AN ADAPTIVE-MESH PROJECTION METHOD FOR VISCOUS INCOMPRESSIBLE FLOW*

LOUIS H. HOWELL† AND JOHN B. BELL†

Abstract. Many fluid flow problems of practical interest—particularly at high Reynolds number—are characterized by small regions of complex and rapidly-varying fluid motion surrounded by larger regions of relatively smooth flow. Efficient solution of such problems requires an adaptive mesh refinement capability to concentrate computational effort where it is most needed. We present in this paper a fractional-step version of Chorin's projection method for incompressible flow, with adaptive mesh refinement, which is second-order accurate in both space and time. Convection terms are handled by a high-resolution upwind method which provides excellent resolution of small-scale features of the flow, while a multilevel iterative scheme efficiently solves the parabolic and elliptic equations associated with viscosity and the projection. Numerical examples demonstrate the performance of the method on two-dimensional problems involving vortex spindown with viscosity and inviscid vortex merger.

Key words. projection method, incompressible flow, adaptive mesh refinement

AMS subject classifications. 65M05, 65N05, 65N20, 76–04, 76D05

1. Introduction.

In this paper we develop a second-order adaptive mesh projection method for the incompressible Navier-Stokes equations

$$U_t + (U \cdot \nabla)U = -\nabla p + \varepsilon \nabla^2 U + F, \tag{1.1}$$

$$\nabla \cdot U = 0, \tag{1.2}$$

on a domain Ω, where U represents the velocity field, p represents the hydrodynamic pressure and F represents any external forces. We denote the x and y components of velocity by u and v, respectively. The method presented here represents an adaptive mesh version of the algorithm presented by Bell, Colella and Glaz [3]. We refer the interested reader to that paper for a discussion of the history of the development of projection methods.

Bell et al. were motivated by a desire to apply higher-order upwind methods developed for inviscid, compressible flow to the incompressible Navier-Stokes equations. In particular, they used a specialized version of the unsplit second-order Godunov methodology introduced for gas dynamics by Colella [13]. The Godunov method provides a robust discretization of the convection terms that avoids any cell-Reynolds-number stability restriction for high Reynolds number flow.

In this paper we present an adaptive mesh refinement version of this algorithm. The adaptive strategy is based on a mesh structure analogous to that used for hyperbolic conservation laws by Berger and Colella [7]. In this approach fine grids are recursively embedded in the coarse grid until sufficient resolution is obtained. An error estimation procedure automatically determines where the solution resolution is inadequate, and grid generation procedures dynamically create rectangular fine grid patches in these regions. (In [7] and the present work, fine grids are aligned with coarse

* This work was performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48. Partial support under contract No. W-7405-Eng-48 was provided by the Applied Mathematical Sciences Program of the Office of Energy Research. Other support was provided by the Defense Nuclear Agency under IACRO 94-831.

† Lawrence Livermore National Laboratory, Livermore, CA 94550

grids at all levels. Previous work by Berger and Oliger [8] used a more general structure in which fine grids could be placed in arbitrary orientations. An algorithm for solving the Navier-Stokes equations on this mesh structure was presented by Caruso, Ferziger and Oliger in [11].)

For the description of the algorithm in this paper, we will restrict our attention to homogeneous Dirichlet boundary conditions and assume that there are no external forces. We will also assume that the mesh spacing on the base grid is uniform in the x and y directions. These restrictions are not inherent limitations of the method; they have been adopted here for clarity of exposition. In addition, we note that the algorithm described here can easily be extended to variable density flows, cf. Bell and Marcus [5].

We begin our exposition in the following section by introducing the grid structure we are using and showing how the grid hierarchy is constructed. We also describe the overall time step strategy. In the third section we describe the discretization of the convection-diffusion step of the algorithm with emphasis on the issues related to adaptive mesh refinement. The fourth section describes the discretization of the projection and discusses in detail a multigrid algorithm for solving the discrete projection on the adaptive mesh hierarchy. In the fifth section we present computational results which demonstrate the convergence properties of the method, its robustness as a vortex pair crosses a coarse-fine grid interface, and the ability of the method to model and preserve fine details of a complex flow field.

2. Overview of the method. In this section we describe the hierarchical grid structure associated with the adaptive mesh refinement algorithm and review the basic fractional step scheme used in the algorithm.

2.1. Grid structure. The numerical calculations are defined on an adaptive hierarchy of logically rectangular grids. A single base grid, denoted as level $l = 0$, covers the entire problem domain. Grids at finer levels may be defined over parts of the domain as necessary to improve the accuracy of the computed solution. Each level of grids is refined by a factor of r from the level below it.

By clustering refined cells into grids we attempt to combine the advantages of a locally-uniform mesh spacing with those provided by adaptivity. Irregularities are confined to the interfaces between levels of refinement. We find that the difficulties associated with these interfaces are minimized for a refinement ratio $r = 4$. Greater values of r increase discretization errors at the interface and require an unnecessarily large portion of the domain to be refined. Setting $r = 2$, on the other hand, roughly doubles the required number of levels—and interfaces—and greatly increases the wasted storage devoted to coarse cells underlying finer grids.

Details of the problem discretization yield certain constraints on the placement of fine grids. At a minimum, we require a buffer of cells at level l surrounding the grids of level $l + 1$, except where those grids touch the boundary. The current algorithm is more restrictive than that, permitting interfaces to be placed only at intervals of eight coarse cells. (This is overkill, as the widest stencils currently used are six cells wide, and these are only applied parallel to interfaces. There is, however, a side benefit, in that all fine grid dimensions are guaranteed to be divisible by at least 32. This yields good multigrid convergence rates without the need to consider coarsening schemes for odd-length grids.)

The spatial discretization is based on a cell-centered approximation that provides the most natural setting for Godunov-type methods. For the base grid, we let i, j denote the cell whose center is located at $((i + 1/2)\Delta x_0,\, (j + 1/2)\Delta y_0)$ for

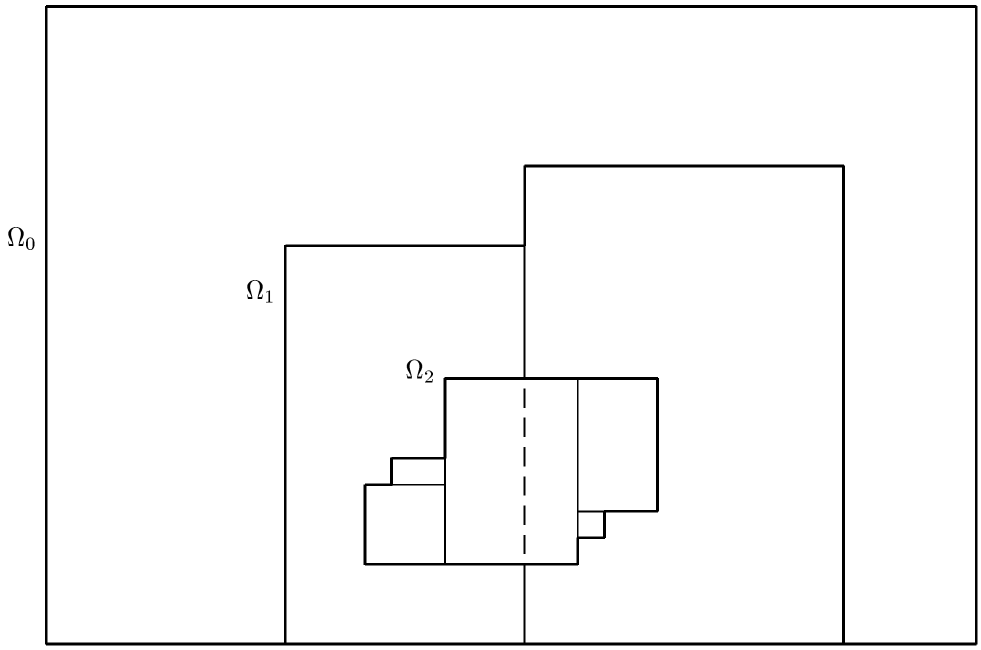

FIG. 2.1. *A properly nested hierarchy of grids*

$i \in \{0, \dots, I-1\}$, $j \in \{0, \dots, J-1\}$. The right edge and top edge of cell i, j are denoted by $i + {}^1\!/_2, j$ and $i, j + {}^1\!/_2$, respectively. Each finer level uses a refined index space with the same origin. Thus, cell i, j at level $l = 0$ may be covered by cells $ri + p, rj + q$ for $p \in \{0, \dots, r-1\}$, $q \in \{0, \dots, r-1\}$ at level $l = 1$.

At appropriate intervals we apply an error estimation procedure to the solution on the adaptive mesh to determine which cells have unacceptably large errors. Working from the top level down, we tag cells that have large errors, cells that have been covered by higher levels, and cells that must be refined to provide borders around higher levels. The tagged cells are then clustered into grids using an algorithm based on signature analysis techniques used in pattern recognition developed by Berger and Rigoutsos [9].

We denote the grids at level l by $G_{l,0}, \dots, G_{l,k-1}$, and the subdomain consisting of all cells refined to level l by Ω_l. The exact pattern of grids constructed at each level is not of great importance, as we require the solution algorithm to give identical results—down to the convergence tolerance required of the linear algebra solutions—regardless of how each level domain Ω_l is broken up into grids. See Figure 2.1 for a schematic representation of the grid hierarchy.

In hyperbolic versions of this algorithm, e.g. [7], a Richardson extrapolation procedure was used for estimating errors. The present incompressible method does nothing so elaborate as yet. The numerical experiments presented in section 5 use

$$C(\Delta x_l)^2(\overline{u})^{-1}\left(|u_{xx}| + |u_{yy}| + |v_{xx}| + |v_{yy}|\right) > 1 \tag{2.1}$$

as an error indicator, where $\Delta x_l = \Delta y_l$ is the mesh spacing at level l, $\overline{u}$ is a typical velocity used to scale the problem, and C is an order-1 constant that can be varied to tune the refinement criterion. We also choose a ceiling level, $l_{\max}$, above which no refinement is permitted.

2.2. Fractional step scheme. For the algorithm described in this paper all grids are advanced using the same time step so the fractional step scheme is the same as that introduced in Bell et al. [3]. Our strategy for solving the system (1.1)-(1.2) is a fractional step scheme having two parts: first we solve the convection-diffusion equations (2.2) without strictly enforcing the incompressibility constraint. Then, we project an intermediate vector field onto the space of discretely divergence-free vector fields. We emphasize that the three main components of the algorithm, namely, the convection discretization, the parabolic solves and the projection are all performed on the composite grid hierarchy. The algorithm will only be sketched here for a uniform spatial grid; the reader is referred to [3, 6] for a more detailed description.

The only boundary conditions presently used are slip walls for Euler flow and no-slip walls for viscous flow. These are applied to cell-centered velocity data by reflection to ghost cells on the other side of the wall, with a change in sign for those velocity components which go to zero at the wall. Some discussion of the adequacy of these boundary conditions appears in [3]. In numerical experiments we have confirmed second-order convergence in the velocity field with no degradation at the boundaries.

For the convection-diffusion step we solve for the intermediate velocity field U^*,

$$\frac{U^* - U^n}{\Delta t} + [(U \cdot \nabla)U]^{n+1/2} = \varepsilon \nabla_h^2 \left(\frac{U^* + U^n}{2} \right) - \nabla p^{n-1/2}, \tag{2.2}$$

where ∇_h^2 is a finite-difference approximation to the Laplacian. The pressure gradient is evaluated at $t^{n-1/2}$ and is treated as a source term in (2.2). (The pressure gradient is only computed at the $1/2$-time levels.) The convection terms in (2.2), namely $[(U \cdot \nabla)U]^{n+1/2}$, are approximated at time $t^{n+1/2}$ to second-order in space and time using an explicit predictor-corrector scheme. This scheme uses only data available at t^n. The implicit part of (2.2) thus consists of two decoupled parabolic equation solves, one for each velocity component.

The velocity field computed in the first step is not, in general, divergence-free. The projection step of the algorithm decomposes the result of the first step into a discrete gradient of a scalar potential and a discretely divergence-free vector field which correspond, respectively, to the new approximation to the pressure gradient and an update for the velocity. In particular, if $\mathbf{P}$ represents the composite grid projection then

$$\frac{U^{n+1} - U^n}{\Delta t} = \mathbf{P} \left[\frac{U^* - U^n}{\Delta t} + \nabla p^{n-1/2} \right], \tag{2.3}$$

$$\nabla p^{n+1/2} = (\mathbf{I} - \mathbf{P}) \left[\frac{U^* - U^n}{\Delta t} + \nabla p^{n-1/2} \right]. \tag{2.4}$$

(Note that the vector field we project is not U^*; it is an approximation to $U_t + \nabla p$.)

3. Discretization of the convection-diffusion step. In this section we describe the algorithm for the convection-diffusion step in the fractional step scheme, equation (2.2). We begin with a description of the convection algorithm for a single grid. We then discuss how the single grid algorithm is applied to compute the convection terms on the grid hierarchy. Finally, we discuss the solution of the parabolic part of (2.2) with $[(U \cdot \nabla)U]^{n+1/2}$ and $\nabla p^{n-1/2}$ treated as known source terms. We note that at the beginning of each time step the data is consistent between the different levels. In particular, if a cell at level l is covered by cells at level $l+1$ then the values of the velocity and pressure gradient in that cell are the average of the values of the cells at the higher level.

The convection algorithm is a second-order upwind method based on ideas first introduced by Colella [13].

3.1. Predictor. In the predictor we extrapolate the velocity to cell edges at $t^{n+1/2}$ using a second-order Taylor series expansion. For edge $i+1/2, j$ this gives

$$U^{n+1/2,L}_{i+1/2,j} = U^n_{i,j} + \frac{\Delta x}{2} U^n_{x,\, i,j} + \frac{\Delta t}{2} U^n_{t,\, i,j} \tag{3.1}$$

extrapolating from i,j and

$$U^{n+1/2,R}_{i+1/2,j} = U^n_{i+1,j} - \frac{\Delta x}{2} U^n_{x,\, i+1,j} + \frac{\Delta t}{2} U^n_{t,\, i+1,j} \tag{3.2}$$

extrapolating from $i+1, j$. The differential equation (1.1) is then used to eliminate the time derivatives to obtain

$$\begin{aligned} U^{n+1/2,L}_{i+1/2,j} &= U^n_{i,j} + \left[\frac{\Delta x}{2} - \frac{u_{i,j}\Delta t}{2}\right] U^n_{x,\, i,j} \\ &\quad + \frac{\Delta t}{2}\left[\varepsilon \nabla^2_h U^n_{i,j} - v_{i,j} U^n_{y,\, i,j} - \nabla p^{n-1/2}_{i,j}\right], \end{aligned} \tag{3.3}$$

$$\begin{aligned} U^{n+1/2,R}_{i+1/2,j} &= U^n_{i+1,j} - \left[\frac{\Delta x}{2} + \frac{u_{i+1,j}\Delta t}{2}\right] U^n_{x,\, i+1,j} \\ &\quad + \frac{\Delta t}{2}\left[\varepsilon \nabla^2_h U^n_{i+1,j} - v_{i+1,j} U^n_{y,\, i+1,j} - \nabla p^{n-1/2}_{i+1,j}\right]. \end{aligned} \tag{3.4}$$

Analogous formulae are used to predict values at each of the other edges of the cell. Here, ∇^2_h is the standard five point finite difference approximation to the Laplacian. The first derivatives normal to the edge (in this case U_x) are evaluated using a monotonicity-limited fourth-order centered slope approximation. This is equivalent to

$$U_x \approx \frac{-U_{i+2,j} + 8U_{i+1,j} - 8U_{i-1,j} + U_{i-2,j}}{12\Delta x} \tag{3.5}$$

throughout most of mesh, but limiting according to the procedure described in [12] is applied at some points to prevent the initial slopes from introducing new maxima and minima into the velocity field.

The transverse derivative terms (U_y) are evaluated using an upwind difference. In particular, we define

$$\overline{U}^B_{i,j+1/2} = U^n_{i,j} + \left(\frac{\Delta y}{2} - \frac{v_{i,j}\Delta t}{2}\right) U_{y,\, i,j} \tag{3.6}$$

$$\overline{U}^T_{i,j+1/2} = U^n_{i,j+1} - \left(\frac{\Delta y}{2} + \frac{v_{i,j+1}\Delta t}{2}\right) U_{y,\, i,j+1} \tag{3.7}$$

where U_y are limited slopes in the y direction with similar formulae for the lower edge. A procedure analogous to the corrector described below is applied to the $\overline{U}$ states to evaluate a convective difference approximation to vU_y.

3.2. Corrector. In the corrector we first resolve the ambiguity in the edge values. The convective part of (1.1) corresponding to the velocity normal to the edge is of the form

$$u_t + uu_x = \textit{source terms}. \tag{3.8}$$

This suggests the following upwind determination of the normal velocity component:

$$u_{i+1/2,j} = \begin{cases} u^L & \text{if } u^L \geq 0, u^L + u^R \geq 0, \\ 0 & \text{if } u^L < 0, u^R > 0, \\ u^R & \text{otherwise.} \end{cases} \tag{3.9}$$

(We suppress the $i + 1/2, j$ spatial indices on left and right states here and in the next equation. u^L and u^R are the x-components of U^L and U^R, respectively.) We now upwind U based on $u_{i+1/2,j}$:

$$U_{i+1/2,j} = \begin{cases} U^L & \text{if } u_{i+1/2,j} > 0, \\ U^R & \text{if } u_{i+1/2,j} < 0, \\ 1/2(U^L + U^R) & \text{if } u_{i+1/2,j} = 0. \end{cases} \tag{3.10}$$

Upwinding on the top and bottom edges uses $v_{i,j+1/2}$ in a similar fashion. Finally, we use these upwind values to form an approximation to the convective derivatives in (1.1)

$$\begin{aligned} uU_x + vU_y \quad \approx \quad & 1/2(u_{i+1/2,j} + u_{i-1/2,j})(U_{i+1/2,j} - U_{i-1/2,j}) + \\ & 1/2(v_{i,j+1/2} + v_{i,j-1/2})(U_{i,j+1/2} - U_{i,j-1/2}). \end{aligned} \tag{3.11}$$

The extension of the convection scheme to the adaptive grid is fairly straightforward since the difference scheme is explicit. To evaluate $[(U \cdot \nabla)U]^{n+1/2}$ on grid G_{k_l} we need only define data in a sufficient number of boundary cells surrounding the grid to evaluate the difference approximation. (Three cells are required in this case, due to use of the fourth-order slope formula (3.5).) These cells are filled by first intersecting the grid with other grids of the same level. If that does not completely fill the boundary patch the required data are interpolated from coarser grids using a biquadratic (third-order) formula.

The hyperbolic AMR algorithm [7] incorporates a "refluxing" step, in which edge fluxes for coarse grid cells at the coarse-fine interface are adjusted to equal those computed for neighboring fine cells, thus enforcing conservation. We experimented with refluxing for the incompressible algorithm, but found its effects to be insignificant compared to those of the interface stencils for the projection. The numerical experiments in section 5 do not use refluxing. Other complications of the method in [7] are due to advancing the fine grids at smaller time steps than the coarse grids, and do not apply to the present incompressible algorithm.

The Godunov method is an explicit difference scheme and, as such, requires a time step restriction. A linear, constant-coefficient analysis of the convective part of the scheme shows that we must require

$$\max_{i,j \in G_{k,l}} \left(\frac{|u_{i,j}| \Delta t}{\Delta x_l}, \frac{|v_{i,j}| \Delta t}{\Delta y_l} \right) \leq 1 \tag{3.12}$$

for stability. (Here Δx_l and Δy_l correspond to the grid spacing at level l.) A more thorough analysis by Minion [19] includes the influence of the viscous term and suggests that we must reduce this time step by half to obtain stability, independent of the size of ε. This agrees with our empirical observation that a time step one half of (3.12) yields stability in practice—that is the step we use in our computational examples later in the paper. Minion also derives a modification to the convective difference scheme that he believes to be stable up to the full time step (3.12), but we have not yet incorporated this change into our code.

(Another variation of the convection scheme, presented in [4], applies a staggered-grid projection to the edge velocities instead of using the lagged pressure gradient $\nabla p^{n-1/2}$ in the predictor. This version appeared in practice to be stable at or near the full time step (3.12), though Minion suggests that it should be subject to the same stability criterion as the original method. We have not yet implemented the staggered-grid projection in an adaptive context.)

3.3. Parabolic approximation. Once the convection terms are evaluated on all of the grids, to complete the solution of (2.2) we must solve the diffusion part of the system with the pressure and convective forces held fixed. The time discretization for the parabolic terms in (2.2) uses a Crank-Nicholson formula, applied simultaneously to all grids of the mesh hierarchy. For the space discretization we apply the standard 5-point stencil for the Laplacian in uniform parts of the mesh, with interpolated extensions across coarse-fine grid interfaces. Note that this is not the same Laplacian approximation as will be used for the projection.

The multilevel scheme we use to solve the implicit system (2.2) for U^* is nearly identical to that used for the projection operator, which is described in the next section. The primary differences are that the linear operator for diffusion is parabolic rather than elliptic, U^* is a vector rather than a scalar, and the complications generated by the decoupled nature of the projection do not apply to the diffusion calculation.

The method operates in residual-correction form. In order to evaluate the residual on all points of the composite mesh, we need rules for applying the operator

$$LU = \frac{U}{\Delta t} - \frac{\varepsilon}{2}\nabla_h^2 U, \tag{3.13}$$

where U is the current approximation to U^*, to points on both sides of the coarse-fine interface. Our approach is to apply the same 5-point Laplacian as in the interior, using suitably interpolated values at the interface.

We use third-order biquadratic interpolation from the coarser levels to fill ghost cells around the edges of each finer level. Only a single row of ghost cells is required to make the 5-point Laplacian well-defined throughout the interior of the fine region. Since the Laplacian is a second-order difference, applied to a formally third-order accurate interpolate, the accuracy of the approximation drops to first-order along the interface.

The ghost cells which intersect other fine grids are naturally filled from those grids, and updated with the most recent values whenever necessary. The method thus resembles a domain decomposition method on each level, with minimal overlap but with additional coupling provided through the interactions with coarser levels.

Values of U in coarse cells underlying the fine grid appear in the scheme in two different ways. For the calculation of fine grid residuals they appear in the interpolation formula, which must yield formal third-order accuracy. For the evaluation of the Laplacian on coarse cells bordering the interface formal third-order accuracy is also required, in order to yield a first-order Laplacian approximation. It is thus not sufficient to simply average the fine values in each coarse cell, since this yields only second-order accuracy at the coarse cell center. Instead, we use a fourth-order interpolation stencil of the form

$$\frac{1}{256}\begin{bmatrix} 1 & -9 & -9 & 1 \\ -9 & 81 & 81 & -9 \\ -9 & 81 & 81 & -9 \\ 1 & -9 & -9 & 1 \end{bmatrix} \tag{3.14}$$

to obtain U values at the centers of coarse cells just inside the fine level, which yields a second-order Laplacian approximation in neighboring coarse cells just outside the interface.

This is the only instance in which an unweighted average onto coarse cells is not sufficient. Averaging is used for defining coarse grid values for the Godunov algorithm, and is used within the multilevel parabolic solver for restricting residuals onto coarser grids. Equation (3.14) is also the only stencil presented in this paper that would require modification if the refinement ratio between levels were 2 instead of 4.

In case our use of third- and fourth-order interpolation schemes has caused confusion, we emphasize that the solutions we compute are at best second-order accurate. The higher-order interpolation stencils are needed only for filling in fine level ghost cells and coarse cells under the fine grids, as an intermediate step in computing Laplacian stencils that bridge the interface. These interpolated cells are in a sense fictitious, in that they are not the primary grid points where the solution is defined. Numerical results given in section 5 confirm that the first-order Laplacian approximation at coarse-fine interfaces allows for a velocity solution which is globally second-order, even in the extreme case of Stokes flow.

4. Multilevel projection. Given an intermediate velocity field U^*, we wish to obtain its projection $U^{n+1} = \mathbf{P}(U^*)$ onto the space of discretely divergence-free vector fields. More specifically, we want to compute a discrete gradient field $Gp^{n+1/2}$ such that

$$\frac{U^{n+1} - U^n}{\Delta t} + Gp^{n+1/2} = \frac{U^* - U^n}{\Delta t} + Gp^{n-1/2}. \tag{4.1}$$

If we denote the (known) right hand side of this expression by V and let D be the discrete divergence operator, this yields

$$DG\phi = DV, \tag{4.2}$$

where $G\phi$ is the desired discrete gradient field approximating $\nabla p^{n+1/2}$. The potential ϕ satisfies a Neumann boundary condition at solid walls, which we model by reflection.

The projection thus defined is a discretization of the continuous Hodge decomposition, which is discussed more thoroughly in [3]. For our purposes it will suffice to point out that if the operators D and G are adjoint—i.e., given an appropriate choice of inner products, $(DV, \phi) = -(V, G\phi)$ for any vector field V and scalar field ϕ—then $\mathbf{P}$ will be a true projection: orthogonal and idempotent ($\mathbf{P}^2 = \mathbf{P}$). If D and G are not adjoint then $\mathbf{P}$ will not be an orthogonal projection. It will still be idempotent, however, and $U^{n+1} = \mathbf{P}(U^*)$ will still be discretely divergence free. Finally, if we were to discretize the Hodge decomposition using some other discrete Laplacian approximation instead of DG, then the "projection" $\mathbf{P}$ would not even be idempotent and the computed vector field U^{n+1} would not be discretely divergence-free.

We wish to solve (4.2) for ϕ on an adaptive hierarchy of grids. Our Laplacian discretization will be derived from separate stencils for D and G at all points, including those points bordering coarse-fine interfaces. ("Approximate" projection methods, which do not share this property, will be cited below but are beyond the scope of this paper.) Our method thus differs from other multilevel algorithms, e.g. [18], where the coarse-fine discretizations are based on multigrid transfer operators. There is much more separation in our approach between the specification of the problem to be solved and the iterative algorithm used to solve it. This separation is not complete, however, for we will discuss below how the requirements of the multilevel iteration influence the choice of discretizations for the divergence and gradient operators.

In order to accommodate the need for special stencils at the coarse-fine interface, and to avoid additional complications stemming from the factor of four refinement between levels of the adaptive hierarchy, we use multigrid iterations on the individual grids that are distinct from the overall multilevel iteration. Coarser levels of the individual multigrids do not communicate with each other directly, nor do they communicate directly with coarser levels of the adaptive hierarchy. Little modification of the multilevel algorithm would be required even if the individual grids employed a completely different kind of solver, such as the conjugate gradient scheme used in [3].

To describe the multilevel iteration we will write (4.2) in residual-correction form. A linear operator $L = DG$ is defined across the entire adaptive mesh structure. The primary equation to be solved is $L\phi = f$, which has the residual form $L\psi = r$ for $r = f - Lv$ and $\psi = \phi - v$. Here v is the current best approximation to ϕ. To relax on individual grids we apply the FMV multigrid cycle from [10] (similar to the F-cycle given in [21] but with no smoothing on the initial descent). The multilevel algorithm is as follows:

- Set v equal to the value of ϕ obtained at the previous time step, if available, or zero otherwise (after initialization or regridding).
- Repeat until the residual norm $||r||_\infty$ is less than the required tolerance:
  - Compute residual $r = f - Lv$ on all grids, including coarse-fine interfaces.
  - Restrict residuals from fine to coarse grids.
  - Set correction ψ to zero at coarse level.
  - For each level l, from coarse to fine, do:
    - Execute FMV cycle for residual equation $L\psi = r$ on each grid of level l, using values of ψ from adjacent grids at levels l and $l-1$ for boundary conditions as necessary.
    - Add correction ψ into v at level l.
    - Interpolate correction ψ to next finer level $l+1$, if any.

The multilevel iteration is necessary to achieve convergence at grid interfaces, both between grids at the same level and between coarse and fine grids at adjacent levels. In this context there is nothing gained by applying more than a single FMV cycle on each individual grid. We obtain the best multilevel performance when each smoothing step of the FMV cycle consists of two Gauss-Seidel updates. Convergence rates generally range between one and two multilevel cycles per order of magnitude decrease in residual, depending on the complexity of the mesh structure. Numerical evidence presented in [15] suggests that the limited communication between multilevel and multigrid iterations slows the overall convergence, so future versions of the adaptive projection method will likely incorporate a more tightly integrated algorithm.

It remains to present the stencils used for divergence, gradient, restriction and interpolation. We have chosen the simplest second-order discretizations for divergence and gradient, using centered differences for the derivatives in uniform parts of the mesh:

$$(DU)_{i,j} = \frac{1}{\Delta x}(u_{i+1,j} - u_{i-1,j}) + \frac{1}{\Delta y}(v_{i,j+1} - v_{i,j-1}), \tag{4.3}$$

$$(G\phi)_{i,j} = \left(\frac{1}{\Delta x}(\phi_{i+1,j} - \phi_{i-1,j}), \frac{1}{\Delta y}(\phi_{i,j+1} - \phi_{i,j-1})\right). \tag{4.4}$$

Composition of these then yields a Laplacian stencil DG of the form

$$\frac{1}{4(\Delta x)^2}\begin{bmatrix} & & 1 & & \\ & & 0 & & \\ 1 & 0 & -4 & 0 & 1 \\ & & 0 & & \\ & & 1 & & \end{bmatrix}, \tag{4.5}$$

where Δy has been set equal to Δx. This stencil has the unfortunate effect of providing no coupling between adjacent grid cells. The result is four separate locally decoupled sets of points which only interact at the boundaries.

Though some alternative projection formulations avoid this decoupling problem, they have corresponding disadvantages of their own. There are approximate projections, [2] and [17], which are not idempotent. There are also exact projections that yield Laplacian stencils which are large and asymmetrical [20], or require placement of the velocity components at cell edges [14] in a manner incompatible with the Godunov convection scheme. The decoupled approach (4.5) which we are using requires certain alterations to the conventional multigrid transfer operators, but has not degraded the performance of the projection in the fluid algorithm as a whole.

Consider the effect of decoupling on a single grid. Multigrid depends on the fact that a solution to a coarsened system provides a good approximation to the desired fine solution. However smooth the initial right hand side, later residuals in a multigrid iteration tend to have significant components at all wavenumbers. The key ingredient to a successful multigrid acceleration for the decoupled stencil (4.5) on the fine level is to preserve the same decoupling pattern on all coarser levels. That is, solution components which are not coupled at one level must be kept separate by the grid transfer operators to adjacent levels.

We define transformations between coarse and fine index spaces as follows,

$$I = 2 \cdot \lfloor i/4 \rfloor + i \bmod 2, \tag{4.6}$$

$$i = 4 \cdot \lfloor I/2 \rfloor + I \bmod 2 \tag{4.7}$$

and similarly for J, j. Capitals denote indices on the coarse grid, lower case on the fine grid, and $\lfloor\ \rfloor$ reduces its argument to the next lower (or equal) integer. Each coarse point (I, J) then has four fine points associated with it: (i, j), $(i, j+2)$, $(i+2, j)$, $(i+2, j+2)$. Decoupled restriction is a simple average of residuals from these four fine points to their associated coarse point. For interpolation, we use a decoupled piecewise-constant formula—the value from the coarse point is distributed unchanged to each of the associated fine points.

There are both theoretical results and experiments, discussed in [21], which suggest that for second-degree problems at least one of these operations must employ

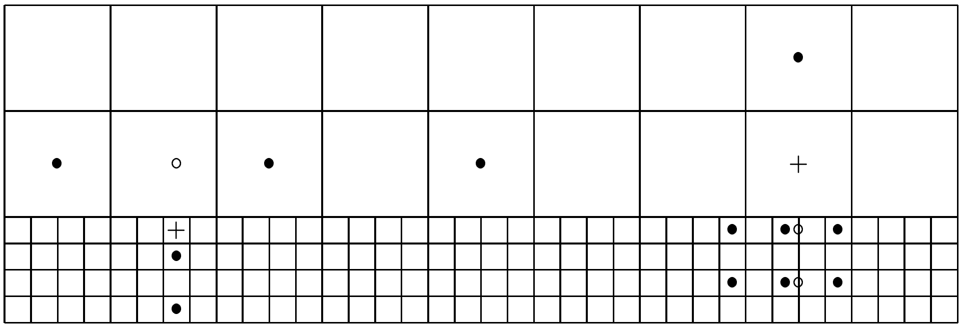

FIG. 4.1. *Examples of decoupled derivative stencils across a coarse-fine interface. The crosses indicate a fine cell (left) and a coarse cell (right) at which y-derivatives are evaluated. Bullets show which cells participate in the stencils. In each case, values on the opposite side of the interface are interpolated in the transverse direction to the circled points, giving three values on a line normal to the interface from which the derivative can be computed.*

a higher-order formula in order to give satisfactory convergence rates. Our own experience does not bear out this assertion for the constant-coefficient stencil (4.5). (For a variable-coefficient version of the projection we have found that replacement of piecewise-constant interpolation with a piecewise-bilinear formula does improve performance. The precise formula used, however, relies on the regularity of the single-grid geometry and may not be applicable to the adaptive case. For additional discussion of the variable-coefficient method, and experiments regarding multigrid convergence rates, see [15].)

The decoupling must also be respected by the multilevel adaptive algorithm. Since the refinement ratio is four in this case, the index transformations are

$$I = 2 \cdot \lfloor i/8 \rfloor + i \bmod 2, \tag{4.8}$$

$$i = 8 \cdot \lfloor I/2 \rfloor + I \bmod 2, \tag{4.9}$$

and each coarse point has 16 fine points associated with it. Restriction is again a simple average over the fine points associated with each coarse point. Piecewise-constant interpolation, however, does not work as well for this case as it does for the factor-of-two multigrid. We have achieved best results with a third-order biquadratic formula using coarse cells from the appropriate decoupled grid component.

Finally, the success of the multilevel iteration depends on the stencils chosen for divergence and gradient across coarse-fine interfaces. The coarse level solution is computed in decoupled fashion based on decoupled residuals restricted from the next finer level. In order for the multilevel iteration to exhibit reasonable convergence behavior, this coarse solution must be a good approximation to the composite solution on coarse and fine levels taken together. This requires that the operator $L = DG$, and thus the divergence and gradient operators themselves, also respect the decoupled structure of the grid.

Stencil outlines for both fine and coarse points near the interface are shown in Figure 4.1. In both cases we use quadratic interpolation to obtain third-order accurate values on the opposite side of the interface, then a three-point difference formula to give a second-order accurate derivative at the desired point. Composition of second-order accurate derivative approximations in D and G gives a Laplacian approximation that is first-order accurate along the interface, which is sufficient for global second-order accuracy of the projected velocity field.

	Base Grid Sizes, Convergence Rates				
	16–32	Rate	32–64	Rate	64–128
Stokes	0.000797724	1.77719	0.000232737	1.93666	6.07959e-05
Re 100	0.00983920	2.38129	0.00188851	2.17930	0.000416951
Euler	0.0184371	2.36604	0.00357639	2.25092	0.000751364

TABLE 5.1

Convergence rates for a smooth problem on grid pairs with one base grid and one refined patch. We give L_2 norms of the differences in final velocity fields between adjacent cases, and convergence rates defined as $\log_2$ of the ratios of these differences.

Base Grid Sizes, Convergence Rates						
16–32	Rate	32–64	Rate	64–128	Rate	128–256
0.407349	0.8598	0.224456	1.4026	0.0848984	1.5410	0.0291751

TABLE 5.2

Convergence rates for the inviscid vortex merger problem of Figure 5.1.

These derivative stencils are used for computing residuals and for obtaining divergence and gradient in the projection formula. Note that D is no longer equal to $-G^T$. This means that the adaptive projection is no longer quite orthogonal, and we have to add a slight correction to DV in (4.2) to make the system solvable. (Specifically, the Neumann boundary conditions used for ϕ make (4.2) singular. The right hand side must discretely integrate to zero in order for a solution to exist. This we insure by subtracting off the discrete sum per unit area of DV, which is small but nonzero in practice.) The alternative, however, would be to use less accurate stencils for either D or G at the interface, which would seriously degrade the accuracy of the algorithm.

We stress that the interface stencils for D and G are the only ones chosen due to decoupling requirements that actually affect the computed solution. The others only affect the multigrid and multilevel convergence rates. Other parts of the fluid algorithm—the Godunov convection step, the parabolic diffusion step with its own multilevel iteration, and the regridding procedure—all use ordinary coupled stencils for restriction and interpolation.

5. Numerical examples. Our first examples concern the convergence properties of the method. It has already been shown in [3] that the single-grid version of a similar scheme gives second-order accuracy in the computed velocity field in both space and time. To confirm that the same is true for the adaptive version, we integrate the smooth flow field with initial velocities

$$\begin{aligned} u &= -\sin^2(\pi x)\sin(2\pi y), \\ v &= \sin(2\pi x)\sin^2(\pi y), \end{aligned} \tag{5.1}$$

up to time $t = 0.5$. The domain is the unit square with a $2^n \times 2^n$ base grid and a centered $2^{n+1} \times 2^{n+1}$ refined patch (identical to the grid layout shown for the next example in Figure 5.1). This and all other examples have a refinement ratio of four between levels. The time step for this example is fixed at $\Delta t = \Delta x_{\min}/2$, where $\Delta x_{\min}$ is the mesh spacing on the finer grid. The boundary conditions are slip walls for Euler flow, no-slip walls for viscous flow. Table 5.1 gives convergence results for base grids ranging from 16 to 128 cells wide.

We next compute convergence rates for the more complicated flow field shown in Figure 5.1, an inviscid vortex merger problem. The initial conditions have two patches of vorticity with radii 0.04 and strengths -1.0 placed in the unit square at

(0.44, 0.50) and (0.56, 0.50). Each patch has uniform vorticity except for a linear ramp 3/256 wide down to zero vorticity at the edge—the radius of a patch is the distance from the center to the halfway point of the ramp. The initial velocity field is obtained by solving for the stream function associated with the given vorticity field. This is identical to the projection calculation, except that the stream function satisfies a Dirichlet boundary condition.

The flow field is advanced to time $t = 0.01$ with variable time steps determined by the CFL condition given in section 3.2. This time corresponds to roughly one quarter revolution of the vortices about the center of the grid. We have performed these calculations on the same fixed grid configuration used for the previous example, with base grids ranging from 16 to 256 cells wide. Table 5.2 gives convergence rates for this problem, while Figure 5.1 shows that the errors in the velocity field are concentrated in the active parts of the flow field, not at the coarse-fine interface. (Convergence is slower than in Table 5.1 because this in not a smooth problem, but the method still appears to be approaching a reasonable asymptotic rate.)

For our third example (Figure 5.2), we demonstrate the robustness of the method by having a vortex pair pass across the edge of a fixed refined patch. This exercise is more extreme than one would ordinarily encounter in practice, since a regridding criterion would normally cause the refined patch to follow the moving vortices. It is desirable in some cases, however, to have the option of suppressing refinement over part of the domain. This example shows that even a flow field which is barely resolved on the coarse grid experiences little degradation in crossing the coarse-fine interface. Except for the positions, (0.375, 0.44) and (0.375, 0.56), and the opposite vortex signs, the blobs are identical to those in the previous example. No viscous dissipation is used.

Figure 5.3 illustrates the algorithm with adaptive regridding. A 64×64 base grid is refined twice, by a factor of four each time, so the finest level has resolution equivalent to a single 1024×1024 grid. Every 10 time steps grids are re-allocated according to a procedure based on second derivatives of the velocity field. In the initial conditions, four patches of vorticity with radii 0.025 are placed in the unit square at (0.5, 0.5), (0.5, 0.575), and the two 120° rotations of this position. As before, the edge of each patch has a ramp 3/256 wide down to zero vorticity.

Note how well the Godunov convection scheme preserves fine details of the flow field, even in the highly stretched regions near the vortex core. These details are too small to even be represented on the coarse grid. Consider also that the three-way symmetry of the field is not disrupted either by the fixed orientation of the grid lines or by the frequent and asymmetric redistribution of fine grids covering the detailed region. This observation, along with behaviors shown in Figures 5.1 and 5.2, provide evidence that the accuracy of the simulation is determined by the fine grid resolution, and is not significantly lowered by the distant presence of coarser grids.

While we will not give a detailed analysis of the error behavior of the scheme under regridding, it is worthwhile to note that third-order biquadratic interpolation has proved useful when filling newly-refined regions. Though a simpler bilinear scheme is second-order, experiments like that of Figure 5.3 showed spurious generation of vorticity in new fine grids filled by the bilinear formula.

Our final example (Figure 5.4) illustrates the adaptive algorithm in a viscous problem, the spin-down of a single off-center vortex. The initial condition is a Gaussian

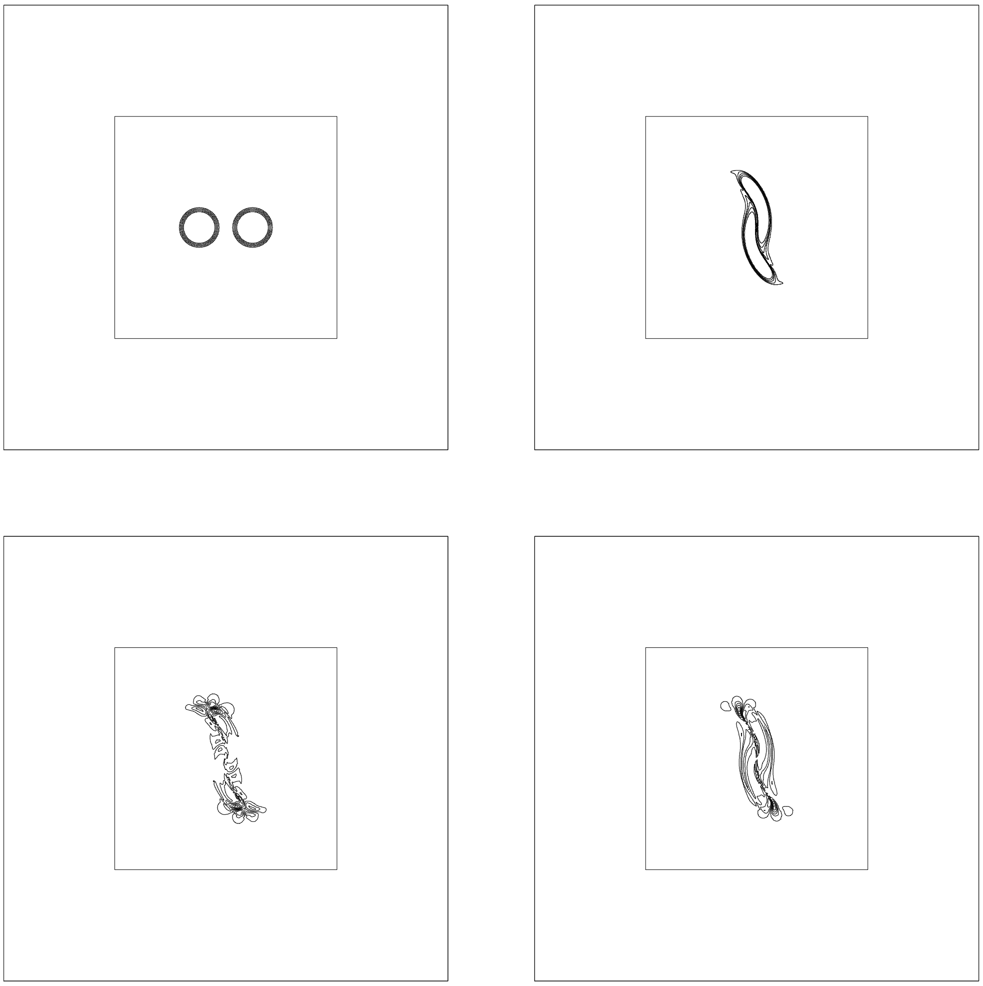

FIG. 5.1. *Pair of vortices merging within a fixed fine grid. The first panel shows the two vortices at time 0, the second at a time where they have completed one quarter revolution, both plotted from a grid configuration with a 256×256 base grid and a 512×512 refined patch. The third and fourth panels show the differences in u and v velocity components, respectively, between the final flow field and one computed on a coarser grid pair with a 128×128 base grid.*

vorticity distribution of the form

$$\omega = \frac{1}{\sigma^2} e^{-\frac{(x-x_0)^2+(y-y_0)^2}{2\sigma^2}} \tag{5.2}$$

for $x_0 = 0.4$, $y_0 = 0.5$, $\sigma = 0.04$. This distribution is scaled to give unit velocity at unit distance. We make the viscous term as small as can be effectively modeled at the fine grid scale, $\varepsilon^{-1} = 15000$, so the effective Reynolds number is over 40000.

Two runs are compared. The first has a 256×256 coarse grid and a single level of refinement, so the effective fine resolution is the same as in the preceding inviscid example. We show the vorticity fields after 6000 and 14000 time steps. The second run has only a 128×128 coarse grid, again with a single level of refinement, with the error tolerance relaxed so as to require refinement of roughly the same portion of the

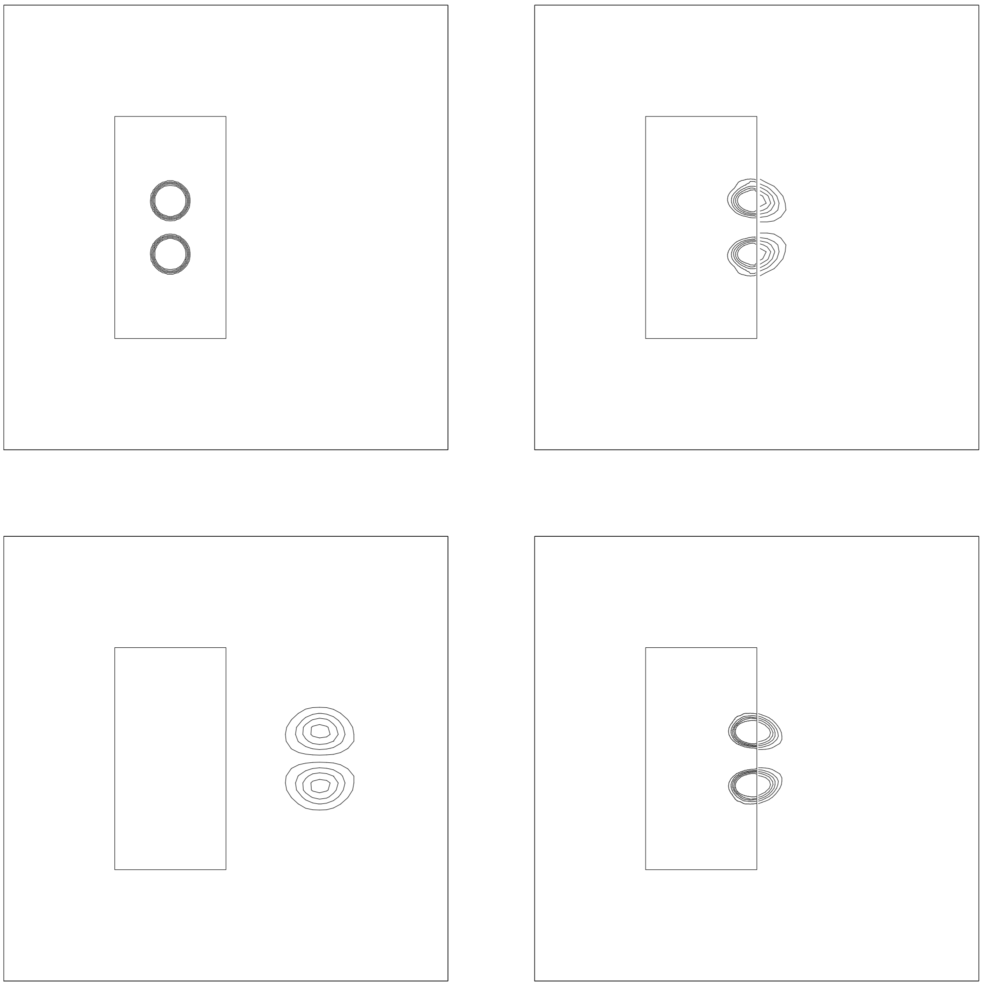

FIG. 5.2. *Vortex pair passing across the edge of a fixed fine grid. The first three panels show results from a 64×64 base grid with a 64×128 refined patch. The pair moves cleanly across the interface, even though it is inadequately resolved on the coarse grid. The last panel is from a similar calculation with a 128×128 base grid.*

domain. After 7000 time steps this calculation has reached approximately the same time value ($t = 0.59201$) as step 14000 of the finer run ($t = 0.59263$). Note that at this time level the two boundary layers which have not fully separated show almost identical development, while the two unstable separated boundary layers have begun to evolve along divergent paths.

Due to the large part of the domain which must be refined, this example problem derives less benefit from adaptivity than the inviscid example. At the outset the fine calculation refines 19% of the domain, the coarse one 31%, running at 17.1 and 19.7 μsec per cell, respectively, on one processor of a Cray YMP. The times required to advance a single time step are therefore 4.6 and 1.9 seconds, respectively.

By time step 7000 the coarse run has refined fully half of the domain and has become significantly slower (up to 29 μsec per cell) through increased communication overhead, reduced vector lengths, and slower multilevel convergence, all related to the

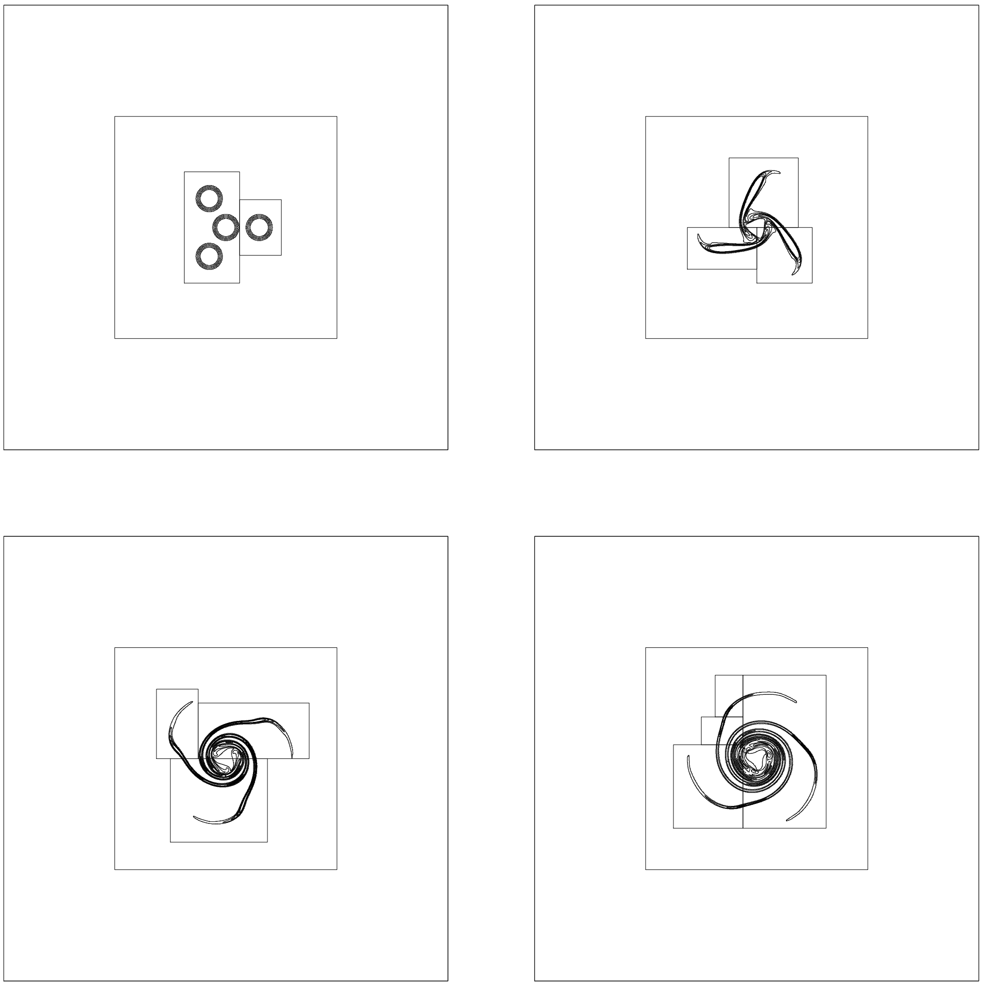

FIG. 5.3. *Adaptive simulation of a four-way vortex merger problem, showing contours of vorticity.*

more complex grid structure. Somewhat after step 9000, when more than two thirds of the domain is refined, the disturbed region near the boundary has grown to the point that fine grids sporadically bridge the gap to the central vortex. Frequent refinement and coarsening in this clear region introduce errors which grow large enough to contaminate the plotted solutions.

Since adaptivity had become a liability by this point, we restarted the computation from step 8000 with the domain fully refined. At 12.4 μsec per cell, 3.5 seconds per time step, the single grid calculation was faster than the later parts of the adaptive calculation, though it required nearly twice as much memory. The final plot shows this single grid run at time step 14000 ($t = 1.1914$), by which time the central vortex has completed nearly one revolution around the center of the box.

Though adaptivity was not helpful during the later stages of the coarse calculation, the initial stages ran almost twice as fast as the single grid method, while the initial stages of the fine calculation ran over three times as fast as would be expected if its entire domain had been refined. The larger grids and less constrained grid place-

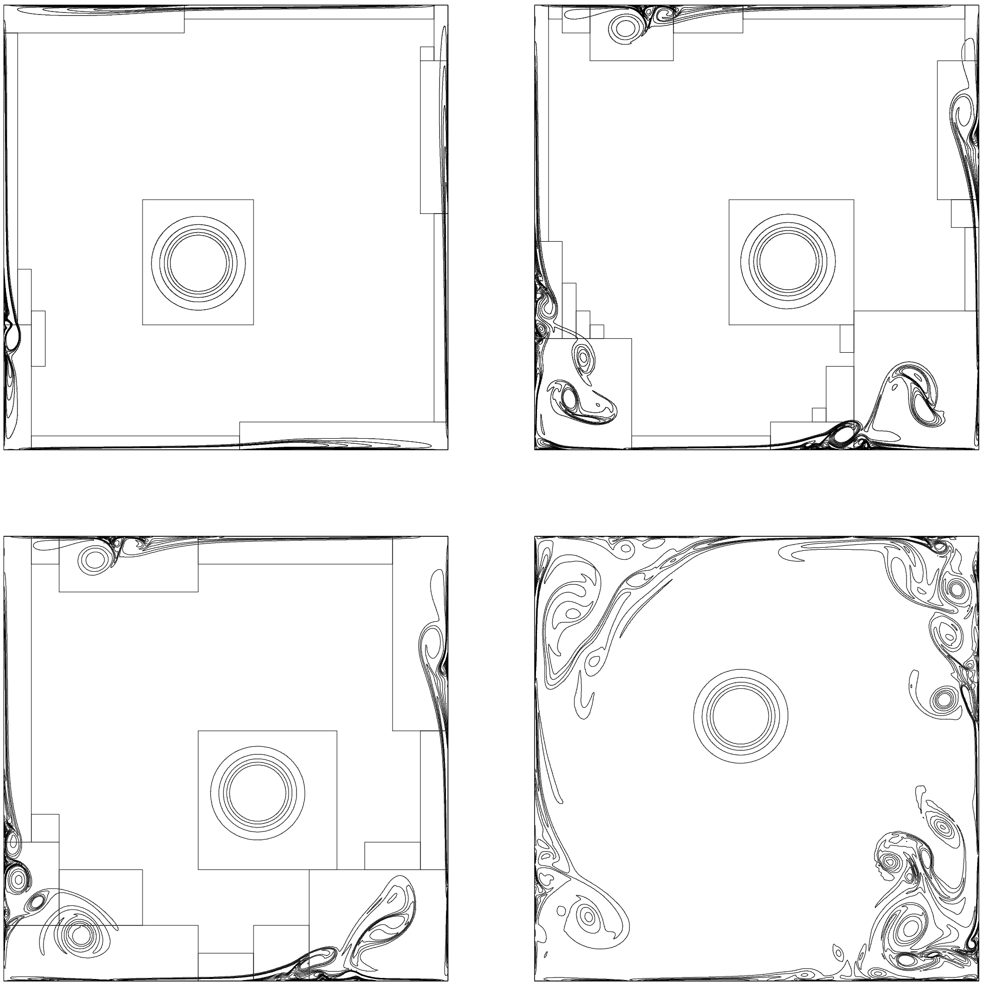

FIG. 5.4. *Viscous boundary effects generated by an off-center vortex. The top plots are from a run with an effective 1024×1024 resolution at the fine level. The bottom plots are from a coarser run with only 512×512 resolution at the fine level. In the final frame the domain is entirely refined.*

ment characterizing the finer calculation also suggest that useful refinement could be maintained in this case even for later stages in the evolution of the flow. In more compact flow configurations like the inviscid example the adaptive advantage is much greater. We therefore conclude that the adaptive algorithm is competitive so long as less than half of the domain must be fully refined, and becomes more competitive for calculations with finer grids.

6. Conclusions and Future Plans. In this paper we have presented an adaptive projection algorithm for the incompressible Navier-Stokes equations. The adaptive formulation requires differencing of the various terms across coarse-fine grid interfaces, efficient solution algorithms on the adaptive grid hierarchy, and a mechanism for clustering the refined patches in interesting parts of the flow field. We have demonstrated that the discretization remains second-order in the presence of refined grids, and that flow patterns can cross over grid interfaces without undue distortion.

The decoupling of adjacent grid points in the projection stencils introduces some complications into the multilevel iteration, but does not degrade the performance of the algorithm as a whole. An extension of the decoupled multigrid scheme to single-grid problems with variable density is presented in [15]. Lai [16] contends, however, that in projection problems with localized source terms the decoupling creates more serious difficulties. In our present development efforts we are therefore replacing the decoupled projection with the coupled but non-idempotent forms presented in [2] and [17]. Associated complications, particularly the interaction of the non-idempotent projections with a new time-stepping scheme, have not as yet been entirely resolved.

The computer program that generated the examples for the previous section can be considered a prototype. Not only was it our first attempt at an adaptive incompressible solver, it was also our first major programming effort in the language C++. Recent work has centered on writing an entirely new fluids program that will overcome the limitations of this first model, both in programming technology and in the ability to simulate more complex physical systems. Completed goals include simulating a wider variety of boundary conditions, advancing fine grids at smaller time steps than coarse grids, and modeling of variable-density and three-dimensional flow fields. Early results of these efforts are presented in [1]. Viscous solution capability and a staggered-grid projection for the Godunov edge velocities are still under development. Longer-term goals include the addition of combustion effects, front tracking, and treatment of more complex flow geometries to the basic algorithm.

REFERENCES

[1] A. S. Almgren, J. B. Bell, P. Colella, and L. H. Howell, *An adaptive projection method for the incompressible Euler equations*, in 11th AIAA Computational Fluid Dynamics Conference, Orlando, July 6–9, 1993.

[2] A. S. Almgren, J. B. Bell, and W. G. Szymczak, *A numerical method for the incompressible Navier-Stokes equations based on an approximate projection*, Tech. Report UCRL-JC-112842, LLNL, January 1993.

[3] J. B. Bell, P. Colella, and H. M. Glaz, *A second-order projection method for the incompressible Navier-Stokes equations*, J. Comput. Phys., 85 (1989), pp. 257–283.

[4] J. B. Bell, P. Colella, and L. H. Howell, *An efficient second-order projection method for viscous incompressible flow*, in 10th AIAA Computational Fluid Dynamics Conference, Honolulu, June 24–27, 1991.

[5] J. B. Bell and D. L. Marcus, *A second-order projection method for variable-density flows*, J. Comput. Phys., 101 (1992), pp. 334–348.

[6] J. B. Bell, J. M. Solomon, and W. G. Szymczak, *A second-order projection method for the incompressible navier stokes equations on quadrilateral grids*, in 9th AIAA Computational Fluids Dynamics Conference, Buffalo, June 14–16, 1989.

[7] M. J. Berger and P. Colella, *Local adaptive mesh refinement for shock hydrodynamics*, J. Comput. Phys., 82 (1989), pp. 64–84.

[8] M. J. Berger and J. Oliger, *Adaptive mesh refinement for hyperbolic partial differential equations*, J. Comput. Phys., 53 (1984), pp. 484–512.

[9] M. J. Berger and I. Rigoutsos, *An algorithm for point clustering and grid generation*, IEEE Trans. Systems, Man and Cybernet., 21 (1991), pp. 1278–1286.

[10] W. L. Briggs, *A Multigrid Tutorial*, SIAM, Philadelphia, 1987.

[11] S. C. Caruso, J. H. Ferziger, and J. Oliger, *Adaptive grid techniques for elliptic fluid-flow problems*, Tech. Report CLaSSiC-85-10, Stanford U., 1985.

[12] P. Colella, *A direct Eulerian MUSCL scheme for gas dynamics*, SIAM J. Comput., 6 (1985), pp. 104–117.

[13] ———, *Multidimensional upwind methods for hyperbolic conservation laws*, J. Comput. Phys., 87 (1990), pp. 171–200.

[14] F. H. Harlow and J. E. Welch, *Numerical calculation of time-dependent viscous incompressible flow of fluids with free surfaces*, Physics of Fluids, 8 (1965), pp. 2182–2189.

[15] L. H. Howell, *A multilevel adaptive projection method for unsteady incompressible flow*, in

6th Copper Mountain Conference on Multigrid Methods, Copper Mountain, CO, April 4–9 1993.
[16] M. F. Lai, *A Projection Method for Reacting Flow in the Zero Mach Number Limit*, PhD thesis, University of California at Berkeley, 1993.
[17] M. F. Lai, J. B. Bell, and P. Colella, *A projection method for combustion in the zero Mach number limit*, in 11th AIAA Computational Fluid Dynamics Conference, Orlando, July 6–9, 1993.
[18] S. F. McCormick, *Multilevel Adaptive Methods for Partial Differential Equations*, SIAM, Philadelphia, 1989.
[19] M. L. Minion, *A note on the stability of Godunov projection methods*, Tech. Report 95-002, Courant Mathematics and Computing Laboratory, January 1995.
[20] J. C. Strikwerda, *Finite difference methods for the Stokes and Navier-Stokes equations*, SIAM J. Sci. Stat. Comput., 5 (1984), pp. 56–67.
[21] P. Wesseling, *An Introduction to Multigrid Methods*, Wiley, New York, 1992.