Journal of Computational Physics **180,** 183–199 (2002)
doi:10.1006/jcph.2002.7087

# Analysis of an Exact Fractional Step Method

Wang Chang,* Francis Giraldo,† and Blair Perot*

**Department of Mechanical and Industrial Engineering, University of Massachusetts, Amherst, Massachusetts 01003-2210; and †Naval Research Laboratory, Monterey, California 93943-5502*
E-mail: perot@ecs.umass.edu

Received June 1, 2001; revised January 8, 2002

An exact fractional step or projection method for solving the incompressible Navier–Stokes equations is analyzed. The method is applied to both structured and unstructured staggered mesh schemes. There are no splitting errors associated with the method; it satisfies the incompressibility condition to machine precision and reduces the number of unknowns. The exact projection technique is demonstrated on a two-dimensional cavity flow and a multiply connected moving domain with a free surface. Its performance is compared directly to classic fractional step methods and shown to be roughly twice as efficient. Boundary conditions and the relationship of the method to streamfunction-vorticity methods are discussed. © 2002 Elsevier Science (USA)

*Key Words:* fractional step method; projection method; incompressible; numerical; unstructured; Navier–Stokes.

## 1. INTRODUCTION

The incompressible Navier–Stokes equations in primitive variables are well-known and given by the equations

$$\begin{aligned} \frac{\partial \mathbf{u}}{\partial t} + \mathbf{u} \cdot \nabla \mathbf{u} &= -\nabla p + \nu \nabla^2 \mathbf{u}, \\ \nabla \cdot \mathbf{u} &= 0, \end{aligned} \tag{1}$$

where $\mathbf{u}$ is the velocity vector, $p$ is the pressure (divided by density), and $\nu$ is the kinematic viscosity. The numerical solution of these equations represents a difficult computational challenge. The problem stems largely from the pressure term in the momentum equations, which couples the momentum equations and which must be implicitly updated for incompressibility to be satisfied. The resulting discrete system of equations for the pressure and two or three velocity components is elliptic, large, indeterminate, and generally difficult to solve efficiently. Projection methods or fractional step methods were conceived to remove most of these computational difficulties, and to allow the system to be solved as a series of individual uncoupled advection–diffusion equations for each of the velocity components and as a Poisson equation for the pressure or for a pressurelike variable. Fractional step methods allow existing computational machinery for advection–diffusion equations and the Poisson equation to be rapidly applied to solving the Navier–Stokes equations. The price is a certain loss of temporal accuracy in the solution, which is the subject of many articles [1–3] and some debate about how boundary conditions should be specified [4–6], which may now be largely resolved [7, 8], and some difficulties if convection is treated implicitly [9]. These are minor issues in most respects compared to the computational savings and code simplicity that the method produces, and so it is not surprising that fractional step methods are a popular technique for solving the incompressible Navier–Stokes equations.

This work focuses on the application of a new type of fractional step (or projection) method that is exact in the sense that it does not display any "splitting error," and which may result in reduced computational costs. It is believed that the method is quite general and applicable to systems based on almost any spatial discretization scheme. When appropriate, general statements about the method are made. However, some of the specific details of the method are intricate, and for the sake of concreteness, the formal analysis and computational results that are presented herein focus on exact fractional step methods in the context of staggered mesh spatial discretizations. Staggered mesh discretizations are very frequently used in conjunction with fractional step methods because they do not require pressure coupling terms to stabilize the pressure solution.

Section 2 presents a review of classic fractional step methods, since this is the standard against which this new methodology is compared. Section 3 discusses some important details of the staggered mesh spatial discretization, which are necessary for the reader to fully understand the motivation behind the exact projection method discussed in Section 4. Section 5 shows an example of the method applied to a complex flow problem, and Section 6 discusses the accuracy and efficiency of the method. A discussion of the exact projection method and its relation to other numerical techniques is presented in Section 7.

## 2. CLASSIC FRACTIONAL STEP METHODS

Fractional step methods (or projection methods) were originally presented and analyzed as semidiscrete approximations to the Navier–Stokes equations [10, 11]. The semidiscrete approach discretizes the equations in time but leaves spatial derivatives as continuous operators. The appeal of such a formulation is that the analysis is independent of any particular spatial discretization. It can be applied equally well to finite elements, staggered meshes, or spectral methods. The primary disadvantage of the semidiscrete approach (and one which necessitates that the current analysis be applied to the fully discrete system) is that it also leads to a great deal of confusion. Continuous differential operators require boundary conditions whereas discrete operators (basically matrices) do not. If the semidiscrete system is analyzed, then boundary conditions on intermediate variables must be posed and are invariably ambiguous. Intermediate variables in the fully discrete analysis (vectors) are unambiguously defined. Furthermore, continuous differential operators have commutative and associative properties which are rarely shared by their discrete counterparts. This leads to deceptive conclusions about the order of accuracy of the method when it is applied in practice to fully discrete systems [12].

When fully discretized the incompressible Navier–Stokes equations shown in Eq. (1) can be represented in the generic matrix form,

$$\begin{bmatrix} \mathbf{A} & \mathbf{G} \\ \mathbf{D} & \mathbf{0} \end{bmatrix} \begin{bmatrix} u^{n+1} \\ q \end{bmatrix} = \begin{bmatrix} r^n \\ 0 \end{bmatrix} + \begin{bmatrix} bc_1 \\ bc_2 \end{bmatrix}, \tag{2}$$

where $\mathbf{A}$ is the implicit operator for the advection–diffusion part of the momentum equations, $\mathbf{G}$ is the gradient operator, $\mathbf{D}$ is the divergence operator, $r^n$ is the explicit right-hand side of the momentum equations, $bc_1$ is the boundary condition vector for the momentum equations, and $bc_2$ is the boundary condition vector for the incompressibility constraint. Note that $\mathbf{A}$ is a square matrix but $\mathbf{G}$ and $\mathbf{D}$ are not. The goal of any numerical method is to solve for the vector of velocity unknowns $u^{n+1}$ and pressure unknowns $q$. No time level is placed on the pressure unknown to emphasize that this unknown is simply a Lagrange multiplier that is present so that the discrete incompressibility constraint is satisfied. If $r^n$ does not contain an explicit pressure gradient term then $q \approx p^{n+1/2}$. If $r^n$ has a term of the form $-\mathbf{G}p^n$, then $q \approx p^{n+1/2} - p^n$. This latter form can reduce the subsequent fractional step splitting error so that it is second order in time [8, 13].

The advantage of the semidiscrete analysis (independent of any particular spatial discretization) is retained in the fully discrete analysis represented by Eq. (2), since the discrete matrix operators described above can represent any discretization scheme. Eventually, we will need to be much more specific about the exact form of the gradient and divergence operators, but for the purpose of understanding the classic fractional step method, they may remain quite generic.

The detailed structure of the matrix $\mathbf{A}$ depends a great deal on the specifics of the time advancement scheme and the mesh. The most common formulation is to keep convection explicit and require diffusion to be implicit to avoid the stringent time-step restrictions associated with that term. In that case, $\mathbf{A}$ can be written as $\mathbf{I}/\Delta t - \nu\mathbf{L}$, where $\mathbf{I}$ is the identity matrix, $\mathbf{L}$ is the Laplacian operator, and $\nu$ is the kinematic viscosity. If the mesh is Cartesian and staggered, or unstructured and colocated, then velocity components are uncoupled in each direction and $\mathbf{A}$ is block diagonal with one block for each coordinate direction. If a finite element discretization is assumed, then the identity matrix becomes a mass matrix, $\mathbf{A} = \mathbf{M}/\Delta t - \nu\mathbf{L}$, where $\mathbf{M}$ is the mass matrix. If convection is also made implicit then $\mathbf{A} = \mathbf{M}/\Delta t - \nu\mathbf{L} + \mathbf{N}$, where $\mathbf{N}$ is a linearized convection operator. The convection operator $\mathbf{N}$ can be specified so that $\mathbf{A}$ is still block diagonal, but this reduces the temporal accuracy of the method to first order.

As shown in Refs. [7, 14], fractional step methods are related to the block LU factorization of Eq. (2),

$$\begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{D} & -\mathbf{D}\mathbf{A}^{-1}\mathbf{G} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{A}^{-1}\mathbf{G} \\ & \mathbf{I} \end{bmatrix} \begin{bmatrix} u^{n+1} \\ q \end{bmatrix} = \begin{bmatrix} r^n \\ 0 \end{bmatrix} + \begin{bmatrix} bc_1 \\ bc_2 \end{bmatrix}. \tag{3}$$

While Eq. (3) is exact, and is sometimes referred to as the Uzawa method, it is not the basis for a practical computational method because computing the inverse of the matrix $\mathbf{A}$ is very expensive. Practical methods for solving Eq. (2) usually approximate $\mathbf{A}^{-1}$ with the matrices $\mathbf{B}_1$ and $\mathbf{B}_2$, and then (3) can be rewritten as

$$\begin{aligned} \mathbf{A}w &= r^n + bc_1, \\ \mathbf{D}\mathbf{B}_1\mathbf{G}q &= \mathbf{D}w, \\ u^{n+1} &= w - \mathbf{B}_2\mathbf{G}q. \end{aligned} \tag{4}$$

Different approximations to the two matrix inverses result in different classes of fractional step methods. The classic fractional step method approximation is $\mathbf{B}_1 = \mathbf{B}_2 = \Delta t\mathbf{I}$. If $q \approx p^{n+1/2}$ this results in a first-order error term in the momentum equation, and if $q \approx p^{n+1/2} - p^n$ this results in a second-order error term in the momentum equation. If the approximate inverses are equal ($\mathbf{B}_1 = \mathbf{B}_2$), then all the error in the approximation appears in the momentum equation and discrete continuity is satisfied exactly. However, if the approximate inverses are chosen to be different (such as $\mathbf{B}_1 = \Delta t\mathbf{I}$ and $\mathbf{B}_2 = \mathbf{A}^{-1}$), then the splitting error term can be moved entirely from the momentum equation so that it appears only in the incompressibility constraint. It is not entirely clear which type of error (momentum or divergence error) is the least detrimental. In all probability, the answer to that question is problem dependent. However, the most frequent assessment is that mass should be conserved and the splitting error is best placed in the momentum equation (with $\mathbf{B}_1 = \mathbf{B}_2$). Antidotal evidence which confirms the detrimental nature of mass conservation error (versus momentum errors) is found in Section 6 of this paper, where we look at the error that results from solving the elliptic Possion equation with an iterative method to some small but finite error tolerance. Higher order splitting methods can be constructed by constructing higher order approximations to $\mathbf{A}^{-1}$ [7]. It is also important to note that the intermediate variable, $w$, does not require boundary conditions; it is simply a way of rewriting Eq. (3).

Fractional step methods are also referred to as projection methods because the system of equations given by (4) can be interpreted as projection in velocity space. The "trial velocity," $w$, is an approximate solution to the momentum equations, but because the trial velocity is obtained with an explicit pressure it cannot satisfy the incompressibility constraint at the next time level. The Poission equation (second equation) determines the minimum perturbation that will make the trial velocity incompressible, and the final equation perturbs the trial velocity to obtain the final velocity. Only if $\mathbf{B}_1 = \mathbf{B}_2 = \mathbf{A}^{-1}$ is the classical fractional step method free from splitting errors. Because the exact projection method presented in Section 4 is constructed in a manner entirely different from classic fractional step methods it remarkably avoids the need for constructing $\mathbf{A}^{-1}$ or any approximations to $\mathbf{A}^{-1}$ while eliminating any splitting error.

## 3. STAGGERED MESH DISCRETIZATION

Fractional step methods are a technique for dealing with the pressure and the incompressibility constraint and can be applied to any discretization scheme. However, for clarity we need to discuss a particular discretization scheme. Staggered mesh schemes of the type introduced by Harlow & Welch [15] are frequently used in conjunction with fractional step methods. The exact projection method and accompanying examples will therefore be presented in the context of the staggered mesh discretization. We consider both unstructured two-dimensional and three-dimensional staggered meshes in order to keep the discussion as general as possible. The unstructured analysis is, of course, applicable to structured Cartesian meshes as a subclass.

Staggered mesh schemes display a number of mathematical and physical properties which make them attractive for solving the incompressible Navier–Stokes equations. Foremost in the context of incompressibility is the fact that these discretizations do not display spurious pressure oscillations and therefore do not require pressure coupling terms. In addition, they have low memory requirements, are computationally efficient, and display a number

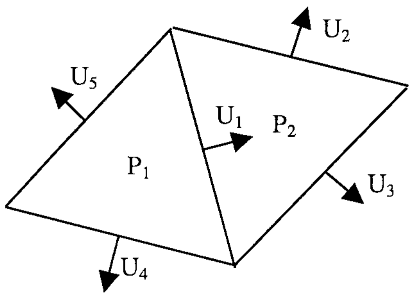

**FIG. 1.** Location of face normal velocity components and pressure unknowns in a two-dimensional unstructured mesh.

of interesting conservation properties, including conservation of mass, momentum, kinetic energy, and vorticity [16]. Unstructured staggered mesh schemes are still under development [17, 18] but can also possess very attractive numerical properties [19].

The common aspect of both Cartesian and unstructured staggered mesh methods is that the primary unknowns of the method are the normal velocity components, located at mesh faces. Pressure and other scalar variables are located at the center of mesh cells in a manner akin to standard finite volume methods. The velocity vector in the mesh cell must be reconstructed from the face normal components, and it is the different reconstruction methods that lead to different types of staggered mesh methods with different numerical properties. The location of variables in a two-dimensional staggered mesh is shown in Fig. 1.

On a staggered mesh, the incompressibility constraint is imposed on average in each mesh cell. Gauss's divergence theorem can be used to convert the incompressibility constraint into integration over the cell faces:

$$0 = \int_{\text{cell}} \nabla \cdot \mathbf{u}\, dV = \sum^{\substack{\text{cell}\\\text{faces}}} \int_{\text{face}_i} \mathbf{u} \cdot \mathbf{n}\, dS. \tag{5}$$

If the normal velocity is assumed to vary linearly on the face (a second-order approximation), then the integration can be performed analytically and the incompressibility constraint becomes a simple summation over the primary velocity unknowns,

$$0 = \sum^{\substack{\text{cell}\\\text{faces}}} \hat{u} A_{\text{f}}, \tag{6}$$

where $A_{\text{f}}$ is the cell face area and the circumflex (hat) on the velocity indicates that this is the normal velocity component pointing *out* of the cell in question. If Eq. (6) is applied to every cell in the mesh at the final time level, then a matrix system is obtained, $\mathbf{D}U_{\text{f}}^{n+1} = 0$, where $U_{\text{f}} = uA_{\text{f}}$ is the mass flux through the face. The matrix $\mathbf{D}$ is sparse and nonsquare. For the simple mesh shown in Fig. 1, the matrix $\mathbf{D}$ has the form

$$\mathbf{D} = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 \\ -1 & 1 & 1 & 0 & 0 \end{bmatrix}. \tag{7}$$

In general $\mathbf{D}$ has nonzero entries that are either $+1$ or $-1$ ($-1$ if the face normal vector points into the cell). It has $N_{\text{f}}$ columns, where $N_{\text{f}}$ is the number of faces in the mesh, and $N_{\text{c}}$ rows, where $N_c$ is the number of cells. Every column of $\mathbf{D}$ that corresponds to an internal mesh face will have a single $+1$ and a single $-1$ entry. Every column of $\mathbf{D}$ that corresponds to a mesh face on the boundary of the computational domain will only have one nonzero entry, and if the normal vector is assumed to point out of the computational domain that entry will be $+1$. The matrix $\mathbf{D}$ will behave like a discrete divergence operator, but it should be noted that it has been stripped of its geometric variables (the face area and cell volume). The negative transpose of the divergence operator is a gradient operator, $\mathbf{G} = -\mathbf{D}^T$. On interior faces this operator takes the difference of the two neighboring cell values. On boundary faces it takes the negative of the value of the interior cell. The gradient operator for Fig. 1 is

$$\mathbf{G} = \begin{bmatrix} -1 & 1 \\ 0 & -1 \\ 0 & -1 \\ -1 & 0 \\ -1 & 0 \end{bmatrix}. \tag{8}$$

A discrete system, in the form of Eq. (2), is constructed by discretizing the line integral of the momentum equation at each cell face [19, 20] and constructing a discrete equation for the time evolution of $\int \mathbf{u} \cdot \mathbf{n}\, dl$. This gives a single equation at each mesh face; one equation for each velocity unknown. For Cartesian meshes and unstructured meshes using cell circumcenters, the line connecting the two neighboring cell centers is normal to the face, and the time derivative produces a diagonal matrix contribution to $\mathbf{A}$. Each element of the diagonal matrix is $\frac{1}{\Delta t}\frac{L_f}{A_f}$, where $L_f$ is the distance between neighboring cell centers. For unstructured staggered meshes using a cell position other than the cell circumcenter, the discretization of the time derivative results in a mass matrix. The mass matrix is similar, but not identical, to a finite element mass matrix. In either case, the matrix $\mathbf{A}$ is square and has a size $N_f$ by $N_f$. It should also be noted that the line integral of the pressure gradient in the momentum equation is exactly equal to the difference of the pressure at the two cell centers, so the pressure discretization is, in some sense, exact for staggered mesh formulations.

With this discretization scheme, the matrix equations that must be solved are written as

$$\begin{bmatrix} \mathbf{A} & \mathbf{G} \\ -\mathbf{D} & \mathbf{0} \end{bmatrix} \begin{bmatrix} U_f^{n+1} \\ q \end{bmatrix} = \begin{bmatrix} r^n \\ 0 \end{bmatrix} - \begin{bmatrix} q_b \\ 0 \end{bmatrix}. \tag{9}$$

It should be noted that for simplicity, the matrix system as it is now posed includes all the mesh faces, including those on the boundary that may actually be known. This eliminates any explicit boundary condition treatment of the incompressibility constraint. We have also chosen (purely for aesthetic reasons) to multiply the divergence-free constraint by $-1$. This means that if $\mathbf{A}$ is symmetric (explicit convection), then the entire system is also symmetric. The entire system is also square and has a size $N_c + N_f$ by $N_c + N_f$. On a boundary where the normal velocity is not known, such as a moving free surface or an outflow boundary, it is often convenient to specify the boundary pressure. This boundary pressure is located at the cell faces, or more logically in infinitesimally thin virtual boundary cells that surround the domain. These boundary pressures are explicitly represented in Eq. (9) by the vector $q_b$. The minus sign on this term assumes that boundary normals point outward. This notation also implies that the right-hand-side vector $r^n$ now only has contributions from the velocity field, not the pressure field. A more detailed discussion of specific staggered mesh discretizations can be found in Refs. [17–21].

## 4. EXACT PROJECTION METHOD

Classic fractional step methods can never be exact because each conservation equation (momentum, then continuity) is satisfied sequentially. Satisfaction of one equation produces error in the solution of the other equation. One way to enforce both mass and momentum conservation and evolve the system at the same time (and without any associated error) is to assume that the solution lies in a function space that is already incompressible. In the finite element context this is equivalent to assuming basis functions which already satisfy incompressibility. However, standard conforming basis functions which satisfy the incompressibility constraint have never been particularly popular because they require basis functions which are at least fourth-order polynomials [22]. Nonconforming (discontinuous) finite element spaces have been proposed [23] that have some similarities to the proposed method. In this section, we show that low-order finite volume basis functions can be easily constructed which satisfy the constraint of *discrete* incompressibility.

The method relies on the fact that matrix **D** is wider than it is tall and therefore defines a null space. The null space is the maximal set of linearly independent vectors that equal zero when they are multiplied by the matrix **D**. The null space of **D** is therefore another nonsquare matrix with a height $N_{\mathrm{f}}$. There are many possible null spaces, but we define an easily constructed null space **C**. In two dimensions, **C** has $N_{\mathrm{n}}$ columns, where $N_{\mathrm{n}}$ is the number of nodes (vertices) in the mesh. In three dimensions, it has $N_{\mathrm{e}}$ columns, where $N_{\mathrm{e}}$ is the number of edges. The matrix **C** acts like a discrete curl operator. In two dimensions it has two entries in every row, one $+1$ and one $-1$. The $+1$ corresponds to the node 90 degrees from the normal vector at the face in question, and the $-1$ corresponds to the node that is $-90$ degrees from the normal vector. In three dimensions there is one entry for each edge which forms the face in question, and the entry is $+1$ if that edge points counterclockwise with respect to the face normal, and it is $-1$ if the edge points clockwise with respect to the face normal. In the interior of the domain, both the edge directions and face normals are chosen arbitrarily from the two possible choices for each. There is no logical way to choose these directions in three dimensions. The curl matrix, **C**, for the mesh shown in Fig. 1 is

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & -1 & 0 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 0 & 1 & -1 \\ -1 & 0 & 0 & 1 \end{bmatrix}. \tag{10}$$

Using the curl matrix, we can guarantee by construction that $\mathbf{DC} = 0$. The result is always zero, because each row is a closed summation over differences, and the differences all cancel out with one positive and one negative. The presence of the matrix **C** is not exclusive to staggered mesh discretizations. All discrete divergence operators have a null space. The staggered mesh discretization just makes the construction of the null space very simple and makes it clear that the null space, **C**, is a discrete curl operator. An example of the curl operator for irregular quadrilateral meshes is discussed in Hyman and Shashkov [24]. The basic idea of using a null space dates back to Hall *et al.* [25].

The discrete curl operator allows us to define a discrete streamfunction vector. We will assume that $U_{\mathrm{f}} = \mathbf{C}s$, where $s$ is the discrete streamfunction. In two dimensions the streamfunction points out of the two-dimensional plane and is located at the mesh nodes. In three dimensions the discrete variable, $s$, actually represents the streamfunction vector integrated along the edge, $s = \int_{\text{edge}} \Psi \cdot d\mathbf{l}$. Irrespective of its physical interpretation, this mathematical construction guarantees discrete incompressibility ($\mathbf{D}U_{\text{f}} = \mathbf{DC}s = \mathbf{0}s = 0$).

It is important that the curl operator be complete. That is, its columns should span the entire null space which has a size $N_{\text{f}} - N_{\text{c}}$. In two dimensions, one of Euler's many formulas tells us that $N_{\text{f}} - N_{\text{c}} = N_{\text{n}} - 1 + N_{\text{h}}$, where $N_{\text{h}}$ is the number of holes in a possibly multiconnected domain. In three dimensions we have $N_{\text{f}} - N_{\text{c}} = N_{\text{e}} - N_{\text{n}} - 1 + N_{\text{h}}$. Since $\mathbf{C}$ has $N_{\text{n}}$ columns in two dimensions and $N_{\text{e}}$ columns in three dimensions there are always more columns than necessary to span the null space if the domain does not contain holes. The $-1$ in the 2D formula is an indication that the streamfunction is only defined up to an arbitrary constant; the $-N_{\text{n}} - 1$ in 3D is a reflection that the streamfunction in three dimensions is only defined up to an arbitrary gradient. These properties of the discrete streamfunction are true for the continuous streamfunction as well. When holes are present in the domain, the presence of $N_{\text{h}}$ indicates that certain other constraints are automatically satisfied by this particular construction. In particular, the net mass flux into or out of a hole in the domain must always be zero using this construction for the velocity.

Automatically satisfying the incompressibility constraint is only half the goal of the exact projection procedure. The other goal is to eliminate the pressure. This is done by noting that the gradient operator is the negative transpose of the divergence operator. The gradient operator is therefore the null space of another matrix operator, $\mathbf{R}$, ($\mathbf{0}^T = (\mathbf{DC})^T = \mathbf{C}^T\mathbf{D}^T = -\mathbf{RG}$). The rotation operator $\mathbf{R} = \mathbf{C}^T$ is another type of curl operator. It has $N_{\text{f}}$ columns and $N_{\text{e}}$ (or $N_{\text{n}}$ in 2D) rows. Each column has one nonzero entry for every edge (in 3D) or node (in 2D) which the face touches. The sign is given by the same convention as the curl operator. For interior edges, the rotation operator defines a complete counterclockwise circuit around and edge (in 3D) or node (in 2D). At boundary edges or nodes, these circuits are not complete, but begin and end with the boundary faces touching that edge or node.

Applying these operators to the discrete system given by Eq. (9) gives

$$\begin{bmatrix} \mathbf{RA} & \mathbf{RG} \\ -\mathbf{D} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{C}s^{n+1} \\ q \end{bmatrix} = \begin{bmatrix} \mathbf{R}r^n \\ 0 \end{bmatrix} - \begin{bmatrix} \mathbf{R}q_b \\ 0 \end{bmatrix}. \tag{11}$$

The off-diagonal terms in the matrix are now zero, so the exact fractional step method is

$$\begin{aligned} \mathbf{RAC}s^{n+1} &= \mathbf{R}(r^n - q_b), \\ u_{\text{f}}^{n+1} &= \frac{1}{A_{\text{f}}}\mathbf{C}s^{n+1}. \end{aligned} \tag{12}$$

This is an exact projection which inverts both the matrix $\mathbf{A}$ and the Poisson equation ($\mathbf{RC}$) at the same time. There are no approximations to any of the matrices and so there is no splitting error. Note that all the matrix operators involved are sparse and only require nearest-neighbor communication. If $\mathbf{A}$ is symmetric (explicit convection), then the total system is also symmetric. If $\mathbf{A}$ is positive semidefinite (implicit upwind convection), then the total system is also positive semidefinite. So any matrix inversion technique which is appropriate for $\mathbf{A}$ is also appropriate for the total system, $\mathbf{RAC}$. In the example problems described in Section 5, this system is solved using a Jacobi preconditioned conjugate gradient solver, but the exact matrix solution choice is not critical to the exact projection method.

## 4.1. *Comparison with Classical Fractional Step Methods*

Classic fractional step methods require inverting the matrix **A** and then solving a separate Poisson equation for the pressure variable. If the viscosity is small and the convection term is explicit, then the matrix **A** is highly diagonally dominant (due to the time derivative term) and an iterative solution can be obtained in very few iterations. Large viscosity or implicit convection will require more iterations. Cartesian meshes speed the solution by allowing **A** to be decomposed for each velocity direction, and allowing the diffusion to be approximated by a series of tridiagonal matrices in each coordinate direction. The bulk of the computational effort of classic fractional step methods therefore lies in the solution of the Poisson equation. The solution of the Poisson equation can only be simplified in very special geometries (periodic uniform Cartesian meshes). Usually, it requires solution via iterative methods, either Krylov subspace methods (typically conjugate gradients) or multigrid relaxation schemes. The Krylov methods will require on the order of $\sqrt{N_c}$ iterations in 2D and on the order of $N_c$ work per iteration, since there are $N_c$ pressurelike unknowns in the Poisson equation.

The work involved in solving the exact projection method will be very similar to solving a simple Poisson equation if **A** is very diagonally dominant, which it frequently is. However, the number of iterations and the work per iteration now scales with the number of nodes, $N_n$ (in 2D), or the number of edges, $N_e$ (in 3D). Table I shows a comparison of the number of streamfunction unknowns versus the number of pressure unknowns for various dimensions and meshes. This gives a good indication of the relative cost per iteration, and some idea as to the number of iterations required to achieve convergence. Note that the exact projection method is equal or lower in cost per iteration for unstructured meshes or 2D Cartesian meshes but requires more work per iteration for Cartesian meshes in three dimensions.

The number of iterations is also an important issue. The exact method requires slightly more work per iteration because the matrix **A** is imbedded in the inversion. However, the Poisson equation in the classic fractional step method must be converged to very tight tolerances requiring numerous iterations. In the classic fractional step method small errors in the Poisson solution lead to relative large numerical divergence or local mass creation, and to significant errors in the resulting velocity field. It is hypothesized (with some evidence given in Section 6) that far fewer iterations are required for the exact projection method. The exact projection maintains exact incompressibility (local mass conservation), even when the solution is not fully converged. Error appears as error in the vorticity, not the dilatation, and is expected to be less detrimental to the final solution in most cases. Therefore, the convergence tolerances acceptable in the exact projection method are larger (and the number of iterations fewer) than what is required to obtain solutions via the classical fractional step method.

While the exact method introduces a new variable, it also eliminates the need to store the pressure in the domain (however boundary pressures may still be retained). So the overall

**TABLE I**
**Number of Edges (3D) or Nodes (2D) Divided by the Number of Cells**

| | Cartesian | Unstructured |
|---|---|---|
| 2D | 1 | 0.5 |
| 3D | 3 | $\approx$1 |

storage requirements are not expected to be fundamentally different. If pressure is required, it can still be obtained from the exact projection method. However, it is a postprocessing step, which can occur as infrequently as desired and which does not affect the velocity field evolution.

One final advantage of classic fractional step methods may be their conceptual simplicity. However, while the novelty of the exact projection method may make it appear less straightforward, it is not fundamentally more complex to understand or implement than classic fractional step methods.

### 4.2. *Boundary Conditions*

The use of the streamfunction, $s$, as the primary unknown rather than the normal velocity components complicates some boundary conditions and simplifies others. There are two types of boundary conditions on the velocity: the inviscid boundary condition on the normal velocity, and the viscous condition applied to the tangential velocity components. The viscous boundary condition is the boundary condition that disappears when the Euler (inviscid Navier–Stokes) equations are solved and it is not affected by the change in the primary variable. It is the inviscid boundary condition that is of concern in this section.

Boundary conditions in which the streamfunction is known are trivial to implement in the exact projection method. A solid wall (without transpiration) or a stationary free surface is a streamline and can be represented by a constant streamfunction value. A known inflow velocity (or transpiration velocity) can often be analytically converted into a known streamfunction profile (by numerical or analytical integration). Note that where the streamfunction or normal velocity is fixed on the boundaries, the boundary pressure does not affect the solution and is not required. However, outflow boundary conditions and moving free surfaces are easily implemented by fixing the boundary pressure and allowing the boundary streamfunction (and resulting boundary normal velocity) to be left as unknowns. Inflow boundaries (potential inflow) can also be implemented by specifying the dynamic pressure rather than the streamfunction; this allows fluid to be sucked into the domain by a momentum source in the interior. Walls in the interior of the domain (causing multiply connected domains) are the most difficult to specify. These walls have a constant streamfunction value, but that value is unknown and can change with time. An example is the high-Reynolds-number flow perpendicular to a long cylinder. The streamfunction on the cylinder changes as the resulting Von Karman vortex street oscillates back and forth. The system of equations for this boundary condition is created by allowing one streamfunction on the interior wall to remain unknown and specifying that all other streamfunction values on the wall are equal to that unknown value. Careful examination of such a boundary conditions shows that this approach is equivalent to specifying the line integral around the object. Consequently, boundary pressure on interior objects is not required. In summary, pressure boundary conditions can be specified if so desired (often they are very convenient), but they are not required. The second example problem in Section 5 demonstrates most of these boundary conditions, including the difficult case of an interior wall and a moving boundary.

## 5. EXAMPLE PROBLEMS

A test of the exact projection method was performed on two-dimensional cavity flow at a Reynolds number of 100. This is a test case that has been computed extensively in the past and is well understood. An unstructured mesh containing roughly 4024 triangles and

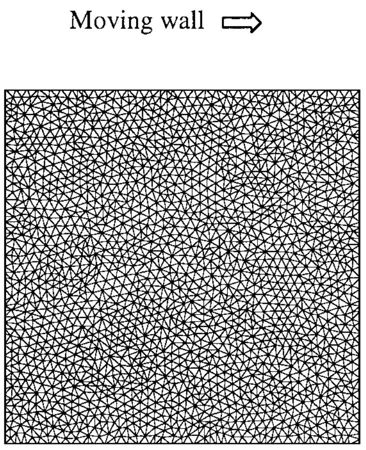

**FIG. 2.** Unstructured mesh and boundary conditions used to calculate driven cavity flow at a Reynolds number of 100.

2013 nodes was used for the calculation and is shown in Fig. 2. Since no fluid leaves the cavity, the streamfunction is zero on the boundary. The primary vortex center was computed to be at (0.6166, 0.7477) when the coordinate system is located at the bottom left corner of the cavity. This compares well with the predictions of Ghia *et al.* [26], who report a vortex center at (0.6172, 0.7344) using $128 \times 128$ Cartesian mesh resolution. Cross sections of the steady state velocity profiles at the two midlines are shown in Fig. 3. Figure 3a shows the horizontal velocity on the line $y = 0.5$, and Fig. 3b shows the vertical velocity on the line $x = 0.5$. The agreement with Ref. [26] is what would be expected from this level of mesh resolution.

The time accuracy of the method was analyzed by looking at the cavity flow start-up process. The flow was initialized with zero velocity and allowed to evolve to a time of 20 using different-size time steps. It takes 1000 time units for the flow to move from the top left to the top right corner of the domain. The total kinetic energy was evaluated at time $= 20$ and its error is plotted in Fig. 4 versus the time step for both the exact and the first-order classic projection method. The exact solution was obtained by extrapolating the results to zero time step. Since the total kinetic energy grows monotonically during this

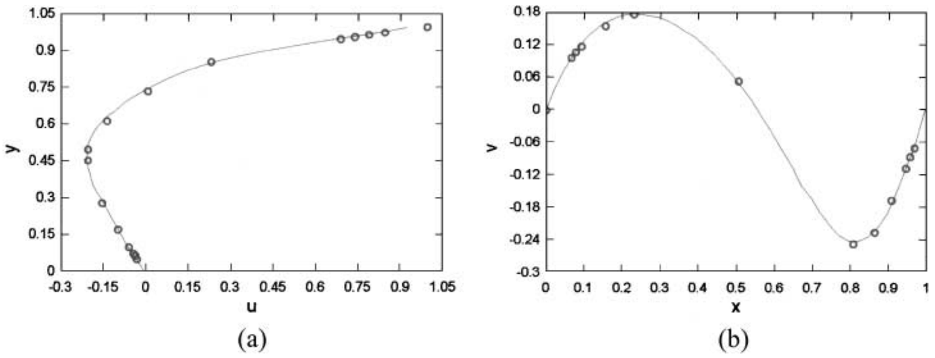

**FIG. 3.** Mean flow predictions along the midlines. Lines are the exact projection method and symbols are the high resolution data from Ghia *et al.* (a) u, Velocity parallel to moving wall; (b) v, velocity perpendicular to the moving wall.

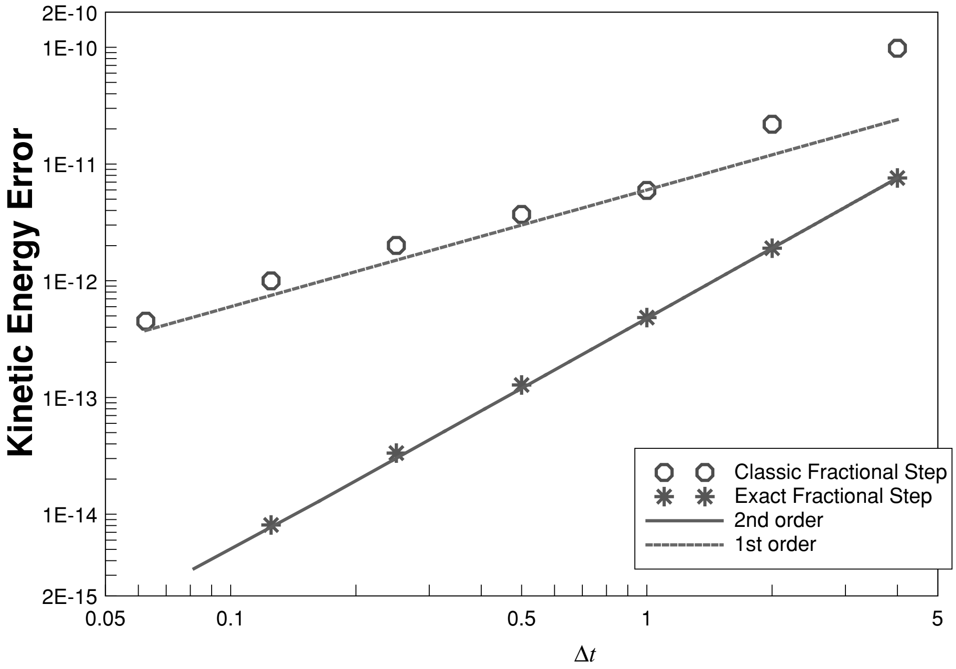

**FIG. 4.** Error in the total kinetic energy at time = 20 versus the time step.

initial start-up process it is a good global indicator of the overall accuracy of the method. The exact method shows second-order accuracy over the entire range of time steps. The classic method shows second-order convergence for the larger time steps (greater than 1) and then first-order convergence when the time step is very small (less than 1). Note that a time step of 1 corresponds to a maximum CFL of 0.05, which is smaller than one would normally use. In the practical operating range the classic method is second-order accurate, and the splitting error only begins to dominate in this case when the CFL number is very small. Therefore, in this particular flow, the primary advantage of the proposed method is not its temporal accuracy but its numerical efficiency. The classic method required very tight error tolerances on the conjugate gradient solver in order to obtain these results. Even moderate tolerances resulted in erratic convergence behavior of the classical method, because small errors in the Poisson solution for the pressure correction result in large errors in the velocity. This point is discussed further in the next section.

In order to demonstrate the ability of the method to predict flows in complex domains and with a variety of boundary conditions, the exact projection method was used to compute the flow of water around a square cylinder which is below a free surface. The mesh and initial computational domain are shown in Fig. 5. The flow is laminar and two dimensional but displays fairly complex behavior in time. An unsteady Von Karman vortex street develops behind the cylinder with a shedding frequency that depends on the Reynolds number. The free-surface response to the cylinder depends on the Froude number. The simulation uses an unstructured mesh, and the mesh moves with the free surface to retain an accurate representation of the free-surface motion. The interior mesh moves and occasionally reconnects using a flipping algorithm to maintain a high-quality mesh in the interior. The first calculation, shown in Fig. 6a, has a Reynolds number of 1000 based on the inlet velocity and

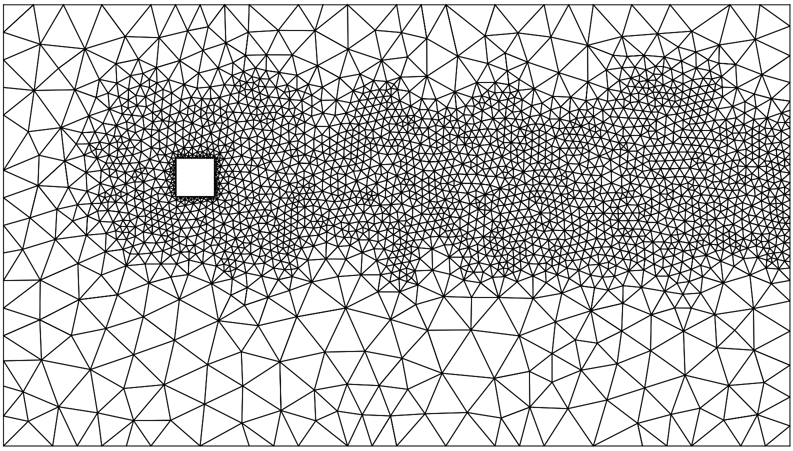

**FIG. 5.** Mesh and initial domain for the calculation of free-surface channel flow with a submerged square cylinder cross section.

cylinder height and a supercritical Froude number of 1.5 based on the inlet velocity and initial channel height. The second calculation, shown in Fig. 6b, uses the same Reynolds number but a subcritical Froude number of 0.75. Figures 6a and 6b show contours of the streamwise velocity at one instant in time.

It should be noted that the streamfunction on the cylinder is constant on the cylinder but varies in time and is therefore a more difficult boundary condition than the bottom of the domain, which is a slip wall (constant-in-time streamfunction). The free surface is not a streamline because it is not steady. The free surface uses a constant pressure boundary condition (zero gage pressure in this case), and the outflow uses a constant hydrostatic pressure assumption (pressure proportional to distance below the waterline). The inflow is uniform velocity. The unsteady nature of this particular flow is a good example of where one might be particularly concerned about time accuracy of the simulation, and where exact projection could be of significant value. The complexity of the algorithm and resulting code is equivalent to the classic projection method.

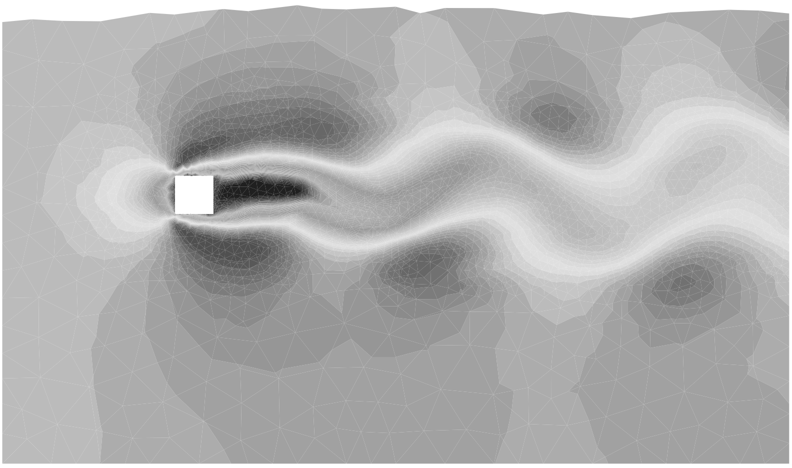

**FIG. 6a.** Prediction of the free-surface location and streamwise velocity field at Re = 1000, Fr = 1.5.

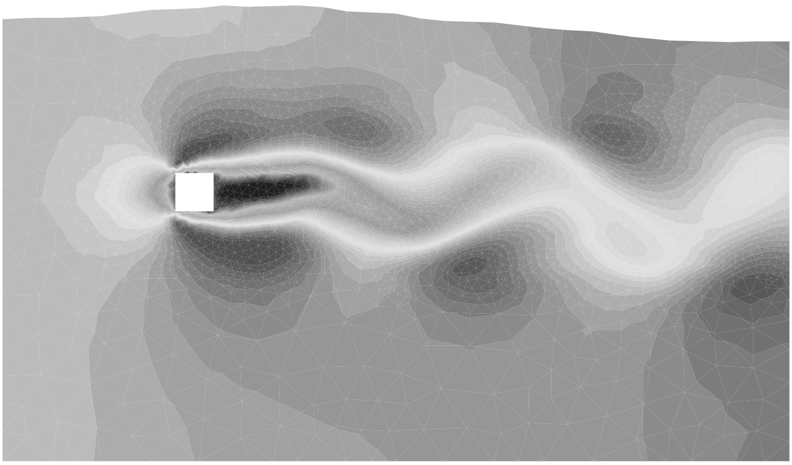

**FIG. 6b.** Prediction of the free-surface location and streamwise velocity at Re = 1000, Fr = 0.75.

## 6. ANALYSIS OF CONVERGENCE ERROR

The solution of the Poisson equation, in either the classic or the exact fractional step methods, is almost always computed using an iterative method. Whether the iterative method is a Krylov-based method or a multigrid scheme this invariably results in what is called convergence error. There is a direct trade-off between the convergence error that is tolerated and the speed of the Poisson solver (which dominates the overall solution time). It is clear from the analysis of Section 2 that convergence error in the classic fractional step method appears as nonzero dilatation (or local mass creation/destruction). On the other hand, the exact projection method maintains dilatation at machine precision at all times and convergence error appears as erroneous vorticity. It was suggested earlier that vorticity errors may be less detrimental to the overall flow solution than dilatation errors, thereby allowing fewer iterations for the exact projection and a faster solution than when using the classical projection method. In all likelihood, the exact impact of the error is problem dependent, but the test below suggests that errors in the mass cause larger errors in the velocity field than do errors in the vorticity. So exact projection often requires far fewer iterations to achieve the same accuracy in the velocity field.

To test this idea, one time step of the cavity flow problem was calculated using a fixed number of Jacobi preconditioned CG iterations. The solution was calculated using both the proposed exact fractional step method and the classic (second order) projection method of Dukowitcz and Dvinsky [8]. All other aspects of the code remained identical. The error after one time step was taken to be the $L_2$ (or RMS) error of the normal velocity components over all the cell faces. The exact solution for the error calculation was assumed to be the solution obtained after a very large number of iterations (128 iterations for the exact method and 512 iterations for the classic method). Because of the splitting error in the classic method, the "exact" solutions for the two methods are not identical. The error is defined this way on purpose. The affect of the splitting error has already been discussed. We wish to focus exclusively in this test on convergence error. This error norm gives a precise indication of the convergence error that results from solving an elliptic system with a finite number of iterations. While convergence error has not been discussed in prior publications concerning fractional step methods it is of very real practical importance, since the elliptic Poisson

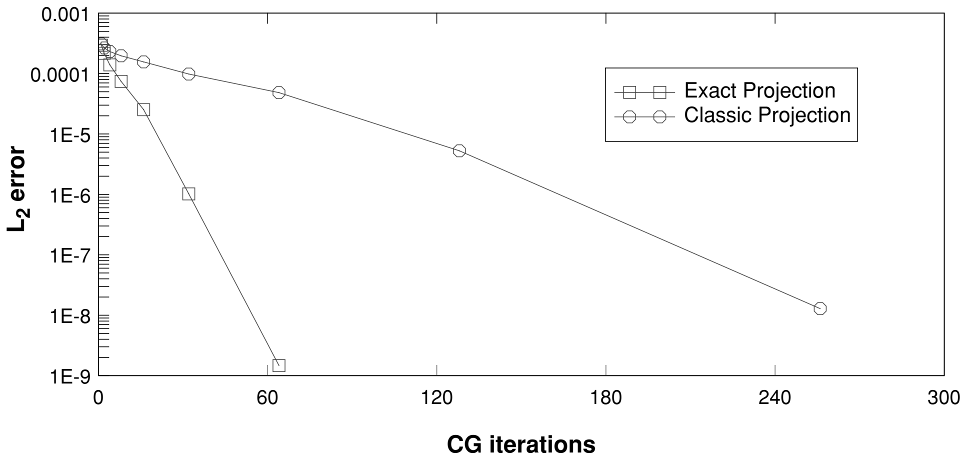

**FIG. 7.** Error in the velocity field for a fixed number of iterations of the CG solver and one time step.

equation and the tolerances that are imposed on its solution dominate the overall solution times of these methods.

A plot of the velocity error versus the number of iterations is shown in Fig. 7, for both the exact projection method and the classic projection method. The plot indicates that the classic projection method requires at least four times as many iterations to achieve the same level of accuracy in the velocity as the exact projection method. Taking into account the fact that the exact method requires more work per iteration (multiplying by the matrix **A** at each iteration) suggests that the computational advantages of the exact method will be somewhat less than this figure implies. If it is estimated that the computation of **A** is roughly equivalent to the computation of the Laplacian, then the exact projection method will be roughly twice as computationally efficient as the classic projection method for the same convergence error. Because the classic method solves for pressurelike variables at cells and the exact method solves for the streamfunction at nodes (2D) or edges (3D) this statement is really only correct for 2D Cartesian meshes and 3D unstructured meshes (see Table I). The computational advantages of the exact method are actually enhanced for 2D unstructured meshes and possibly almost completely removed for 3D Cartesian meshes.

## 7. DISCUSSION

An exact projection or fractional step method for solving the incompressible Navier–Stokes equations has been demonstrated which does not have any splitting error. The details of the method were discussed in the context of a staggered mesh discretization, but there appear to be no fundamental difficulties in applying this type of projection to other spatial discretization schemes. The greatest hurdle of generalizing this method to other discretization schemes is the construction of the divergence null space and its transpose. The null space and its transpose should preferably be sparse and explicit. It is unclear at this time whether other discretization schemes display simple null spaces like the staggered mesh does, but the authors would be surprised if a finite element scheme (possibly Petrov–Galerkin) could not be manipulated to do so.

The method was demonstrated on a two-dimensional cavity flow and a relatively complex unsteady flow in a multiconnected domain involving a moving free surface and a

Von Karman vortex street. The cavity flow showed good agreement with other numerical simulations and the free surface showed qualitatively correct behavior at both subcritical and supercritical Froude numbers. While the demonstrations in this work are for two-dimensional flows, the exact projection scheme is currently being used in research for three-dimensional unstructured staggered mesh simulations.

Analyses of the errors associated with using iterative solvers for the Poisson equation were performed. The results suggest that, at least for some flows, the convergence error in the exact method (extraneous vorticity) is more benign than that created by classic fractional step methods (extraneous dilatation). This allows fewer iterations of the solver and faster solution times for a given level of accuracy.

It is obvious that the exact projection method has some relationship to streamfunction methods. The essential distinction is that classic streamfunction methods manipulate the Navier–Stokes equations first and then discretize those equations. The exact projection method discretizes the Navier–Stokes equations first (and their boundary conditions) and only then manipulates the equations discretely into a form very similar to a streamfunction method (and using similar, but discrete, operators). This second approach eliminates two very important disadvantages of classic streamfunction methods [27]. First, in three dimensions the computational effort does not grow by a factor of three from the two-dimensional case. This is because exact projection does not solve for the whole streamfunction vector; it just solves for the component of streamfunction along each edge. In this way, the exact projection method applied to three-dimensional simulations is not fundamentally different from two-dimensional simulations. The other advantage is that boundary conditions are applied to the Navier–Stokes equations in primitive variables, so it is not necessary to specify the streamfunction or the vorticity on the boundaries. One can specify the streamfunction if one so desires, but it is also possible to specify the pressure or a normal velocity condition. This was demonstrated by the second test problem, where the streamfunction on the cylinder, the free surface, and the outflow are not specified.

Another possible advantage of the exact projection scheme over classic fractional step methods (but one that is not demonstrated herein) is that implicit convection may cause fewer difficulties. In classic fractional step methods, implicit convection often leads to large splitting errors (even if the order of the error is not low). Exact projection avoids splitting errors and therefore circumvents any problems associated with implicit convection.

While the purpose of the proposed exact method is equivalent to fractional step or projection methods it is clearly no longer well described by either moniker. If there is any projection going on, it is a projection of the basis functions into a discretely incompressible function space, rather than a projection of the solution. A more apt description of the method would be to refer to it as a discrete streamfunction method, or perhaps an incompressible basis function method.

## ACKNOWLEDGMENTS

This work was supported in part by the Office of Naval Research, the Air Force Office of Scientific Research, and a grant from Aquasions Inc. The second author (FG) acknowledges the support of the Office of Naval Research through element PE-0602435N.

## REFERENCES

1. R. Rannacher, *The Navier–Stokes Equations II—Theory and Numerical Methods*, Lecture Notes in Mathematics (Springer-Verlag, Berlin, 1992), Vol. 1530.

2. J. C. Strikwerda and Y. S. Lee, The accuracy of the fractional step method, *SIAM J. Numer. Anal.* **37**, 37 (1999).
3. D. Brown, R. Cortez, and M. Minion, Accurate projection methods for the incompressible Navier–Stokes equations, *J. Comput. Phys.* **168**, 464 (2001).
4. M. Lee, B. D. Oh, and Y. B. Kim, Canonical fractional-step methods and consistent boundary conditions for the incompressible Navier–Stokes equations, *J. Comput. Phys.* **168**, 73 (2001).
5. R. Temam, Remark on the pressure boundary condition for the projection method, *Theor. Comput. Fluid Dyn.* **3**, 181 (1991).
6. W. E and J. Liu, Projection method. I: Convergence and numerical boundary layers, *SIAM J. Numer. Anal.* **32**, 1017 (1995).
7. J. B. Perot, An analysis of the fractional step method, *J. Comput. Phys.* **108**, 51 (1993).
8. J. K. Dukowitcz and A. S. Dvinsky, Approximation factorization as a high order splitting for the implicit incompressible flow equations, *J. Comput. Phys.* **102**, 336 (1992).
9. J. B. Bell, P. Collela, and H. M. Glaz, A second order projection method for the incompressible Navier–Stokes equations, *J. Comput. Phys.* **85**, 257 (1989).
10. A. J. Chorin, Numerical solution of the Navier–Stokes equations, *Math. Comput.* **22**, 745, (1968).
11. P. M. Gresho and S. T. Chan, On the theory of semi-implicit projection methods for viscous incompressible flow and its implementation via a finite element method that also introduces a nearly consistent mass matrix. Part 2: Implementation, *Int. J. Numer. Methods Fluids* **11**, 621 (1990).
12. J. Kim and P. Moin, Application of the fractional-step method to the incompressible Navier–Stokes equations, *J. Comput. Phys.* **59**, 108 (1985).
13. J. Van Kan, A second order accurate pressure-correction scheme for viscous incompressible flow, *SIAM J. Sci. Comput.* **7**, 870 (1986).
14. J. B. Perot, Comments on the fractional step method, *J. Comput. Phys.* **121**, 190 (1995).
15. F. H. Harlow and J. E. Welch, Numerical calculations of time dependent viscous incompressible flow of fluid with a free surface, *Phys. Fluids* **8**(12), 2182 (1965).
16. D. K. Lilly, On the computational stability of numerical solutions of time-dependent non-linear geophysical fluid dynamics problems, *Mon. Weather Rev.* **93**, 11 (1965).
17. R. A. Nicolaides, The covolume approach to computing incompressible flows, in *Algorithmic Trends in Computational Fluid Dynamics*, edited by M. Y. Hussaini, A. Kumar, and M. D. Salas (Springer-Verlag, Berlin/New York, 1993), p. 295.
18. S. H. Chou, Analysis and convergence of a covolume method for the generalized Stokes problem, *Math. Comput.* **66** (217), 85 (1997).
19. J. B. Perot, Conservation properties of unstructured staggered mesh schemes, *J. Comput. Phys.* **159**, 58 (2000).
20. X. Zhang, D. Schmidt, and J. B. Perot, Accuracy and conservation properties of a three-dimensional staggered mesh scheme, *J. Comput. Phys.* **175**, 764–791 (2002).
21. R. A. Nicolaides, The covolume approach to computing incompressible flow, in *Incompressible Computational Fluid Dynamics*, edited by M. D. Gunzburger and R. A. Nicolaides (Cambridge Univ. Press, Cambridge, UK, 1993), p. 295.
22. V. Girault and P. Raviart, *Finite Element Approximation of the Navier–Stokes Equations* (Springer-Verlag, Berlin, 1986).
23. S. Turek, Tools for simulating non-stationary incompressible flow via discretely divergence-free finite element models, *Int. J. Numer. Methods Fluids* **18**, 71 (1994).
24. J. M. Hyman and M. Shashkov, The orthogonal decomposition theorems for mimetic finite difference methods, *SIAM J. Numer. Anal.* **36**(3), 788 (1999).
25. C. A. Hall, J. C. Cavendish, and W. H. Frey, The dual variable method for solving fluid flow difference equations on Delaunary Triangulations, *Comput. Fluids* **20**(2), 145 (1991).
26. U. Ghia, K. N. Ghia, and C. T. Shin, High Re solutions for incompressible flow using the Navier–Stokes equation and multigrid methods, *J. Comput. Phys.* **48**, 387 (1982).
27. R. Peyret and T. Taylor, *Computational Methods for Fluid Flow* (Springer-Verlag, Berlin, 1983).