

A FAST FINITE DIFFERENCE METHOD FOR SOLVING NAVIER-STOKES EQUATIONS ON IRREGULAR DOMAINS*

ZHILIN LI[†] AND CHENG WANG[‡]

Abstract. A fast finite difference method is proposed to solve the incompressible Navier-Stokes equations defined on a general domain. The method is based on the vorticity stream-function formulation and a fast Poisson solver defined on a general domain using the immersed interface method. The key to the new method is the fast Poisson solver for general domains and the interpolation scheme for the boundary condition of the stream function. Numerical examples that show second order accuracy of the computed solution are also provided.

Key words. Navier-Stokes equations, irregular domains, vorticity stream-function formulation, vorticity boundary condition, immersed interface method

AMS subject classifications. 65M06, 65M12, 76T05

1. Introduction

In this paper, we consider non-dimensional incompressible Navier-Stokes equations (NSE) in a general and bounded domain Ω :

$$\left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} \right) + \nabla p = \nu \Delta \mathbf{u}, \quad \mathbf{x} \in \Omega, \quad (1.1)$$

$$\nabla \cdot \mathbf{u} = 0, \quad (1.2)$$

$$\mathbf{u}|_{\partial\Omega} = \mathbf{0}, \quad \text{BC}, \quad (1.3)$$

$$\mathbf{u}(\mathbf{x}, 0) = \mathbf{u}_0, \quad \text{IC}, \quad (1.4)$$

where $\mathbf{u} = (u, v)$ is the velocity, p is the pressure, ν is the viscosity. We assume that the boundary of the domain, denoted as $\partial\Omega$ is piecewise smooth. We wish to solve the NSE numerically using a Cartesian grid that encloses the domain Ω .

Traditionally, finite element methods with a body-fitted grid are used to solve such problems defined on irregular geometries. The use of Cartesian grids for solving problems with complex geometry, moving interface, and free boundary problems has become quite popular recently, especially since popular Cartesian grid methods such as Peskin's immersed boundary (IB) method, see [24] for an overview, the level set method, see the original paper [23], the Clawpack [17], and others have been developed. One of advantages of Cartesian grid methods is that there is almost no cost in the grid generation. This is quite significant for moving interface and free boundary problems.

Depending on the magnitude of the Reynolds number $1/\nu$, numerical methods can be divided into two categories for two-dimensional problems: (1) numerical methods

*Received: April 15, 2002; Accepted (in revised version): July 19, 2002.

[†]Center for Research in Scientific Computation & Department of Mathematics, North Carolina State University, Raleigh, NC 27695-8205 (zhilin@math.ncsu.edu).

[‡]Department of Mathematics, Indiana University, Bloomington, IN 47405-5701 (cwang@indiana.edu).

based on the primitive variables formulation for problems with small to medium-sized Reynolds numbers, such as the projection method, see [3, 5, 26]. Usually an implicit or semi-implicit discretization in time is needed to take a reasonable time step; (2) the vorticity stream-function formulation for problems with large Reynolds numbers, see, for example, [10, 13] and the references therein.

There are a few articles in the literature that use projection-type methods based on Cartesian grids for interface problems or for problems with irregular geometries using an embedding technique. Among them are, just to name a few, Peskin's immersed boundary method with numerous applications [7, 8, 11, 24, 28]; the ghost fluid method [15]; the immersed interface method [16]; the finite volume method [2, 26, 29, 32], etc.

However, there has been little work about solving the full incompressible Navier-Stokes equations defined on complex domains using the *vorticity stream-function formulation* based on Cartesian grids except the very recent work by Calhoun [6]. In [6], a finite volume method coupled with the software package Clawpack [17] is used to solve the Navier-Stokes equations on irregular domains.

In this paper, we use a more direct finite difference approach based on a Cartesian grid and the vorticity stream-function formulation to solve the incompressible Navier-Stokes equations defined on an irregular domain. We believe that our method is simpler than the one developed in [6]. The computed vorticity and velocity from our method are nearly second order accurate, see Section 5.

The paper is organized as follows. In Section 2, we introduce the computational frame of the vorticity stream-function formulation and an outline of our algorithm. The fast Poisson solver for complex domains is explained in Section 3. In Section 4, we explain how to deal with the vorticity boundary condition. Numerical examples and conclusion are presented in Section 5 and Section 6 respectively.

2. The Computational Frame of the Vorticity Stream-Function Formulation for NSE on Irregular Domains

We use a rectangular box $R = [a, b] \times [c, d]$ to enclose the physical domain Ω . For simplicity of presentation, we use a uniform Cartesian grid

$$x_i = a + ih, \quad i = 0, 1, \dots, M, \quad y_j = c + jh, \quad j = 0, 1, \dots, N, \quad (2.1)$$

where $h = (b - a)/M = (d - c)/N$. Usually we choose R in such a way that the distance between the boundaries $\partial\Omega$ and ∂R is at least $2h$ apart.

The boundary of Ω is expressed as the zero-level set of a two-dimensional function $\varphi(x, y)$:

$$\varphi(x, y) \begin{cases} < 0, & \text{if } (x, y) \text{ is in the inside of } \Omega, \\ = 0, & \text{if } (x, y) \text{ is on the boundary of } \Omega, \\ > 0, & \text{if } (x, y) \text{ is in the outside of } \Omega. \end{cases} \quad (2.2)$$

Since $\partial\Omega$ is piecewise smooth, the level set function φ should be chosen to be at least Lipschitz continuous in the neighborhood of $\partial\Omega$. Usually φ is chosen as the signed distance function. Note that the level set function can be easily defined at grid points.

Let

$$\begin{aligned}\varphi_{i,j}^{max} &= \max\{\varphi_{i-1,j}, \varphi_{i,j}, \varphi_{i+1,j}, \varphi_{i,j-1}, \varphi_{i,j+1}\}, \\ \varphi_{i,j}^{min} &= \min\{\varphi_{i-1,j}, \varphi_{i,j}, \varphi_{i+1,j}, \varphi_{i,j-1}, \varphi_{i,j+1}\}.\end{aligned}\quad (2.3)$$

We define (x_i, y_j) as an *irregular grid point* if $\varphi_{i,j}^{max} \varphi_{i,j}^{min} \leq 0$. Otherwise the grid point is a *regular grid point*.

The two-dimensional Navier-Stokes equation in vorticity stream-function formulation is

$$\partial_t \omega + u\omega_x + v\omega_y = \nu \Delta \omega, \quad (2.4)$$

$$\Delta \psi = \omega, \quad \psi|_{\partial\Omega} = 0, \quad (2.5)$$

$$u = -\psi_y, \quad v = \psi_x, \quad (2.6)$$

where $\omega = \nabla \times \mathbf{u} = -u_y + v_x$ is the vorticity, ψ is the stream-function which satisfies the nonslip boundary condition

$$\psi_n|_{\partial\Omega} = \frac{\partial\psi}{\partial\mathbf{n}}\Big|_{\partial\Omega} = 0, \quad (2.7)$$

in addition to the no-penetration boundary condition $\psi|_{\partial\Omega} = 0$, where \mathbf{n} is the unit normal vector of the boundary $\partial\Omega$ pointing outward.

We define the standard central finite difference operators applied to the grid function u_{ij} below

$$D_x u_{i,j} = \frac{u_{i+1,j} - u_{i-1,j}}{2h}, \quad D_y u_{i,j} = \frac{u_{i,j+1} - u_{i,j-1}}{2h}, \quad (2.8)$$

$$\Delta_h u_{i,j} = (D_x^2 + D_y^2) u_{i,j} = \frac{u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}}{h^2}. \quad (2.9)$$

With these notations, the semidiscretized Navier-Stokes equations in vorticity stream-function formulation is

$$\partial_t \omega + u D_x \omega + v D_y \omega = \nu \Delta_h \omega. \quad (2.10)$$

2.1. An Outline of the Vorticity Stream-Function Formulation for NSE Defined on Irregular Domains. We use the classical fourth-order Runge-Kutta method, which is a multistage explicit time-stepping procedure to treat the semi-discretized equations (2.10). The explicit treatment of the convection and diffusion terms appearing in the momentum equations makes the whole scheme very easy to implement. Such explicit treatment can avoid any stability concern caused by the cell-Reynolds number constraint if a high order Runge-Kutta method, such as the classical RK4, is applied. This observation was first made by E and Liu in [10]. As a result, only one Poisson solver, which will be explained in detail in the next section, is required at each Runge-Kutta time stage. That makes the method extremely efficient. We refer the readers to [10, 30] for the discussion of the method of the vorticity stream-function formulation for rectangular domains.

We present the following outline for the explicit Euler method from time level t^k to t^{k+1} to demonstrate the essence of our algorithm. The extension to Runge-Kutta method is straightforward.

1. Solve the Poisson equation

$$\Delta\psi^{k+1} = \omega^k, \quad \psi^{k+1}|_{\partial\Omega} = 0,$$

to get the stream-function ψ^{k+1} .

2. Update the velocity using

$$u^{k+1} = -D_y\psi^{k+1}, \quad v^{k+1} = D_x\psi^{k+1}.$$

3. Update the vorticity from

$$\frac{\omega^{k+1} - \omega^k}{\Delta t} = \nu\Delta_h\omega^k - u^{k+1}D_x\omega^k - v^{k+1}D_y\omega^k, \quad \omega^{k+1}|_{\partial\Omega} = \Delta_h\psi^{k+1}.$$

In our implementation, we actually use a fourth-order Runge-Kutta method to update the vorticity from t^k to t^{k+1} so that global second-order accuracy can be obtained.

There are two crucial components in our algorithm described above. The first one is how to solve the Poisson equation on an irregular domain, which will be explained in the next section. The second one is the treatment of the boundary condition since there are two boundary conditions for ψ . The Dirichlet boundary condition $\psi = 0$ on $\partial\Omega$ is used to solve for the stream-function via the vorticity computed from (2.10). Yet the normal boundary condition, $\frac{\partial\psi}{\partial\mathbf{n}} = 0$ in (2.7), cannot be enforced directly. The way to overcome this difficulty is to convert it into the boundary condition for the vorticity. The detailed process of vorticity boundary condition will be given in Section 4.

3. The Fast Poisson Solver on Irregular Domains

Using the vorticity stream-function formulation to solve the Navier-Stokes equations, we need to solve the Poisson equation $\Delta\psi = \omega$ on the irregular domain Ω . In this section, we outline our fast Poisson solver for irregular domains. We refer the readers to the references [12, 18, 20] for the theoretical analysis and the details of the implementation. The two-dimensional fast Poisson solver is also available to the public [19].

Our Poisson solver on irregular domains is based on the fast immersed interface method (IIM) for interface problems. The main idea is to extend the Poisson equation from Ω to a rectangular domain R . This procedure allows us to use fast Poisson solvers such as FFT on a fixed Cartesian grid independent of the shape of the irregular domain.

We extend the source term of the Poisson equation by zero outside Ω but inside R . We require the normal derivative of the solution ψ to be continuous across the immersed boundary $\partial\Omega$. The solution itself is allowed to have a finite jump g . In the

language of potential theory this requirement is equivalent to introducing a double-layer source on $\partial\Omega$. This extension leads to the following interface problem:

$$\begin{aligned} \Delta\psi &= \begin{cases} \omega(x, y), & \text{if } (x, y) \in \Omega, \\ 0, & \text{if } (x, y) \in R - \Omega, \end{cases} \\ \left[\frac{\partial\psi}{\partial\mathbf{n}}\right] &= 0, \quad [\psi] = g, \quad \text{on } \partial\Omega, \\ \psi &= 0, \quad \text{on } \partial R, \end{aligned} \quad (3.1)$$

where $[\cdot]$ denotes the jump across $\partial\Omega$. We need to determine the particular g , which is defined along the boundary $\partial\Omega$, so that the solution ψ of (3.1) satisfies the Dirichlet boundary condition

$$\psi^- = 0, \quad \text{on } \partial\Omega, \quad (3.2)$$

where ψ^- is the limiting value of the solution on the boundary from within the domain Ω . Note that the solution of the interface problem above is a functional of g . There is a unique solution $\psi(g)$ which is in piecewise $H^2(R)$ space if ω and g are in L^2 space, and the interface $\partial\Omega$ is Lipschitz continuous. We refer the readers to [4] for the discussion of the regularity of the interface problem.

To numerically compute the solution of (3.1)–(3.2) for ψ and g , we use the standard central finite difference

$$\Delta_h\psi_{ij} = \omega_{ij} \quad (3.3)$$

at regular grid points, inside and outside of R . At irregular grid points, the finite difference scheme is

$$\Delta_h\psi_{ij} = \omega_{ij} + C_{ij}, \quad (3.4)$$

where C_{ij} is a correction term that depends on the jump g . Therefore, the solution to (3.1) satisfies a linear system of equations of the form

$$A\Psi + BG = F_1, \quad (3.5)$$

where A is the matrix obtained from the discrete Laplacian, Ψ is the approximate solution ψ at grid points, and G is the discrete value of g defined on a set of points on the boundary $\partial\Omega$, BG is the correction terms at irregular grid points, and F_1 is the source term inside Ω and zero outside Ω . The dimension of G is much smaller than the dimension of Ψ .

We use the least squares interpolation scheme [18] to discretize the boundary condition (3.2) for a given Ψ . At a given point $\mathbf{x}^* = (x^*, y^*) \in \partial\Omega$ where G is defined, the interpolation scheme has the form

$$\sum_{|\mathbf{x}_{ij} - \mathbf{x}^*| \leq \epsilon} \gamma_{ij}\psi_{ij} = 0, \quad (3.6)$$

where $\mathbf{x}_{i,j} = (x_i, y_j)$, ϵ is taken between $2h \sim 3h$ meaning that we use the information of the grid function ψ_{ij} in the neighborhood of \mathbf{x}^* . We refer the readers to [18] about how to find the coefficients γ_{ij} in the interpolation scheme. The interpolation scheme in the matrix-vector form can be written as

$$C\Psi + DG = F_2. \tag{3.7}$$

Thus we obtain the following system of equations for the solution Ψ and the intermediate discrete jump G on the boundary

$$\begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} \Psi \\ G \end{bmatrix} = \begin{bmatrix} F_1 \\ F_2 \end{bmatrix}. \tag{3.8}$$

The Schur complement of (3.8) for G is

$$(D - CA^{-1}B)G = Q, \tag{3.9}$$

where

$$Q = F_2 - CA^{-1}F_1.$$

The dimension of equation (3.9) for G is a much smaller system than that of equation (3.8) for Ψ . We use the generalized minimum residue (GMRES) method to solve the Schur complement system. Each iteration of the GMRES method involves one matrix-vector multiplication, which is $(D - CA^{-1}B)G$ with a specified G . This involves a fast Poisson solver for computing $A^{-1}BG$ on the rectangular domain R , and the interpolation scheme to compute the residual of (3.9). The dominant cost in each iteration is the fast Poisson solver from the Fishpack [1].

This Poisson solver that we outlined above for irregular domains is second-order accurate. The number of calls to the fast Poisson solver on the rectangular domain is the same as the number of GMRES iterations, and it is almost independent of the mesh size but depends only on the geometry of the domain.

There are other elliptic solvers for irregular domains using embedding or fictitious domain techniques. The earlier ones include the capacitance matrix method [25], the integral equation approach [21, 22], and the recent finite volume method [14]. The similarity of all embedding methods is that the standard finite difference/volume schemes are used away from the interface. The difference is the way to deal with the boundary condition. The capacitance matrix method works through manipulating the structure of the coefficient matrix. In the integral equation approach, the integral equation is discretized and the dense system is solved before a fast Poisson solver is called, see [21]. Note that the integral equation is a second kind, and therefore it is well conditioned. We believe that the Schur complement (3.9) is equivalent to the discrete integral equation. In our approach, we do not solve the dense system using a direct method, but use the GMRES iterative method instead since the Schur complement is well conditioned, and the condition number is almost a constant independent of the size of system. This has also been confirmed in [31]. The number of iterations of the GMRES iteration is typically between $5 \sim 20$ independent of the mesh size; see

Table 3.1 for an example. The number of iterations depends only on the geometry of $\partial\Omega$. Theoretically, the GMRES method converges to the solution, or the least squares solution if the system is singular, at most N_1 steps, where N_1 is the number of unknowns, see [27].

In our implementation, the matrices and vectors are never explicitly formed. The fast solvers with examples for Poisson/Helmholtz equations on irregular domains are available to the public through anonymous ftp at *ftp.ncsu.edu* under the directory */pub/math/zhilin/Packages*.

We illustrate the efficiency of the fast Poisson solver on irregular domains through a numerical example. We solve the Poisson equation

$$\begin{aligned} -\Delta\psi &= 4, & \text{in } \Omega, \\ \psi &= x^2 + y^2 + e^x \cos y, & \text{on } \partial\Omega, \end{aligned} \quad (3.10)$$

on an ellipse

$$\Omega = \{(x, y) : x^2/a^2 + y^2/b^2 < 1\}. \quad (3.11)$$

The exact solution is

$$\psi(x, y) = x^2 + y^2 + e^x \cos y.$$

In Table 3.1, we show the error $E(N)$ of the numerical solution in the maximum norm on an $N \times N$ grid (*i.e.* $M = N$) for various values of N . In the table, N_1 is the number of unknowns of the Schur complement. We also show the number of GMRES iterations k . The order of convergence is measured by

$$O = \frac{\log [E(N)/E(2N)]}{\log 2}. \quad (3.12)$$

The order in Table 3.1 approaches number 2 as $N \rightarrow \infty$, which indicates second-order convergence. The number of GMRES iterations decreases slightly as N increases. The stopping criteria for GMRES iteration is $tol = 10^{-8}$. From Table 3.1, we can clearly see second-order convergence and the number of iterations is almost a constant as we refine the mesh.

4. The Vorticity Boundary Condition

To solve the momentum equation (2.10), we need a boundary condition for the vorticity. Physically, the vorticity boundary condition enforces the no-slip boundary condition for the velocity. The no-penetration boundary condition, which can be converted into the Dirichlet boundary condition for ψ : $\psi|_{\partial\Omega} = 0$, is used in the Poisson equation (2.5) for ψ . Therefore we only need the information of the vorticity at interior grid points to solve the stream-function ψ with the homogeneous boundary condition. On the other hand, the no-slip boundary condition $\frac{\partial\psi}{\partial\mathbf{n}}|_{\partial\Omega} = 0$, along with the one-sided approximation of $\omega = \Delta\psi$, is converted into a local vorticity boundary condition.

TABLE 3.1. Results of a grid refinement study for the numerical solution of (3.10) on the domain (3.11) with $a = 0.5$ and $b = 0.15$. Here, N is the number of gridlines in the x and y directions, E is the error of the numerical solution in the maximum norm, O is the order of convergence as we refine the grid, N_1 is the number of unknowns of the Schur complement, and k is the number of GMRES iterations. We see second-order convergence, and the number of iterations is almost a constant as we refine the mesh.

N	E	O	N_1	k
32	$4.21465 \cdot 10^{-3}$		36	9
64	$8.1371 \cdot 10^{-4}$	2.3728	68	7
128	$1.7614 \cdot 10^{-4}$	2.2078	136	6
256	$3.8196 \cdot 10^{-5}$	2.2053	268	6
512	$8.5548 \cdot 10^{-6}$	2.1586	538	5

4.1. A Review of Local Vorticity Boundary Formula in a Rectangular Domain.

Let us assume that Ω is a rectangle for the moment. The Dirichlet boundary condition $\psi = 0$ on $\partial\Omega$ is used to solve the stream-function via the vorticity obtained from (2.10). Yet the normal boundary condition, $\frac{\partial\psi}{\partial\mathbf{n}} = 0$, cannot be enforced directly. The way to overcome this difficulty is to convert it into the boundary condition for the vorticity. We use $\psi|_{\partial\Omega} = 0$ and $\frac{\partial\psi}{\partial\mathbf{n}} = 0$ to approximate the vorticity on the boundary. Take the bottom part of $\partial\Omega$, where the subscript j is zero, for example, we use the central finite difference scheme to approximate the Laplacian, which is simply $D_y^2\psi$ since $D_x^2\psi = 0$ from the boundary condition $\psi = 0$. The second-order finite difference approximation yields

$$\omega_{i,0} = D_y^2\psi_{i,0} = \frac{\psi_{i,1} + \psi_{i,-1} - 2\psi_{i,0}}{h^2} = \frac{2\psi_{i,1}}{h^2} - \frac{2}{h} \frac{\psi_{i,1} - \psi_{i,-1}}{2h}, \tag{4.1}$$

where $\psi_{i,0} = 0$ and $(i, -1)$ refers to a “ghost” grid point outside of the computational domain. Since

$$\frac{\psi_{i,1} - \psi_{i,-1}}{2h} \approx \frac{\partial\psi}{\partial\mathbf{n}} + O(h^2) = O(h^2),$$

we have $\psi_{i,-1} = \psi_{i,1} + O(h^3)$ which leads to **Thom’s formula**

$$\omega_{i,0} = \frac{2\psi_{i,1}}{h^2}. \tag{4.2}$$

We should mention here that by formal Taylor expansion, one can prove that Thom’s formula is only first-order approximation to the boundary condition for ω . However, more sophisticated consistency analysis shows that the scheme actually is indeed second-order accurate. The result was first proved in [13].

The vorticity on the boundary can also be determined by other approximations to $\psi_{i,-1}$. For example, using a third-order one-sided finite difference scheme to approximate the normal boundary condition $\frac{\partial\psi}{\partial\mathbf{n}} = 0$, we can write

$$\begin{aligned} (\partial_y\psi)_{i,0} &= \frac{-\psi_{i,-1} + 3\psi_{i,1} - \frac{1}{2}\psi_{i,2}}{3h} = 0 + O(h^3), \quad \text{which leads to} \\ \psi_{i,-1} &= 3\psi_{i,1} - \frac{1}{2}\psi_{i,2} + O(h^4). \end{aligned} \tag{4.3}$$

Plugging this back to the difference vorticity formula $\omega_{i,0} = \frac{1}{h^2}(\psi_{i,1} + \psi_{i,-1})$ in (4.1), we have **Wilkes-Pearson's formula**

$$\omega_{i,0} = \frac{1}{h^2}(4\psi_{i,1} - \frac{1}{2}\psi_{i,2}). \quad (4.4)$$

This formula is second-order accurate for the vorticity on the boundary.

From the descriptions above, we see that the vorticity boundary condition can be derived from the combination of the no-slip boundary condition $\frac{\partial\psi}{\partial\mathbf{n}}|_{\partial\Omega} = 0$ and some one-sided approximations to $\omega = \Delta\psi$.

4.2. The Extension to a Curved Domain. The extension of the above methodology to a domain with a curved boundary is similar but a little more complicated.

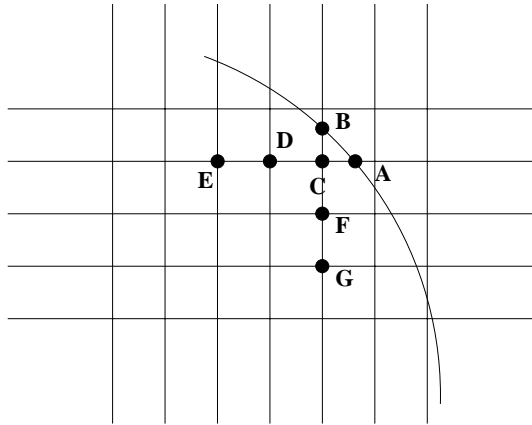


FIG. 4.1. A diagram of the geometry near the boundary.

We use FIG. 4.1 as an illustration. FIG. 4.1 shows several grid points near the boundary $\partial\Omega$. In FIG. 4.1, D, E, F, G are regular grid points, AB is an arc section on the boundary. Special attention is needed at the grid points close to the boundary, such as the point C . We denote $a = |AC|/h$, $b = |BC|/h$. Note that $0 < a, b < 1$.

We explain how to determine the vorticity value on the boundary points like A and B from the combined information of the no-slip boundary condition $\frac{\partial\psi}{\partial\mathbf{n}}|_{\partial\Omega} = 0$ and the one-sided approximation of $\omega = \Delta\psi$. We will derive a simple local formula analogous to (4.2) or (4.4) to obtain a vorticity boundary value at A , which is an intersection of the gridline and the boundary $\partial\Omega$. The discussion for the point B is similar. We need the vorticity value at those points when we solve the vorticity via (2.10) and the stream function ψ in (2.5).

The combination of the Dirichlet boundary condition $\psi|_{\partial\Omega} = 0$ and the Neumann boundary condition $\frac{\partial\psi}{\partial\mathbf{n}} = 0$ implies that

$$\nabla\psi = 0, \quad \text{at } A. \quad (4.5)$$

In other words, the partial derivative of the stream function along any direction is zero on the boundary. This can also be seen by the fact that both $u = -\partial_y\psi$ and $v = \partial_x\psi$ vanish on the boundary.

The local Taylor expansion at the boundary point A gives

$$\psi_C = \frac{a^2h^2}{2}(\partial_x^2\psi_A) - \frac{a^3h^3}{6}(\partial_x^3\psi_A) + O(h^4), \tag{4.6}$$

$$\psi_D = \frac{(1+a)^2h^2}{2}(\partial_x^2\psi_A) - \frac{(1+a)^3h^3}{6}(\partial_x^3\psi_A) + O(h^4), \tag{4.7}$$

where the information of $\psi = 0$ and $\partial_x\psi = 0$ at point A was used in the derivation for (4.6) and (4.7).

The Taylor expansion (4.6) gives a first order approximation to $\partial_x^2\psi$ at the boundary point A

$$(\partial_x^2\psi)_A = \frac{2}{a^2h^2}\psi_C + O(h), \tag{4.8}$$

which corresponds to the Thom's formula (4.2). Or the combination of (4.6) and (4.7) gives a second-order approximation to $\partial_x^2\psi$ at point A

$$(\partial_x^2\psi)_A = \frac{2}{a^2h^2}\left((1+a)\psi_C - \frac{a^3}{(1+a)^2}\psi_D\right) + O(h^2), \tag{4.9}$$

which corresponds to the Wilkes' formula (4.4).

REMARK 4.1. In the case of $a = 1$, i.e., the boundary point A happens to be on regular grid, the formula (4.8) becomes

$$(\partial_x^2\psi)_A = \frac{2}{h^2}\psi_C, \tag{4.10}$$

and (4.9) becomes

$$(\partial_x^2\psi)_A = \frac{4\psi_C - \frac{1}{2}\psi_D}{h^2}, \tag{4.11}$$

which are exactly the same as Thom's formula and Wilkes' formula for rectangular domain, as in (4.2), (4.4), respectively.

However, in a domain with a curved boundary, the value of $\partial_x^2\psi$ on the boundary point like A is not enough to determine $\omega = \Delta\psi = (\partial_x^2 + \partial_y^2)\psi$, due to the fact that $\partial_y^2\psi$ cannot be determined in the same way as we did for ψ_A and ψ_B . To deal with this difficulty, we need to use the information of the stream function around the boundary point A and the relation between $\Delta\psi$ and $\partial_x^2\psi$.

Let θ be the angle between the tangential direction of the boundary $\partial\Omega$ passing through A and the horizontal line, i.e., $\tan\theta$ is the slope of the tangent line for the boundary curve at A . Some simple manipulations indicate

$$\partial_x\psi = \cos\theta\partial_\tau\psi + \sin\theta\partial_{\mathbf{n}}\psi, \quad \partial_y\psi = -\sin\theta\partial_\tau\psi + \cos\theta\partial_{\mathbf{n}}\psi, \tag{4.12}$$

where τ is the unit vector along the tangential direction. It is easy to show that

$$\partial_x^2 + \partial_y^2 = \partial_\tau^2 + \partial_{\mathbf{n}}^2, \quad (4.13)$$

from (4.12). In other words, the Laplacian operator is invariant with respect to orthogonal coordinate systems.

We also can verify that

$$\partial_x^2 \psi = \cos^2 \theta \partial_\tau^2 \psi + \sin^2 \theta \partial_{\mathbf{n}}^2 \psi + 2 \cos \theta \sin \theta (\partial_\tau \partial_{\mathbf{n}} \psi) \quad (4.14)$$

from (4.12). Meanwhile, we have $\partial_\tau \partial_{\mathbf{n}} \psi = 0$, due to the fact that $\partial_{\mathbf{n}} \psi$ is identically zero on the boundary because of the no-slip boundary condition. Thus we arrive at

$$\partial_x^2 \psi = \cos^2 \theta \partial_\tau^2 \psi + \sin^2 \theta \partial_{\mathbf{n}}^2 \psi. \quad (4.15)$$

It is obvious that

$$\partial_\tau^2 \psi = 0, \quad \text{on } \partial\Omega, \quad (4.16)$$

because of no penetration boundary condition $\psi|_{\partial\Omega} = 0$. Therefore, we get the following identity on the boundary:

$$\partial_x^2 \psi = \sin^2 \theta \partial_{\mathbf{n}}^2 \psi, \quad (4.17)$$

which results in

$$\partial_{\mathbf{n}}^2 \psi = \csc^2 \theta \partial_x^2 \psi, \quad \text{at } A. \quad (4.18)$$

The combination of (4.13) and (4.18), along with the argument in (4.16), gives

$$\omega_A = (\partial_x^2 + \partial_y^2) \psi = \partial_{\mathbf{n}}^2 \psi = \csc^2 \theta \partial_x^2 \psi, \quad \text{at } A. \quad (4.19)$$

Thus we obtain the relationship between ω and $\partial_x^2 \psi$ on the boundary.

Plugging (4.8), which is a first-order approximation of $\partial_x^2 \psi$ at A , into the identity (4.19), we get

$$\omega_A = \csc^2 \theta \cdot \frac{2}{a^2 h^2} \psi_C, \quad (4.20)$$

which can be regarded as Thom's formula for a curved boundary. We can get an analogue of Wilkes' formula below for a curved boundary by plugging (4.9), which is a second-order approximation to $\partial_x^2 \psi$ at A , into the identity (4.19):

$$\omega_A = \csc^2 \theta \cdot \frac{2(1+a)\psi_C - \frac{a^3}{(1+a)^2} \psi_D}{a^2 h^2}. \quad (4.21)$$

The derivation of the vorticity value at the projected boundary point B is similar to that of A . As shown in FIG. 4.1, we assume that $CF = FG = \Delta y = h$, and $b = |BC|/h$, (note that $0 < b < 1$). By repeating a similar arguments as in (4.5)–(4.19), the vorticity value at point B can be approximated as

$$\omega_B = \sec^2 \theta \partial_y^2 \psi. \quad (4.22)$$

The corresponding Thom's formula for ω_B turns out to be

$$\omega_B = \sec^2 \theta \cdot \frac{2}{b^2 h^2} \psi_C, \tag{4.23}$$

and the corresponding Wilkes' formula for ω_B turns out to be

$$\omega_B = \sec^2 \theta \cdot \frac{2(1+b)\psi_C - b^3/(1+b)^2\psi_F}{b^2 h^2}. \tag{4.24}$$

5. Numerical Examples

In this section, we present two numerical experiments for solving the Navier-Stokes equations using our algorithm. The first one is a constructed example in which we know the exact solution. The grid refinement analysis is given. The second test is a physical relevant example, the vortex flow inside a disk.

5.1. A Grid Refinement Analysis. In this example, the computational domain Ω is an elliptic disk whose boundary ($4x^2 + 16y^2 = 1$) is the zero level set of the following function $\varphi(x, y)$:

$$\varphi(x, y) = \sqrt{4x^2 + 16y^2} - 1, \tag{5.1}$$

inside the square $[-1, 1] \times [-1, 1]$. The exact mean stream function is chosen to be

$$\psi_e(x, y, t) = \begin{cases} \left(r(x, y) - 0.25\right)^2, & \text{if } \varphi(x, y) \leq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{5.2}$$

where $r(x, y) = x^2 + 4y^2$. It is straightforward to verify that the exact stream function is smooth inside the elliptic domain Ω and

$$\psi = 0, \quad \frac{\partial \psi}{\partial \mathbf{n}} = 0, \quad \text{on } \partial\Omega, \tag{5.3}$$

so that the no-penetration, no-slip boundary condition is satisfied for the velocity field. The corresponding exact velocity and vorticity functions turn out to be

$$\begin{aligned} u_e(x, y, t) &= -\partial_y \psi_e(x, y, t) = -16y \left(r(x, y) - 0.25\right) \text{cost}, \\ v_e(x, y, t) &= \partial_x \psi_e(x, y, t) = 4x \left(r(x, y) - 0.25\right) \text{cost}, \\ \omega_e(x, y, t) &= \Delta \psi_e(x, y, t) = \left[20 \left(r(x, y) - 0.25\right) + 8(x^2 + 16y^2)\right] \text{cost}, \end{aligned} \tag{5.4}$$

inside the elliptic region Ω and zero outside.

The force term \mathbf{f} can be calculated exactly from the vorticity transport equation

$$\begin{cases} \partial_t \omega_e + (\mathbf{v}_e \cdot \nabla) \omega_e = \nu \Delta \omega_e + \mathbf{f}, \\ \Delta \psi_e = \omega_e, \\ \psi_e = 0, \quad \frac{\partial \psi_e}{\partial \mathbf{n}} = 0, \quad \text{on } \partial\Omega, \\ \mathbf{v}_e = \nabla^\perp \psi_e = (-\partial_y \psi_e, \partial_x \psi_e). \end{cases} \tag{5.5}$$

TABLE 5.1. *The errors of the computed velocity and stream function in different norms, and the order of accuracy, at $t = 0.5$. The viscosity is $\nu = 0.001$, and the CFL condition $\frac{\Delta t}{h} = 0.5$. The Wilkes' formulae for a curved boundary is used.*

	N	L^1 error	L^1 order	L^2 error	L^2 order	L^∞ error	L^∞ order
\mathbf{u}	32	5.07e-05		1.01e-04		3.40e-04	
	64	1.49e-05	1.77	2.84e-05	1.83	1.06e-04	1.68
	128	4.00e-06	1.90	7.55e-06	1.91	3.18e-05	1.74
	256	1.05e-06	1.93	2.05e-06	1.88	1.59e-05	1.00
ψ	32	5.66e-06		1.09e-05		3.20e-05	
	64	1.51e-06	1.91	2.90e-06	1.91	8.30e-06	1.95
	128	4.08e-07	1.89	7.71e-07	1.91	2.17e-06	1.95
	256	9.56e-08	2.09	1.81e-07	2.09	5.16e-07	2.07

The second-order accurate numerical method proposed in this paper is used to solve the above system (5.5). The viscosity is chosen to be $\nu = 0.001$ so the Reynolds number is $Re = 1000$. The final time is taken to be $t = 0.5$.

Table 5.1 shows the grid refinement analysis for the velocity \mathbf{u} and stream-function ψ using the Wilkes' formulae. Note that the errors are quite small. The error obtained from a very coarse grid 32 by 32 is $10^{-4} \sim 10^{-5}$ already. Second-order convergence can be clearly seen for L^1 , L^2 , and L^∞ norms. Therefore, the whole scheme leads to very good approximations to both the velocity and the stream function that we are concerned.

5.2. The Simulation of the Vortex Flow Inside a Disk. In this case, we choose the domain as a disk $x^2 + y^2 \leq \frac{1}{4}$. The square used to enclose the disk is $R = [-\frac{3}{4}, \frac{3}{4}] \times [-\frac{3}{4}, \frac{3}{4}]$. The initial vortex is set to be confined into an ellipse region $x^2 + 4y^2 \leq \frac{1}{4}$.

The initial stream-function profile inside the ellipse is given by

$$\psi(x, y, 0) = 2 \left(x^2 + 4y^2 - \frac{1}{4} \right)^2 \quad (5.6)$$

which vanishes outside the ellipse, see FIG. 5.1.

It is straightforward to verify that the initial stream function satisfies the no-penetration, and noslip boundary conditions. Accordingly, the initial data for the vorticity field is chosen as

$$\omega(x, y, 0) = \Delta\psi(x, y, 0) = \begin{cases} 40(x^2 + 4y^2 - \frac{1}{4}) + 16(x^2 + 16y^2), & \text{if } x^2 + 4y^2 \leq \frac{1}{4}, \\ 0, & \text{otherwise.} \end{cases} \quad (5.7)$$

The contour plots for the initial stream function and vorticity are shown in FIG. 5.1.

In our computation, Wilkes' formulae (4.21) and (4.23) were used as the boundary condition for the vorticity. The Reynolds number was chosen to be $Re = 10^4$, which is quite high over an irregular domain. We use a 384×384 grid to carry out the simulation.

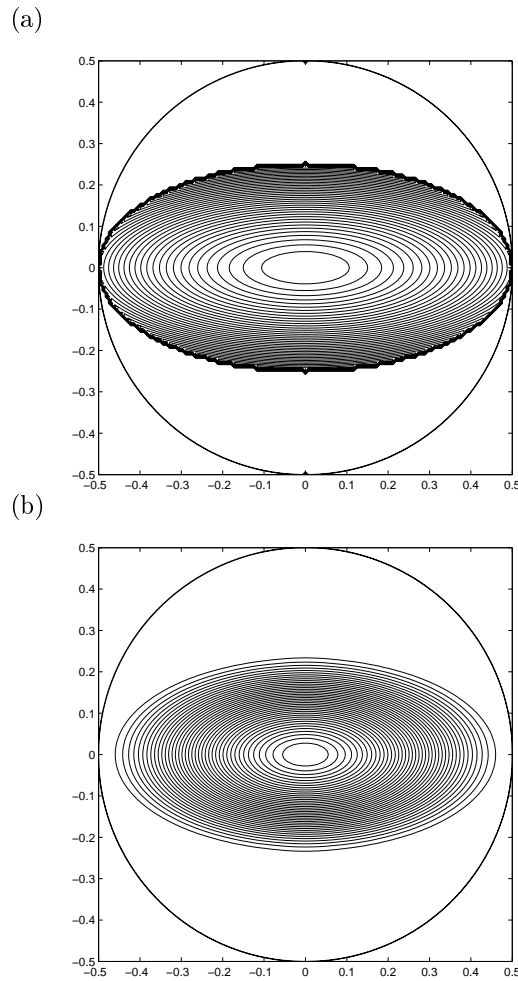


FIG. 5.1. *The contour plot of the initial vorticity and stream function.*

In FIG. 5.2 we show the computed stream function at different time: $t_1 = 0.5$, $t_2 = 1$, $t_3 = 1.5$, $t_4 = 2$, respectively. The vortex flow inside the ellipse $x^2 + 4y^2 \leq \frac{1}{4}$ interacts with the flow in the disk but outside the ellipse. The vortex structure spreads over the whole domain. Meanwhile, the vortex dynamics drive the flow to rotate clockwise. The rolling up structure due to the high Reynolds number can also be observed. The flow keeps symmetry, with generation and separation of bubbles as time goes on. The detailed evolution of the flow structure is clearly presented in the figure.

In FIG. 5.3, we show the contour plot of the x component of the velocity at the same time sequences $t = 0.5, 1, 1.5, 2$. Similar flow behavior can be observed. The y component of the velocity has similar behavior and is not shown here.

6. Conclusions

In this paper, a new second order finite difference method based on the vorticity

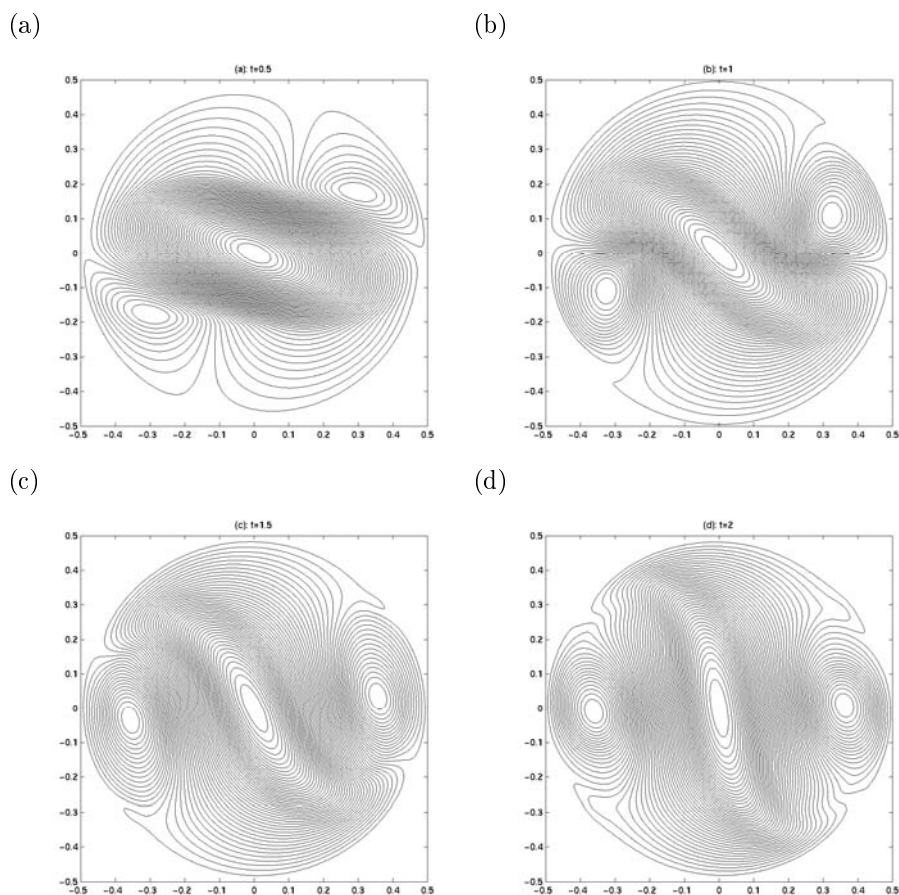


FIG. 5.2. The contour plots of the stream function at different time $t = 0.5, 1, 1.5, 2$ with the Reynolds number $Re = 10^4$. The computation is done using a 384×384 grid.

stream-function formulation is developed for the incompressible Navier-Stokes equations defined on irregular domains. The key to the new method is the fast Poisson solver on irregular domains and the corresponding Thom's and Wilkes' formulae on curved boundaries.

Acknowledgment. The first author is partially supported by a USA ARO grants 39676-MA and 43751-MA, a USA NSF grant DMS-0073403, and a USA NSF/NIH grant DMS/NIGMS-0201094.

REFERENCES

- [1] J. Adams, P. Swarztrauber, and R. Sweet. Fishpack, <http://www.netlib.org/fishpack/>.
- [2] A. Almgren, J. Bell, P. Collella, and T. Marthaler, *A cartesian grid projection method for the incompressible Euler equations in complex geometries*. SIAM J. Sci. Comput., 18:1289–1309, 1997.
- [3] J.B. Bell, P. Colella, and H.M. Glaz, *A second-order projection method for the incompressible Navier-Stokes equations*. J. Comput. Phys., 85:257–283, 1989.

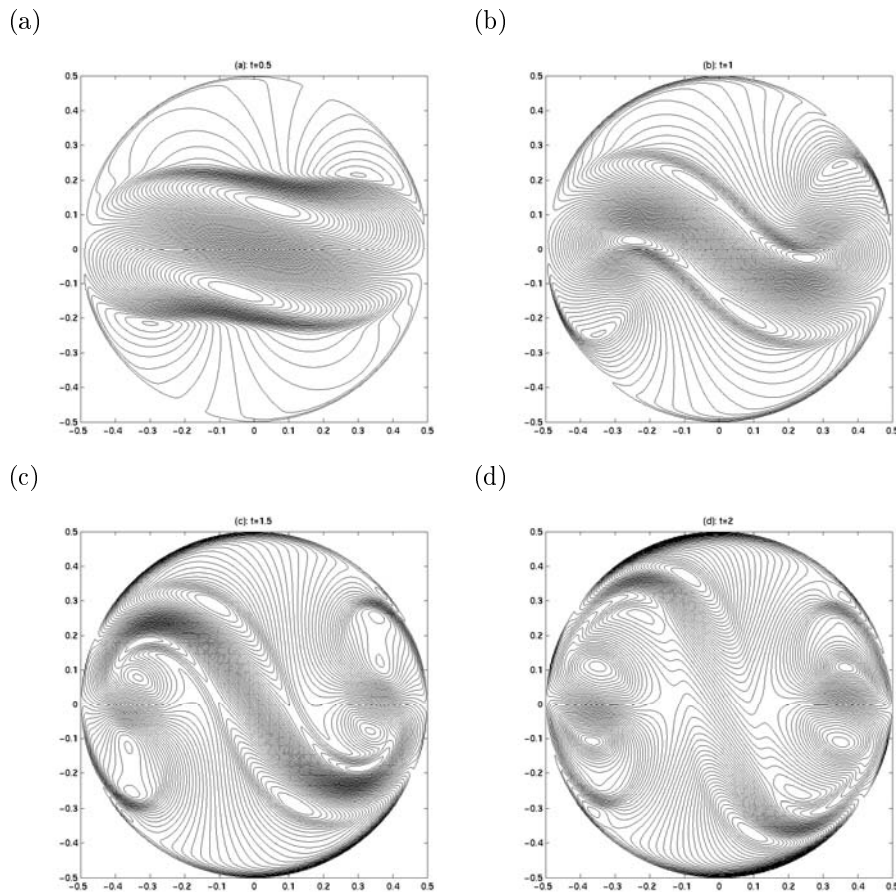


FIG. 5.3. The contour plots of the horizontal velocity with the same physical parameters and resolution as those in FIG. 5.2.

- [4] J. Bramble and J. King, *A finite element method for interface problems in domains with smooth boundaries and interfaces*. Advances in Comput. Math., 6:109–138, 1996.
- [5] D.L. Brown, R. Cortez, and M.L. Minion, *Accurate projection methods for the incompressible Navier-Stokes equations*. J. Comput. Phys., 168:464, 2001.
- [6] D. Calhoun, *A cartesian grid method for solving the streamfunction-vorticity equation in irregular regions*. J. Comput. Phys., 176:231–275, 2002.
- [7] R. Cortez and M.L. Minion, *The blob projection method for immersed boundary problems*. J. Comput. Phys., 161:428–453, 2000.
- [8] R.H. Dillon, L.J. Fauci, and A.L. Fogelson, *Modeling biofilm processes using the immersed boundary method*. J. Comput. Phys., 129:57–73, 1996.
- [9] W. E. and J. Liu, *Essentially compact schemes for unsteady viscous incompressible flows*. J. Comput. Phys., 126:122–138, 1996.
- [10] W. E. and J. Liu, *Vorticity boundary condition and related issues for finite difference schemes*. J. Comput. Phys., 124:368–382, 1996.
- [11] A.L. Fogelson, *Continuum models for platelet aggregation: Formulation and mechanical properties*. SIAM J. Numer. Anal., 52:1089, 1992.
- [12] T. Hou, Z. Li, S. Osher, and H. Zhao, *A hybrid method for moving interface problems with application to the Hele-Shaw flow*. J. Comput. Phys., 134:236–252, 1997.

- [13] T.Y. Hou and B.T.R. Wetton, *Convergence of a finite difference scheme for the Navier-Stokes equations using vorticity boundary conditions*. SIAM J. Numer. Anal., 29:615–639, 1992.
- [14] H. Johansen and P. Colella, *A Cartesian grid embedded boundary method for Poisson equations on irregular domains*. J. Comput. Phys., 147:60–85, 1998.
- [15] M. Kang, R. Fedkiw, and X. Liu, *A boundary condition capturing method for multiphase incompressible flow*. J. Sci. Comput, 15:323–360, 2000.
- [16] M-C. Lai and Z. Li, *The immersed interface method for the Navier-Stokes equations with singular forces*. J. Comput. Phys., 171:822–842, 2001.
- [17] R.J. LeVeque, *Clawpack and Amrclaw—Software for High-Resolution Godunov Methods*. 4-th Intl. Conf. on Wave Propagation, Golden, Colorado, 1998.
- [18] Z. Li, *A fast iterative algorithm for elliptic interface problems*. SIAM J. Numer. Anal., 35:230–254, 1998.
- [19] Z. Li, *IIMPACT, a collection of fortran codes for interface problems*. <http://www4.ncsu.edu/~zhilin/iimpack>, or anonymous ftp.ncsu.edu in the directory: /pub/math/zhilin/Package, Last updated: 2001.
- [20] Z. Li, H. Zhao, and H. Gao, *A numerical study of electro-migration voiding by evolving level set functions on a fixed cartesian grid*. J. Comput. Phys., 152:281–304, 1999.
- [21] A. Mayo, *The fast solution of Poisson's and the biharmonic equations on irregular regions*. SIAM J. Numer. Anal., 21:285–299, 1984.
- [22] A. McKenney, L. Greengard, and Anita Mayo, *A fast poisson solver for complex geometries*. J. Comput. Phys., 118, 1995.
- [23] S. Osher and J.A. Sethian, *Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations*. J. Comput. Phys., 79:12–49, 1988.
- [24] C.S. Peskin and D.M. McQueen, *A general method for the computer simulation of biological systems interacting with fluids*. Symposia of the Society for Experimental Biology, 49:265, 1995.
- [25] W. Proskurowski and O. Widlund, *On the numerical solution of Helmholtz's equation by the capacitance matrix method*. Math. Comp., 30:433–468, 1976.
- [26] E.G. Puckett, A.S. Almgren, J.B. Bell, D.L. Marcus, and W.J. Rider, *A high-order projection method for tracking fluid interfaces in variable density incompressible flows*. J. Comput. Phys., 130:269–282, 1997.
- [27] Y. Saad, *GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems*. SIAM J. Sci. Stat. Comput., 7:856–869, 1986.
- [28] D. Sulsky and J.U. Brackbill, *A numerical method for suspension flow*. J. Comput. Phys., 96:339–368, 1991.
- [29] S.O. Unverdi and G. Tryggvason, *A front-tracking method for viscous, incompressible, multi-fluid flows*. J. Comput. Phys., 100:25–37, 1992.
- [30] C. Wang and J. Liu, *Analysis of finite difference schemes for unsteady Navier-Stokes equations in vorticity formulation*. Numer. Math., 91:543–576, 2002.
- [31] Z. Yang, *A Cartesian grid method for elliptic boundary value problems in irregular regions*. PhD thesis, University of Washington, 1996.
- [32] T. Ye, R. Mittal, H.S. Udaykumar, and W. Shyy, *An accurate Cartesian grid method for viscous incompressible flows with complex immersed boundary*. J. Comput. Phys., 156:209–240, 1999.