      Call relax($cor^{coarse}, rhs^{coarse}, \ell - 1$)
  (d) Prolongate the Correction to the present level $l$ cells covering coarse level
  $\ell - 1$ cells and one layer of "virtual" level $\ell$ cells.
      $sol = sol + I(cor)$
  (e) Postsmoothing Step
  for $i = 1$ to postsmooth do
    Smooth using MGPCG on level $\ell$
  end for
end if
```

2.2. Restriction and Prolongation Operators. Our newly developed MGPCG method, algorithm 3, uses the same restriction and prolongation operators that were used in algorithms 1 and 2. We will define the restriction by operator R and the prolongation by operator P. To clearly define these operators, we will use a numbering system based on the transition from a coarse grid to an adapted fine level grid such as in figure 3.1. Let us assume that we have N coarse grid points at a given level

Algorithm 2 MG Algorithm for AMR [13]

```
Given $x^0$, $r = b - Ax^0$, $x = x^0$, $\delta x = 0$
Repeat until $||r|| < \epsilon$
1. Call relax($\delta x, r, \ell^{max}$) on finest level
2. Let $x = x + \delta x$, $r = r - A(\delta x)$

Recursive Routine relax($sol, rhs, \ell$)
if Coarsest Level then
    Solve exactly using MGPCG
else
    (a) Presmoothing Step
    while $||r_\ell|| > \frac{\epsilon}{10}$ do
        Smooth using MGPCG on level $\ell$
    end while
    (b) Restriction Step
        (i) restrict($r$) to covered level $\ell - 1$ cells and exposed level $\ell - 1$ cells
    neighboring a covered cell.
        (ii) $cor = 0$
    (c) Relaxation on Next Coarser Level
        Call relax($cor^{coarse}, rhs^{coarse}, \ell - 1$)
    (d) Prolongate the Correction to the present level $l$ cells covering coarse level
    $\ell - 1$ cells and one layer of "virtual" level $\ell$ cells.
        $sol = sol + I(cor)$
    (e) Postsmoothing Step
    while $||r_\ell|| > \frac{\epsilon}{10}$ do
        Smooth using MGPCG on level $\ell$
    end while
end if
```

ℓ, $x_1^\ell, x_2^\ell, \cdots, x_N^\ell$, then we will append the subscript with a 1,2,3 or 4 to define it as an adapted grid point at the next finest level. For example, if we adapted the second grid point we would then have $x_1^\ell, x_{21}^{\ell+1}, x_{22}^{\ell+1}, x_{23}^{\ell+1}, x_{24}^{\ell+1}, \cdots, x_N^\ell$, and could continue on adding grid points in this fashion, but here we will just use two levels. Following this numbering system, we find that the prolongation operator can be expressed by the mapping $P : x_i^\ell \to x_{ij}^{\ell+1}$ for an adapted cell, and $P : x_i^\ell \to x_i^{\ell+1}$ for a non-adapted cell. If one were to prolong the pressure on the coarse level ℓ, the matrix equation, describing the prolongation operator, would resemble that of eqn. 2.2.

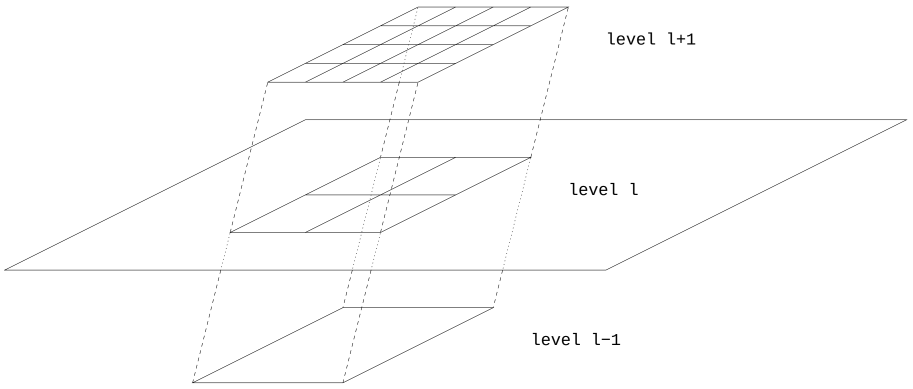

FIG. 2.1. *Adaptive mesh hierarchy in 2D. To compute the solution at level l+1, the MG AMR algorithm (alg. 1) requires calculations at levels l and l-1.*

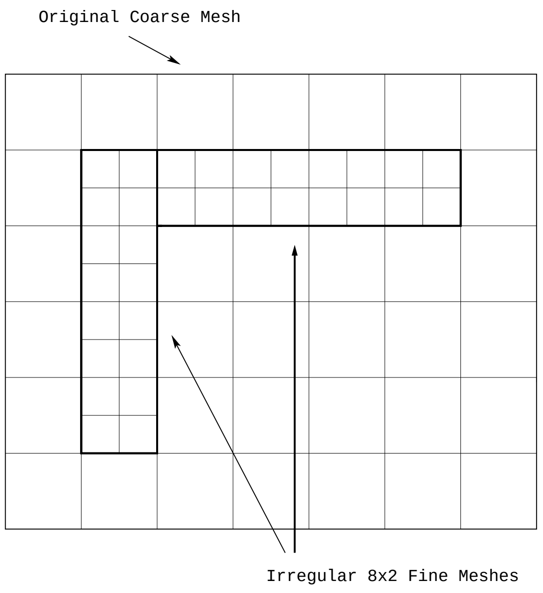

FIG. 2.2. *An MGPCG smoother can only achieve a single coarsening step on the illustrated fine level; then one must use PCG as a bottom solver on the "irregularly shaped" coarsest domain. On the other hand, our new MG preconditioner coarsens only one level too, but the bottom solver is MGPCG on the whole non-"irregularly shaped" domain. The bottom solver of our new method can coarsen two more times.*

$$\begin{bmatrix} 1 & 0 & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & & & & \\ \vdots & & \ddots & & & \\ \vdots & & & 1 & & \\ \vdots & & & 1 & & \\ \vdots & & & 1 & & \\ \vdots & & & 1 & & \\ \vdots & & & & \ddots & \\ 0 & & & & & 1 \end{bmatrix} \begin{bmatrix} p_1^\ell \\ p_2^\ell \\ \vdots \\ p_i^\ell \\ \vdots \\ p_N^\ell \end{bmatrix} = \begin{bmatrix} p_1^{\ell+1} \\ p_2^{\ell+1} \\ \vdots \\ p_{i1}^{\ell+1} \\ p_{i2}^{\ell+1} \\ p_{i3}^{\ell+1} \\ p_{i4}^{\ell+1} \\ \vdots \\ p_N^{\ell+1} \end{bmatrix} \tag{2.2}$$

Algorithm 3 Improved MGPCG Algorithm for AMR

```
Given $x^0$, $r = b - Ax^0$, $x = x^0$, $\delta x = 0$
$z = 0$
Call relaxAMR$(z, r, \ell^{max})$
$\rho = z \cdot r$
if $n = 1$ then
  $p = z$
else
  $\beta = \frac{\rho}{\rho_{old}}$
  $p = z + \beta p$
end if
$\alpha = \frac{\rho}{p \cdot (Ap)}$
$\rho_{old} = \rho$
$x = x + \alpha p$
$r = r - \alpha Ap$

Recursive Routine relaxAMR$(sol, rhs, \ell)$
if Coarsest Adaptive Level then
  Solve exactly using MGPCG
else
  (a) Presmoothing Step
  for $i = 1$ to nsmooth do
    Smooth using block ILU on level $\ell$
  end for
  (b) Restriction Step
      (i) restrict$(r)$ to covered level $\ell - 1$ cells and exposed level $\ell - 1$ cells
  neighboring a covered cell.
      (ii) $cor = 0$
  (c) Relaxation on Next Coarser Level
      Call relaxAMR$(cor^{coarse}, rhs^{coarse}, \ell - 1)$
  (d) Prolongate the Correction to the present level $\ell$ cells covering coarse level
  $\ell - 1$ cells and one layer of "virtual" level $\ell$ cells.
      $sol = sol + I(cor)$
  (e) Postsmoothing Step
  for $i = 1$ to nsmooth do
    Smooth using block ILU on level $\ell$
  end for
end if
```

The restriction operator R can be expressed by the mapping

$$R : p_i^{\ell+1} \rightarrow \begin{cases} \sum_{j=1}^{4} p_{ij}^{\ell} & \textit{adapted cells to coarse cell} \\ p_i^{\ell} & \textit{coarse cell to coarse cell} \end{cases}$$

This restriction yields a matrix equation like that of eqn. 2.3 when restricting grid points $x_{i1}, x_{i2}, x_{i3}, x_{i4}$ on a fine level grid to grid point x_i on a coarse level grid.

$$\begin{bmatrix} 1 & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ 0 & 1 & & & & & & & \\ \vdots & & \ddots & & & & & & \\ \vdots & & & 1 & 1 & 1 & 1 & & \\ \vdots & & & & & & & \ddots & \\ 0 & & & & & & & & 1 \end{bmatrix} \begin{bmatrix} p_1^{\ell+1} \\ p_2^{\ell+1} \\ \vdots \\ p_{i1}^{\ell+1} \\ p_{i2}^{\ell+1} \\ p_{i3}^{\ell+1} \\ p_{i4}^{\ell+1} \\ \vdots \\ p_N^{\ell+1} \end{bmatrix} = \begin{bmatrix} p_1^{\ell} \\ p_2^{\ell} \\ \vdots \\ p_i^{\ell} \\ \vdots \\ p_N^{\ell} \end{bmatrix} \tag{2.3}$$

Matrix eqns. 2.2 and 2.3 show that the restriction and prolongation operators are transposes of each other. We prove this in theorem 2.2 using the corresponding matrix representations of P and R given in definition 2.1.

Definition 2.1. *The prolongation operator P and restriction operator R are defined in matrix notation with a coarse index i and a fine index j such that*

$$P_{ij} = \begin{cases} 1 & \textit{if cell } i \textit{ is covered by cell } j \\ 0 & \textit{otherwise} \end{cases}$$

$$R_{ji} = \begin{cases} 1 & \textit{if cell } i \textit{ is covered by cell } j \\ 0 & \textit{otherwise} \end{cases}$$

Theorem 2.2. *The restriction operator R is the transpose of the prolongation operator P.*

Proof. following definition 2.1, it is straightforward to see that $R_{ji}^T = R_{ij}$, which is then equivalent to P_{ij} and hence the restriction operator is the transpose of the prolongation operator.
□

2.3. Convergence. In order to show that algorithm 3 converges, it is sufficient to show that it meets the criteria set forth by Tatebe [17] for convergence of the MGPCG method, i.e. 1) the MG smoother is symmetric, 2) the restriction operator is the transpose of the prolongation operator, and 3) the matrix A in the smoothing step

$$\begin{array}{lll} x^{k+1} & = x^k + M(b - Ax^k) & \textit{real cells} \\ x^{k+1} & = x^k & \textit{fictitious cells} \end{array}$$

is symmetric. Condition 2 is satisfied by theorem 2.2, and A will be symmetric as long as first order coarse/fine boundary conditions are applied [9, 8], so it is sufficient to show that the MG preconditioner produced by `relaxAMR` is symmetric. To show that the MG preconditioner on an adaptive mesh is symmetric, we will assume that the 'mesh' on each level is a union of 'real' fine cells and 'fictitious' coarse cells such as depicted in figure 3.1. On a given level ℓ, let M be the symmetric smoother produced by block ILU on the 'real' level ℓ grids, and the Jacobi method (i.e. $M = D^{-1}$) on the fictitious coarser level cells." We remark that our condition for a valid adaptive MG preconditioner is stricter than that of Tatebe [17]. We require that M is symmetric and do not allow a method that uses e.g. RBGS down the v-cycle and BRGS up. The corresponding MG smoother on a given level ℓ, $\bar{M}$, is then defined as: $\bar{M} = UMU$ where U is a projection matrix that projects vectors to level ℓ. In other words,

$$\bar{M} = \begin{cases} M & \textit{real cells on level } \ell \\ 0 & \textit{fictitious coarser level cells.} \end{cases}$$

Our equivalent smoothing step on level ℓ is $x^{k+1} = x^k + \bar{M}(b - Ax^k)$. Now, if one were to order the cells in such a manner that the variables corresponding to real cells were listed first, e.g.

$$[x_1, x_2, \cdots, x_N] \rightarrow [x_1^{real}, x_2^{real}, \cdots, x_n^{real}, x_1^{fictitious}, x_2^{fictitious}, \cdots, x_n^{fictitious}]$$

then we find that

$$\bar{M} = \begin{bmatrix} M & 0 \\ 0 & 0 \end{bmatrix}$$

and

$$U = \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}$$

Since M is symmetric, then $\bar{M} = UMU$ is also symmetric. Now $\bar{M}$ is singular, which means we cannot directly extend the theory developed by [17] (Theorem 2) to our adaptive MGPCG algorithm. In [17], it is required that P, which corresponds to the inverse of our smoother matrix $\bar{M}^{-1}$, be symmetric. We extend Tatebe's theory to our adaptive algorithm in theorem 2.3.

THEOREM 2.3. *The preconditioning step of algorithm 3, i.e. the solution of* `relaxAMR`*, is symmetric.*

Proof. We will consider a two level scheme with the $_c$ subscript representing the coarse level, and follow the steps outlined in the `relaxAMR` routine of algorithm 3. The generalization of our proof to multiple levels is analogous to Tatebe [17]. We start with the equation $z = \bar{M}r = UMUr$ corresponding to step a), and note that $R = P^T$.

$$
\begin{array}{lrl}
\text{a)} & z = & UMUr \\[2ex]
\text{b)} & r_c = & R(r - Az) \\
 & = & R(r - AUMUr) \\[2ex]
\text{c)} & z_c = & M_c R(r - AUMUr) \\[2ex]
\text{d)} & z = & UMUr + PM_c R(r - AUMUr) \\[2ex]
\text{e)} & z = & UMUr + PM_c R(r - AUMUr) \\
 & & +UMU(r - A(UMUr + PM_c R(r - AUMUr))) \\[2ex]
 & = & (UMU + PM_c P^T - PM_c P^T AUMU + UMU \\
 & & -UMUAUMU - UMUAPM_c P^T + UMUAPM_c P^T AUMU)r \\[2ex]
 & = & \tilde{M} r
\end{array}
$$

Since A, U and M are symmetric, each of the terms in the matrix sum found in the second equation of step e) are symmetric except for the third and sixth term, which are transposes of each other, and so the resultant matrix $\tilde{M}$ is symmetric. □

3. Results and Discussion. The benefits of using the MGPCG AMR algorithm over MG AMR (or PCG AMR) are the same benefits as outlined in Tatebe's paper for using MGPCG on a uniform grid versus MG or PCG on a uniform grid. To test the performance of the new method described here, we present timing results for the new MGPCG AMR algorithm along with both the original MG AMR algorithm and a PCG AMR algorithm. The problems tested are those of the simulation of a rising gas bubble through a liquid phase in both 2 and 3 space dimensions. All tests were run on a single core of an AMD Athlon X4 620 processor (2.6 GHz, 2 MB L2) with 4 GB of available DDR2 memory at 667 MHz.

To test the performance of the new method described here, we present timing results for the new MGPCG AMR algorithm along with both the original MG AMR algorithm and a PCG AMR algorithm similar to that of [9] that are implemented in the coupled level set and volume-of-fluid AMR (CLSVOF-AMR) code. The problems tested are those of the simulation of a rising gas bubble through a liquid phase in both 2 and 3 space dimensions, and flow past a whale. The overall algorithm for simulating bubble and drop flow is presented in [13]. All tests were run on a single core of an AMD Athlon X4 620 processor (2.6 GHz, 2 MB L2) with 4 GB of available DDR2 memory at 667 MHz.

3.1. 3D Axisymmetric Bubble. The first problem we will discuss is that of a 3D axisymmetric gas bubble rising through a liquid phase in in an r-z coordinate system. This problem uses $\mu_L/\mu_G = 53305$, $\rho_L/\rho_G = 1000$, $Fr = 0.757$, $Re = 0.038$ and $We = 0.0025$. Figure 3.2 displays representative discretization grids for this problem with increasing numbers of adaptive levels. Timing data for this problem is given in table 3.1 and a table giving the speedup factor of the new MGPCG AMR algorithm over the original MG AMR algorithm is provided in table 3.2. On this problem, the new MGPCG AMR algorithm is typically 2-3X faster than the original MG AMR algorithm, and performs up to 4X faster than the original MG AMR

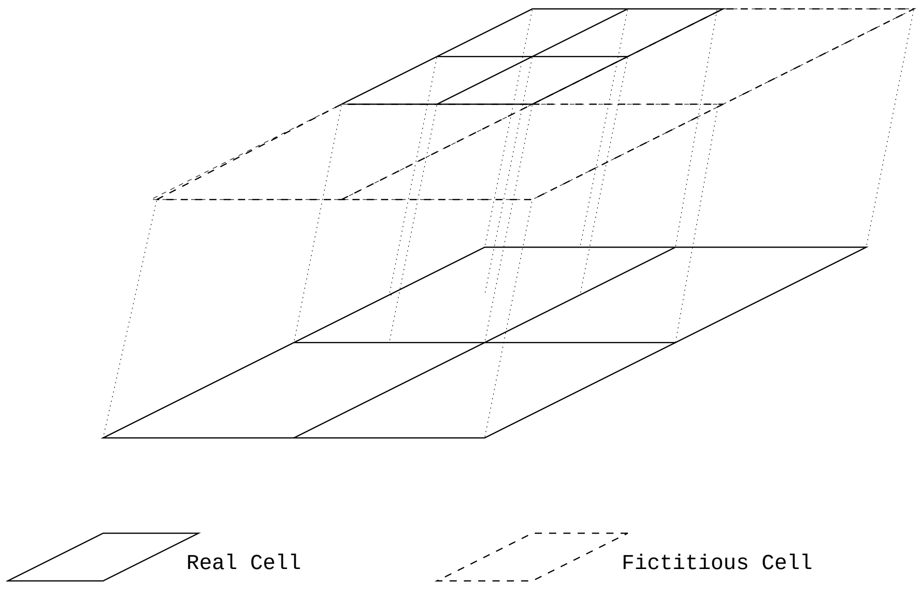

FIG. 3.1. *Coarse and fine grid levels depicting real and fictitious cells.*

TABLE 3.1
Timing results for a single pressure solve for the new MGPCG AMR algorithm along with the old MG-MGPCG AMR algorithm and the PCG algorithm for the 3D axisymmetric test bubble. The new MGPCG algorithm is faster than the old MG AMR algorithm in nearly every scenario. Grid sizes for a fixed blocking factor and number of adaptive levels are identical for each method.

Blocking Factor	2			4			8		
Adaptive Levels	1	3	5	1	3	5	1	3	5
ILU Smoother									
PCG	0.659	5.424	63.49	0.563	3.359	35.54	0.365	2.875	26.78
MG	0.270	2.181	13.93	0.252	1.349	10.05	0.098	0.751	6.060
MGPCG	0.142	0.636	4.109	0.127	0.439	2.493	0.096	0.382	2.156
ICRB Smoother									
PCG	0.653	5.366	69.49	0.567	3.561	39.02	0.377	3.012	25.53
MG	0.281	2.177	16.54	0.278	1.435	10.96	0.112	0.885	6.634
MGPCG	0.157	0.655	4.511	0.152	0.498	2.732	0.123	0.415	2.402
GSRB Smoother									
PCG	0.641	5.706	65.99	0.567	4.037	37.72	0.364	3.014	29.26
MG	0.284	2.165	13.91	0.266	1.367	10.38	0.108	0.845	5.957
MGPCG	0.153	0.678	4.426	0.145	0.464	2.743	0.118	0.408	2.176

algorithm when using a blocking factor of four along with five adaptive levels. The only case when the new solver underperforms is with a blocking factor of eight and only one adaptive level.

3.2. 3D Bubble. The second test problem is also a rising gas bubble through a liquid phase but expanded to three spatial dimensions, $\mu_L/\mu_G = 53305$, $\rho_L/\rho_G = 1000$, $Fr = 0.757$, $Re = 0.038$ and $We = 0.0025$. Figure 3.3 shows the grid for this problem with two adaptive levels and a blocking factor of four. Timing data for this problem is given in table 3.3 and a table giving the speedup factor of the new MGPCG AMR algorithm over the original MG AMR algorithm is provided in table 3.4. For this test problem, the new MGPCG AMR algorithm shows improvement over the MG AMR algorithm when more than one adaptive level is used and achieves a speedup of

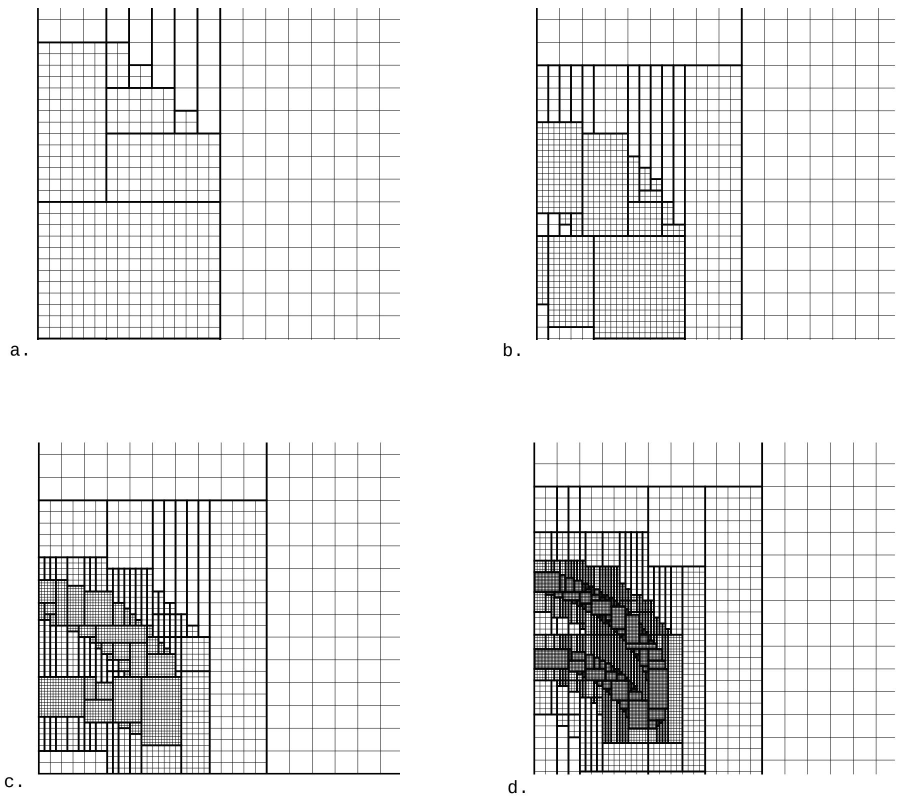

Fig. 3.2. *AMR grids for a rising 3D axisymmetric bubble. The representative grids all use a blocking factor of 2 and contain a) one adaptive level, b) two adaptive levels, c) three adaptive levels, and d) four adaptive levels.* $\mu_L/\mu_G = 53305$, $\rho_L/\rho_G = 1000$, $Fr = 0.757$, $Re = 0.038$ *and* $We = 0.0025$.

Table 3.2

Speedup factor for the new MGPCG AMR method over the original MG-MGPCG AMR method on the 3D axisymmetric test problem of simulating a gas bubble rising through a liquid phase. The new algorithm outperforms the old by an increasing margin as both the blocking factor and the number of adaptive levels are increased. It can be seen that the benefits of the new method are not realized in the case of large blocking factors and a single adaptive level.

Blocking Factor		2			4			8	
Adaptive Levels	1	3	6	1	3	6	1	3	6
ILU	1.91X	3.43X	3.39X	1.99X	3.07X	4.03X	1.01X	1.97X	2.81X
ICRB	1.79X	3.32X	3.67X	1.83X	2.88X	4.01X	0.91X	2.13X	2.76X
GSRB	1.86X	3.19X	3.14X	1.84X	2.95X	3.78X	0.92X	2.07X	2.74X

2X in the case of three adaptive levels and a blocking factor of two.

3.3. 3D Whale. The third problem tested here is simulated flow past a 3D whale body. For flow around an embedded solid, we discretize the solid as a staircase/rasterized geometry and we assign the coefficient, $\frac{1}{\rho_{i+1/2,j}}$, to be

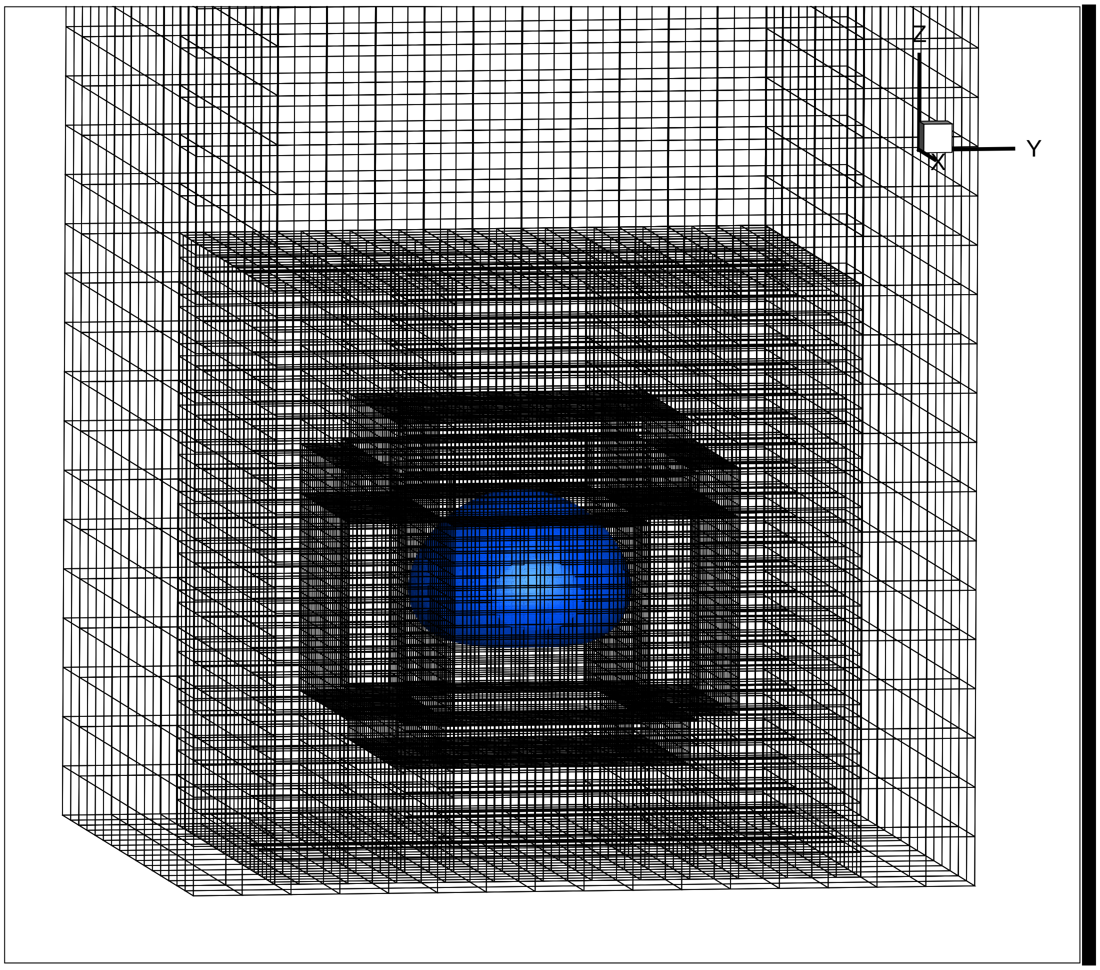

FIG. 3.3. *Mesh for a 3D rising bubble with 2 adaptive levels and a blocking factor of 4.* $\mu_L/\mu_G = 53305$, $\rho_L/\rho_G = 1000$, $Fr = 0.757$, $Re = 0.038$ *and* $We = 0.0025$.

TABLE 3.3
Timing results for a single pressure solve for the new MGPCG AMR algorithm along with the old MG-MGPCG AMR algorithm and the PCG algorithm for a 3D test bubble.

Blocking Factor		2			4			8	
Adaptive Levels	1	2	3	1	2	3	1	2	3
ILU Smoother									
PCG	3.839	18.45	71.11	3.326	14.24	49.79	4.154	22.22	69.70
MG	2.156	9.432	40.46	1.704	6.140	24.02	2.146	7.410	21.68
MGPCG	1.884	6.295	19.05	1.747	5.220	14.63	1.999	6.007	17.53
ICRB Smoother									
PCG	4.158	19.67	75.61	3.682	15.28	56.82	4.770	27.19	84.94
MG	2.227	9.916	41.31	1.910	6.958	30.35	2.490	10.10	28.47
MGPCG	2.354	7.083	19.36	2.263	5.950	16.96	2.686	7.559	20.33
GSRB Smoother									
PCG	4.023	19.99	78.66	3.735	15.43	58.33	4.801	26.19	86.51
MG	2.145	9.708	41.06	1.860	7.086	31.98	2.485	10.10	29.68
MGPCG	2.331	7.149	20.20	2.241	6.191	17.67	2.571	8.001	20.19

TABLE 3.4
Speedup factor for the new MGPCG AMR method over the original MG-MGPCG AMR method on the 3D test problem of simulating a gas bubble rising through a liquid phase.

Blocking Factor		2			4			8	
Adaptive Levels	1	2	3	1	2	3	1	2	3
ILU	1.14X	1.50X	2.12X	0.98X	1.18X	1.64X	1.07X	1.23X	1.24X
ICRB	0.95X	1.40X	2.13X	0.84X	1.17X	1.79X	0.93X	1.34X	1.40X
GSRB	0.92X	1.36X	2.03X	0.83X	1.14X	1.81X	0.97X	1.26X	1.47X

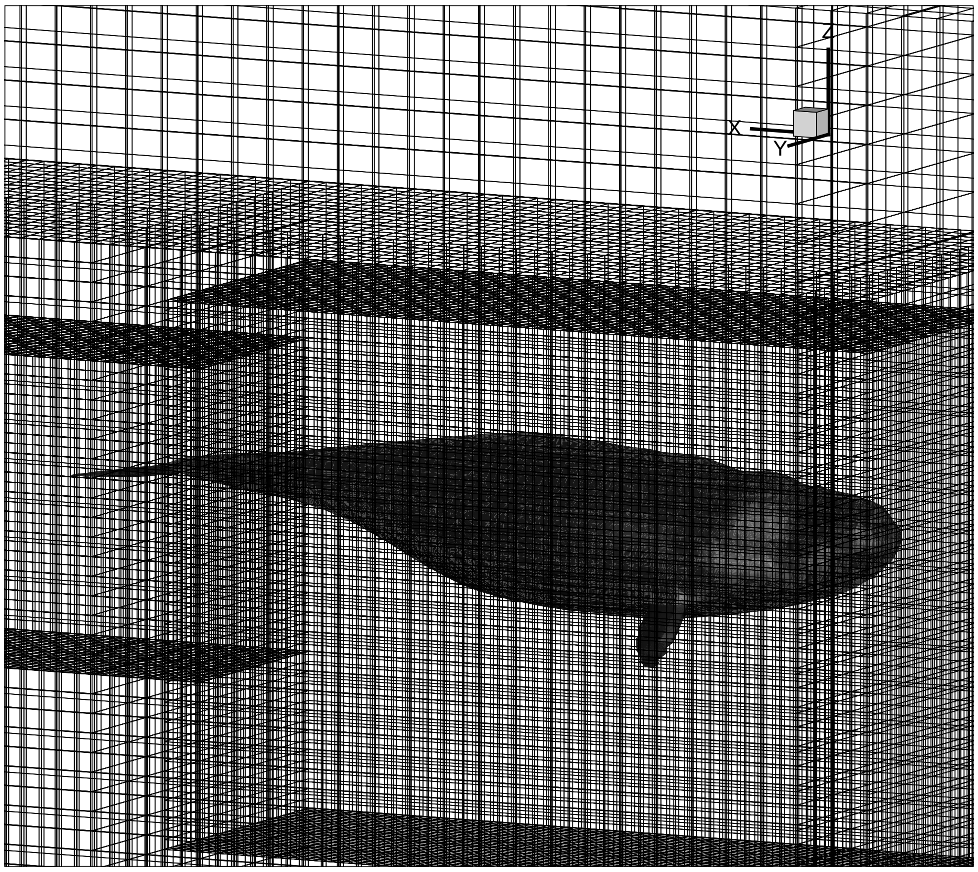

FIG. 3.4. *Mesh for a 3D whale with 2 adaptive levels.*

$$\frac{1}{\rho_{i+1/2,j}} = \begin{cases} 0 & \psi_{i+1,j} \leq 0 \text{ or } \psi_{i,j} \leq 0 \\ \frac{1}{\rho^L} & \text{otherwise} \end{cases}$$

where ψ is a solid/fluid level set function. Figure 3.4 shows a grid with two adaptive levels for this problem, along with the whale body. This test problem shows the best overall speedup of the new method and is 2-4X faster than the old method, even when only solving with one adaptive level as seen in table 3.6.

3.4. 3D Axisymmetric Bubble (zero level set not wholly contained on finest grid level). The last test problem is similar to the first 3D axismmetric test

Table 3.5

Timing results for a single pressure solve for the new MGPCG AMR algorithm along with the old MG-MGPCG AMR algorithm and the PCG algorithm for flow past a 3D whale.

Blocking Factor	2			4			8		
Adaptive Levels	1	2	3	1	2	3	1	2	3
ILU Smoother									
PCG	32.33	126.5	468.1	33.69	122.0	528.6	47.32	198.3	759.9
MG	10.52	41.69	145.6	9.874	35.71	133.4	13.91	57.37	202.2
MGPCG	5.445	15.06	43.66	5.796	14.84	48.74	7.113	21.28	70.07
ICRB Smoother									
PCG	40.15	171.9	636.1	42.80	160.6	705.6	68.93	269.2	1056
MG	14.93	64.01	226.1	14.61	61.08	226.1	24.02	102.5	371.9
MGPCG	7.606	19.93	56.83	8.794	19.45	62.45	10.49	28.93	94.08
GSRB Smoother									
PCG	41.07	179.2	648.8	43.11	167.0	710.2	70.00	276.3	1078
MG	15.99	67.29	230.4	15.42	63.24	234.9	26.03	106.8	395.9
MGPCG	8.063	22.38	57.47	8.451	21.61	65.76	10.39	31.77	94.70

Table 3.6

Speedup factor for the new MGPCG AMR method over the original MG AMR method on the 3D test problem of flow past a 3D whale.

Blocking Factor	2			4			8		
Adaptive Levels	1	2	3	1	2	3	1	2	3
ILU	1.93X	2.77X	3.33X	1.70X	2.41X	2.74X	1.96X	2.70X	2.89X
ICRB	1.96X	3.21X	3.98X	1.66X	3.14X	3.62X	2.29X	3.54X	3.95X
GSRB	1.98X	3.01X	4.01X	1.82X	2.93X	3.57X	2.51X	3.36X	4.18X

problem, but where the zero level set is not wholly contained on the finest level. Only portions of the zero level set that have a curvature that exceed a threshold are contained on the finest level. Figure 3.5 demonstrates the adaptive grid progression through time as the bubble rises for a blocking factor of 8 and 4 adaptive levels. This problem uses $\mu_L/\mu_G = 53305$, $\rho_L/\rho_G = 1000$, $Fr = 0.757$, $Re = 0.038$ and $We = 0.0025$. Timing data for this problem is given in table 3.7.

4. Conclusions.

We have developed an algorithm which allows for the use of the MGPCG method on an adaptive hierarchy of grids. This algorithm preserves the benefits and convergence properties of the original MGPCG algorithm while reducing the number of computational grid points required by uniform grids. It is almost always the case that our new algorithm outperforms our previous MG AMR algorithm for the problems tested, pardon the limiting factor of a single adaptive level. Also, our new method outperforms PCG on an adaptive grid by a comparable margin as reported by Tatebe [17] on a single grid. The speedup factor of the new MGPCG AMR algorithm over the original MG AMR method increases with increasing adaptive levels. We have found that there is an optimum blocking factor, for a given number of levels, with the MGPCG AMR algorithm: 4 in three dimensions and 8 in two dimensions. If we fix the blocking factor to be the optimal blocking factor, and increase the number of adaptive levels, then we find that the new MGPCG AMR method no longer consumes the majority of CPU time in comparison to the level set reinitialization step, for example.

REFERENCES

[1] G. Agresar, J. J. Linderman, G. Tryggvason, and K. G. Powell. An adaptive, cartesian, front-

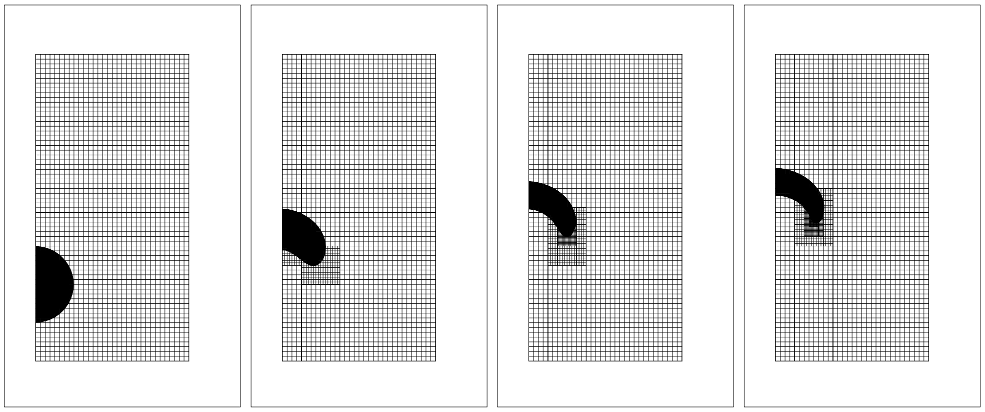

FIG. 3.5. *AMR grids for a rising 3D axisymmetric bubble where the zero level set is not wholly contained on the finest level. Only portions of the zero level set that have a curvature that exceed a threshold are contained on the finest level.* $\mu_L/\mu_G = 53305$, $\rho_L/\rho_G = 1000$, $Fr = 0.757$, $Re = 0.038$ *and* $We = 0.0025$.

TABLE 3.7
Timing results for a single pressure solve for the new MGPCG AMR algorithm along with the old MG AMR algorithm and the PCG algorithm for the 3D axisymmetric bubble test problem in which the zero level set is not wholly contained on the finest level. The new MGPCG algorithm is faster than the old MG AMR algorithm in every scenario. Grid sizes for a fixed blocking factor and number of adaptive levels are identical for each method.

Blocking Factor	4				8			
Adaptive Levels	1	2	3	4	1	2	3	4
ILU Smoother								
PCG	0.263	0.398	0.482	0.620	0.357	0.503	0.624	0.905
MG	1.661	2.627	0.342	0.479	2.413	2.945	0.374	0.569
MGPCG	0.091	0.137	0.135	0.157	0.118	0.152	0.155	0.215
ICRB Smoother								
PCG	0.270	0.409	0.482	0.617	0.365	0.506	0.637	0.928
MG	2.545	3.111	0.418	0.609	3.009	3.477	0.450	0.685
MGPCG	0.123	0.173	0.168	0.208	0.135	0.184	0.186	0.246
GSRB Smoother								
PCG	0.268	0.410	0.495	0.613	0.361	0.508	0.639	0.916
MG	2.046	2.971	0.226	0.555	2.914	3.090	0.421	0.653
MGPCG	0.109	0.149	0.164	0.188	0.127	0.156	0.183	0.233

tracking method for the motion, deformation and adhesion of circulating cells. *J. Comput. Phys.*, 143:346–380, 1998.

[2] S. A. Bayyuk, K. G. Powell, and B. van Leer. A simulation technique for 2-D unsteady inviscid flows around arbitrarily moving and deforming bodies of arbitrary geometries. In *AIAA 11th CFD Conference*, AIAA-93-3391-CP, Orlando, FL, USA, 1993.

[3] M. J. Berger and P. Colella. Local adaptive mesh refinement for shock hydrodynamics. *J. Comp. Phys.*, 82:64–84, 1989.

[4] M. J. Berger and J. Oliger. Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comp. Phys.*, 53:484–512, 1984.

[5] D. Dommermuth, M. Sussman, R. Beck, T. O'Shea, and D. Wyatt. The numerical simulation of ship waves using cartesian grid methods with adaptive mesh refinement. In *Proccedings of the Twenty Fifth Symposium on Naval Hydr.*, St. Johns, New Foundland and Labrador, Canada, 2004.

[6] B. E. Griffith, R. D. Hornung, D. M. McQueen, and C. S. Peskin. An adaptive, formally second order accurate version of the immersed boundary method. *J. Comput. Phys.*, 223:10–49, 2007.

[7] M. Lentine, W. Zheng, and R. Fedkiw. A novel algorithm for incompressible flow using only a

Table 3.8

Speedup factor for the new MGPCG AMR method over the original MG-MGPCG AMR method on the 3D axisymmetric test problem of simulating a gas bubble rising through a liquid phase where the zero level set is not wholly contained on the finest level. The new method is much faster in this scenario when using up to two adaptive levels, but exhibits speedup typical to the other example problems for three or more adaptive levels.

Blocking Factor	4				8			
Adaptive Levels	1	2	3	4	1	2	3	4
ILU	18.23X	19.20X	2.532X	3.046X	20.38X	19.70X	2.417X	2.651X
ICRB	19.89X	18.04X	2.489X	2.923X	22.23X	18.88X	2.423X	2.787X
GSRB	18.84X	19.99X	1.377X	2.947X	22.92X	19.87X	2.304X	2.810X

coarse grid projection. In *ACM SIGGRAPH 2010*, 2010.

[8] F. Losasso, R. Fedkiw, and S. Osher. Spatially adaptive techniques for level set methods and incompressible flow. *Comput. Fluids*, 35(10):995–1010, 2006.

[9] F. Lossaso, F. Gibou, and R. Fedkiw. Simulating water and smoke with an octree data structure. *ACM Trans. Graph.*, 23:457–462, 2004.

[10] J. M. Ortega. *Introduction to Parallel and Vector Solution of Linear Systems.* Frontiers of Computer Science. Plenum Press, New York, NY, 1988.

[11] S. Popinet. Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries. *J. Comput. Phys.*, 190(2):572–600, 2003.

[12] S. Popinet. An accurate adaptive solver for surface-tension-driven interfacial flows. *J. Comput. Phys.*, 228:58385866, 2009.

[13] P. Stewart, N. Lay, M. Sussman, and M. Ohta. An improved sharp interface method for viscoelastic and viscous two-phase flows. *J. Sci. Comput.*, 35(1):43–61, 2008.

[14] M. Sussman. A parallelized, adaptive algorithm for multiphase flow in general geometries. *Comput. Struct.*, 83:435–444, 2005.

[15] M. Sussman, A. Almgren, J. Bell, P. Colella, L. Howell, and M. Welcome. An adaptive level set approach for incompressible two-phase flows. *J. Comput. Phys.*, 148:81–124, 1999.

[16] M. Sussman, K. Smith, M. Hussaini, M. Ohta, and R. Zhi-Wei. A sharp interface method for incompressible two-phase flows. *J. Comput. Phys.*, 221(2):469–505, 2007.

[17] O. Tatebe. The multigrid preconditioned conjugate gradient method. In *6th Copper Mountain Conference on Multigrid Methods*, Copper Mountain, CO, USA, April 1993.

[18] J. Waltz and R. Löhner. A grid coarsening algorithm for unstructured multigrid applications. In *38th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA-00-0925, Reno, NV, USA, 2000.

[19] Y. Wang, S. Simakhina, and M. Sussman. A hybrid level set/volume constraint method for incompressible two-phase flow. *J. Comput. Phys.*, 2011. submitted.

[20] D. De Zeeuw and K. G. Powell. An adaptively refined carteshian mesh solver for the Euler equations. *J. Comput. Phys.*, 103:56–68, 1993.