INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2008; **57**:1527–1543
Published online 5 December 2007 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1694

A staggered grid-based explicit jump immersed interface method for two-dimensional Stokes flows

Vita Rutka*, †

Fachbereich Mathematik und Statistik, Universitaetsstrasse 10, Fach D 194, D-78457 Konstanz, Germany

SUMMARY

In this paper the explicit jump immersed interface method (EJIIM) is applied to stationary Stokes flows. The boundary value problem in a general, non-grid aligned domain is reduced by the EJIIM to a sequence of problems in a rectangular domain, where staggered grid-based finite differences for velocity and pressure variables are used. Each of these subproblems is solved by the fast Stokes solver, consisting of the pressure equation (known also as conjugate gradient Uzawa) method and a fast Fourier transform-based Poisson solver.

This results in an effective algorithm with second-order convergence for the velocity and first order for the pressure. In contrast to the earlier versions of the EJIIM, the Dirichlét boundary value problem is solved very efficiently also in the case when the computational domain is not simply connected. Copyright © 2007 John Wiley & Sons, Ltd.

Received 21 March 2007; Revised 6 August 2007; Accepted 14 October 2007

KEY WORDS: Stokes equations; finite differences; staggered grid; interface problem; discontinuous solutions

1. INTRODUCTION

We consider the stationary Stokes equations for the velocity $\mathbf{u}=(u,v)^\top$ and the pressure p in a bounded domain $\Omega\subset\mathbb{R}^2$

$$-\Delta\mathbf{u}+\nabla p=\mathbf{f} \quad \text{in } \Omega \tag{1}$$

$$-\nabla\cdot\mathbf{u}=0 \quad \text{in } \Omega \tag{2}$$

$$\mathbf{u}=\mathbf{u}_{\mathrm{D}} \quad \text{at } \partial\Omega \tag{3}$$

$$\int_\Omega p=0 \tag{4}$$

*Correspondence to: Vita Rutka, Fachbereich Mathematik und Statistik, Universitaetsstrasse 10, Fach D 194, D-78457 Konstanz, Germany.
†E-mail: vita.rutka@uni-konstanz.de

where $\mathbf{f}=(f^1, f^2)^\top$ is the applied force. Condition (4) is necessary for the uniqueness of the pressure. All problem data (Ω, $\mathbf{f}$ and $\mathbf{u}_D$) are assumed to be sufficiently smooth so that the solution belongs to the class $\mathbf{u}\in C^4(\Omega)$, $p\in C^3(\Omega)$ (we refer to classical books such as [1, 2] for the existence theory and regularity results).

The main motivation of this work was the necessity to solve Stokes problems with different right-hand sides as required for the simulation of rigid particle transport in a viscous polymer [3]. Further, the presented approach can be seen as an improvement of the first-order method proposed in [4, 5]. The geometry is typically given in a voxelized description, so that a first-order treatment is very natural. However, if a more accurate geometry description is available, a second-order method can help to extract more information about the flow and its properties. Finally, also as a building block in algorithms for Navier–Stokes equations, a fast and accurate Stokes solver is of high importance.

The original immersed interface method (IIM) [6] has been exploited for solving various Stokes problems in [7–10] explaining how to deal with the pressure boundary condition. In contrast to our approach, these methods are based on solving three Poisson problems on a regular, non-staggered grid.

2. NUMERICAL METHOD

In this section we apply the explicit jump immersed interface method (EJIIM) [11, 12] to Stokes flows. At first, discretization in an 'easy' rectangular domain is recalled and then the EJIIM approach is used to treat domains of a general shape.

2.1. Discretization of Stokes equations in a rectangular domain

Consider at first a rectangular domain B

$$B:=\{(x, y)\in\mathbb{R}^2 | a_x\leqslant x\leqslant b_x, a_y\leqslant y\leqslant b_y\},\quad a_x, b_x, a_y, b_y\in\mathbb{R} \text{ and } a_x<b_x, a_y<b_y$$

For technical reasons, we allow the fluid to be compressible in this section and replace (2) by

$$-\nabla\cdot\mathbf{u}=g \quad \text{in } B \tag{5}$$

whenever considering the Stokes equations (1)–(3) in the domain B.

2.1.1. Staggered grid. To discretize equations (1), (3), (5), we choose finite differences on a staggered grid (see, e.g. [13] for an introduction or [14, 15]). Let $n_x, n_y\in\mathbb{N}$ be the number of grid points in the x and y directions, where the numbering is started at zero. The corresponding mesh widths are $h_x=(b_x-a_x)/n_x$ and $h_y=(b_y-a_y)/n_y$. Only to simplify the nomenclature, it is assumed from now on that $n_x=n_y=:n$ and $h_x=h_y=:h$.

The staggered grid is composed of three different meshes named as *u-mesh*, *v-mesh* and *p-mesh* (see Figure 2(b) for an illustration). With $x_i:=a_x+ih$, $y_j:=a_y+jh$, $i, j\in\{0, 1, \ldots, n\}$, $x_{i+1/2}:=a_x+(i+\frac{1}{2})h$ and $y_{j+1/2}:=a_y+(j+\frac{1}{2})h$, $i, j\in\{0, 1, \ldots, n-1\}$ we have (see Figure 2(b))

- *u-mesh* (black circles and solid lines) with $u_{ij}:=u(x_i, y_{j+1/2})$,
- *v-mesh* (white circles and dashed lines) with $v_{ij}:=v(x_{i+1/2}, y_j)$,
- *p-mesh* (crosses) with $p_{ij}:=p(x_{i+1/2}, y_{j+1/2})$.

DOI: 10.1002/fld

On such a grid, the following second-order-consistent approximation of Equations (1), (3), (5), is chosen:

$$-\frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{h^2}-\frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{h^2}+\frac{p_{i,j}-p_{i-1,j}}{h}=f^1_{i,j} \tag{6}$$

$$-\frac{v_{i+1,j}-2v_{i,j}+v_{i-1,j}}{h^2}-\frac{v_{i,j+1}-2v_{i,j}+v_{i,j-1}}{h^2}+\frac{p_{i,j}-p_{i,j-1}}{h}=f^2_{i,j} \tag{7}$$

$$-\frac{1}{h}(u_{i+1,j}-u_{i,j})-\frac{1}{h}(v_{i,j+1}-v_{i,j})=g_{i,j} \tag{8}$$

for $i,j=1,2,\ldots,n-1$, where $f^1_{i,j}:=f^1(x_i,y_{j+1/2})$, $f^2_{i,j}:=f^2(x_{i+1/2},y_j)$. The last equation is the approximation of (5) at the grid points of the p-mesh.

Remark 1
In general, the staggered grid approach may be problematic due to shifted Dirichlét boundary conditions on u and v meshes. The IIMs, however, overcome this difficulty in a natural way as it will be shown later and we can assume the Dirichlét boundary conditions to be given along the shifted boundaries without any loss of accuracy.

With $U:=(u_{0,0},u_{1,0},\ldots,u_{n,n-1})^\top$, $V:=(v_{0,0},v_{1,0},\ldots,v_{n-1,n})^\top$ and $P:=(p_{0,0},p_{1,0},\ldots,p_{n-1,n-1})^\top$ we denote the discrete solution vectors of both velocity components and pressure. Then the discrete Stokes system (6)–(8) together with Dirichlét boundary conditions along the shifted boundaries is expressed as

$$\mathbf{A}S=R \tag{9}$$

with

$$\mathbf{A}:=\begin{pmatrix}-\Delta_h & \nabla_h\\ -\nabla_h\cdot & \mathbf{0}\end{pmatrix},\quad \Delta_h:=\begin{pmatrix}\Delta_h^u & \mathbf{0}\\ \mathbf{0} & \Delta_h^v\end{pmatrix},\quad W:=\begin{pmatrix}U\\ V\end{pmatrix},\quad S:=\begin{pmatrix}W\\ P\end{pmatrix}$$

Here, Δ_h^u and Δ_h^v stand for discrete Laplace operators with Dirichlét boundary conditions on u and v meshes, ∇_h is the discrete gradient, where the x-derivative operator is approximated on the u-mesh and the y-derivative on the v-mesh as in (6), (7). The operator $\nabla_h\cdot$ is the discrete divergence according to (8). Notation $\mathbf{0}$ is used for zero matrices with appropriate dimensions. The right-hand side R is given by

$$R=\begin{pmatrix}F\\ G\end{pmatrix},\quad F=\begin{pmatrix}F^1\\ F^2\end{pmatrix}$$

where F^1 and F^2 are the values of the right-hand side term on u and v grids, respectively, and G is the vector corresponding to the discrete right-hand side in (8).

To remove the rank deficiency of system (9) due to the non-uniqueness of the pressure, we approximate condition (4) and require in addition that

$$\sum_{i,j=0}^{n-1}p_{i,j}=0 \tag{10}$$

DOI: 10.1002/fld

2.1.2. Discrete compatibility condition. In order for the overdetermined linear system (9), (10) to have a solution, the right-hand side R has to lie in the image space of the operator $\mathbf{A}$.

In the continuous case, the mass conservation equation (5) in the domain B together with the Dirichlét boundary condition (3) leads to the following compatibility condition (see, e.g. [1, p. 183]):

$$-\nabla\cdot\mathbf{u}=g\Rightarrow\int_{\partial B}\mathbf{u}\cdot\mathbf{n}\overset{!}{=}\int_B g \tag{11}$$

where $\mathbf{n}$ is the outer normal vector to the domain B.

An analogous condition has to hold also in the discrete case. Summation over all grid points of the p-mesh in Equation (8) leads to the *discrete compatibility condition*

$$\sum_{i,j=0}^{n-1} g_{i,j}\overset{!}{=}-\frac{1}{h}\sum_{j=0}^{n-1}(u_{n,j}-u_{0,j})-\frac{1}{h}\sum_{i=0}^{n-1}(v_{i,n}-v_{i,0}) \tag{12}$$

Remark 2
Even if some function g and boundary conditions $(u_{\mathrm{D}}, v_{\mathrm{D}})^{\top}$ satisfy the continuous compatibility condition (11), it still can happen that the discrete version (12) does not hold. In such cases, a small correction has to be added to the discrete right-hand side and/or boundary data.

2.2. Fast Stokes solver

In the EJIIM approach, the discrete Stokes system (9) has to be solved repeatedly as a part of the iteration, thus requiring an excellent performance of this step. We have chosen the *pressure equation method* [16] coupled with a fast Fourier transform (FFT)-based Laplace solver [17]. The algorithm is based on the Schur complement of system (9) for the pressure variable P

$$\mathbf{M}P=Q\quad\text{where }\mathbf{M}=-\nabla_h\cdot(\Delta_h^{-1}\nabla_h),\quad Q=G-\nabla_h\cdot(\Delta_h^{-1}F) \tag{13}$$

Applying a fixed point iteration to the linear system (13) leads to the *Uzawa algorithm* [2, 16], whereas the conjugate gradient method leads to the so-called *pressure equation method* [16], known also as CG *via Uzawa* (see, for example, [18]).

Note that each matrix–vector product with $\mathbf{M}$ involves one application of Δ_h^{-1}, that is, one solve of the discrete Laplace equation with Dirichlét boundary conditions. Here, we use the FFT-based fast solver from [17]. In the last step, the pressure is projected onto the space of functions satisfying (10) by adding a suitable constant. Once the pressure P has been found, the velocity variable W is obtained using one subsequent Poisson solve from

$$W=-\Delta_h^{-1}(F-\nabla_h P)$$

The importance of the compatibility condition (12) is illustrated by the following example.

Example 2.1
Consider solving of the linear system (9) in the unit square with $h=\frac{1}{10}$ and $n=10$. The right-hand side vector F is generated randomly. For the velocity component, the homogeneous Dirichlét boundary condition has been set along the shifted boundaries. For the vector G, we take the

DOI: 10.1002/fld

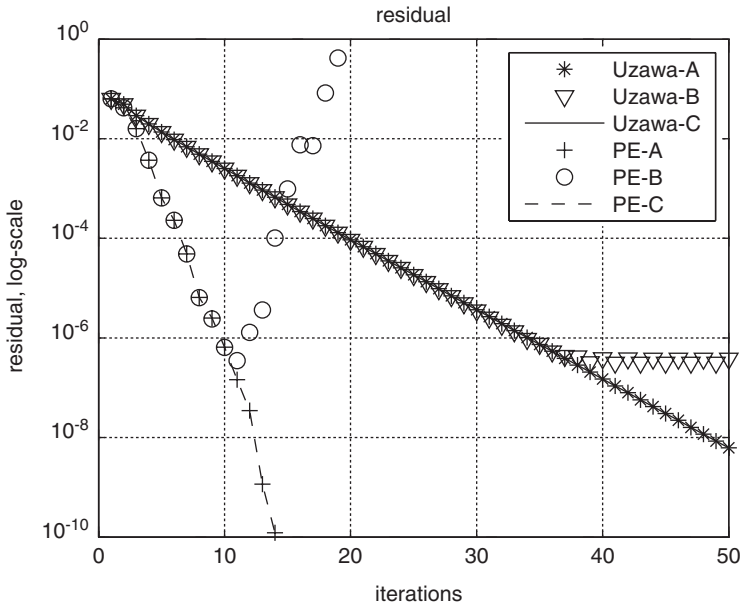

Figure 1. Influence of the compatibility condition (12) in Example 2.1 for pressure equation (PE) and Uzawa algorithms. Condition (12) holds only in the cases (A) and (C) and not in (B).

following possibilities:

(A) $G=0$ everywhere.
(B) $G=10^{-5}$ at one random grid point, otherwise zero.
(C) $G=10^{-5}$ at one random grid point, $-\frac{1}{2}10^{-5}$ at two other grid points.

The compatibility condition (12) is satisfied only in cases (A) and (C).

Figure 1 demonstrates the enormous influence of the compatibility condition. Especially, for the pressure equation approach, small perturbations in the right-hand side data can lead to a blow up of the method.

2.3. EJIIM approximation of Stokes equations in general domains

Once it is clear how to deal with the Stokes equations in rectangular domains, we can go over to computational domains Ω with general shapes. We follow the standard EJIIM approach for boundary value problems [11, 12] (see also [19, 20]) by embedding the original domain into a rectangular domain B and extending in $\Omega^c := B\setminus\Omega$ the solution $(\mathbf{u}, p)$ with zero. Then the original boundary becomes an artificial interface Γ, along which the solution is, in general, discontinuous (Figure 2(a)).

Definition 1
Let $q(x, y)$ be a piecewise continuous function with a possible discontinuity along the interface $\Gamma := \overline{\Omega}\cap\overline{\Omega^c}$. Let $(x^*, y^*)\in\Gamma$. Then we define the *one-sided values of* q as

$$q^+(x^*, y^*) := \lim_{\Omega^c \ni (x,y)\to(x^*,y^*)} q(x, y), \quad q^-(x^*, y^*) := \lim_{\Omega \ni (x,y)\to(x^*,y^*)} q(x, y)$$

DOI: 10.1002/fld

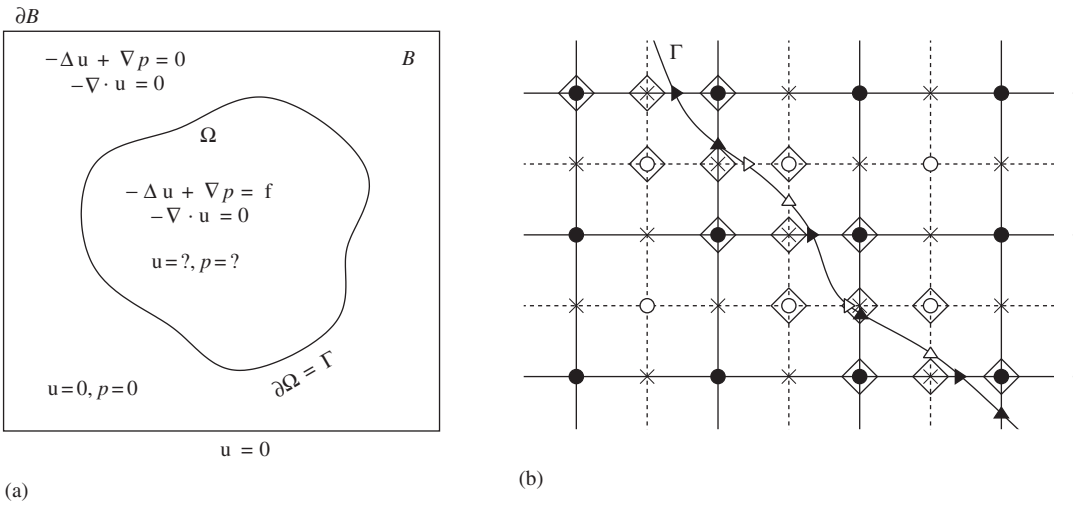

Figure 2. Illustration to the embedding idea: (a) Immersed interface embedding of the domain Ω in a rectangular box B. In $\Omega^c := B \backslash \Omega$ solution is extended by zero. (b) Irregular grid points and intersections near the interface. Black circles and solid lines: u-mesh with intersections marked by black triangles oriented to the right for x-intersections and upward for y-intersections. White circles, dashed lines and white triangles: v-mesh, crosses: p-mesh. The irregular points of all meshes are marked by rhombs.

and the *jump*

$$[q(x^*, y^*)] := q^+(x^*, y^*) - q^-(x^*, y^*)$$

As basis for the EJIIM, a standard mesh from Section 2.1.1 over the domain B is imposed. We classify the grid points as regular or irregular (Figure 2(b)):

Definition 2
A grid point $(x_i, y_{j+1/2})$ for some $i, j \in \{1, 2, \ldots, n-1\}$ of the u-mesh is called *regular* (with respect to discretization (6)–(8)) if the stencil of (6) is not cut by the interface Γ. Otherwise this grid point is called *irregular*. Analogously, the regular and irregular points of v and p meshes are defined by considering the stencils of (7) and (8), respectively.

Points where the interface Γ cuts the grid lines parallel to the x-axis are called *x-intersections*, points where Γ cuts the grid lines parallel to the y-axis are called *y-intersections*.

At all regular points, the $\mathcal{O}(h^2)$ accurate discretization (6)–(8) is valid. The irregular points require more attention due to the discontinuities in functions and their derivatives.

2.3.1. Corrected finite differences. At the irregular grid points the truncation error is improved using the *corrected finite differences* according to the following lemma [11, 12].

Lemma 1
Let $x_j \leqslant \alpha < x_{j+1}$, $h^- = x_j - \alpha$ and $h^+ = x_{j+1} - \alpha$. Suppose $w \in C^4([x_j - h, \alpha)) \cap C^4((\alpha, x_{j+1} + h])$, with derivatives extending continuously up to the boundary α. Then the following approximations

DOI: 10.1002/fld

hold up to $\mathcal{O}(h)$:

$$w_x(x_j) \approx \frac{w(x_{j+1}) - w(x_{j-1})}{2h} - \frac{1}{2h} \sum_{m=0}^{1} \frac{(h^+)^m}{m!} [d^m w] \tag{14}$$

$$w_x(x_{j+1}) \approx \frac{w(x_{j+2}) - w(x_j)}{2h} - \frac{1}{2h} \sum_{m=0}^{1} \frac{(h^-)^m}{m!} [d^m w] \tag{15}$$

$$w_{xx}(x_j) \approx \frac{w(x_{j+1}) - 2w(x_j) + w(x_{j-1})}{h^2} - \frac{1}{h^2} \sum_{m=0}^{2} \frac{(h^+)^m}{m!} [d^m w] \tag{16}$$

$$w_{xx}(x_{j+1}) \approx \frac{w(x_{j+2}) - 2w(x_{j+1}) + w(x_j)}{h^2} + \frac{1}{h^2} \sum_{m=0}^{2} \frac{(h^-)^m}{m!} [d^m w] \tag{17}$$

Remark 3
Extending the sum to one more term in (14), (15) leads to a second-order truncation error, as stated in the original version of Lemma 1 in [11, 12].

Remark 4
Now the necessity for the generalized incompressibility condition (5) becomes apparent. As an illustration we consider an x-type intersection with the interface parallel to the y-axis between the grid points x_i and x_{i+1} of the u-mesh. For simplicity, we assume the v-component of the velocity to be constant. Then, instead of approximation (8) we choose

$$0 = -\frac{1}{h}(u_{i+1,j} - u_{i,j}) - \frac{1}{h}(v_{i,j+1} - v_{i,j}) + \frac{1}{h}([u] + h^+[\partial_x u])$$

which can be viewed as the original divergence approximation (8) with a correction term playing the role of the compressibility function g in (5).

The domain B and Equations (1), (2) are discretized as outlined in Section 2.1.1. However, the discontinuity of the solution near the interface Γ has to be taken into account: at each node where the interface cuts the finite difference stencil, appropriate correction terms involving the jumps in the function and its derivatives are used according to Lemma 1. EJIIM introduces these jumps as additional variables in the system. More specifically we have:

- At x-intersections of the u-mesh: These intersections affect the approximation of $\partial_{xx} u$ and according to Lemma 1 we need $[u]$, $[\partial_x u]$ and $[\partial_{xx} u]$. Note that, in Equation (6), the term $(1/h)(p_{i,j} - p_{i-1,j})$ is a central finite difference approximation of the derivative $\partial_x p(x_i, y_{j+1/2})$. According to (14), (15) we therefore need the jumps $[p]$ and $[\partial_x p]$ in order to obtain a first-order truncation error.
- At y-intersections of the u-mesh: We need to correct only the $\partial_{yy} u$ derivative in (6) demanding $[u]$, $[\partial_y u]$ and $[\partial_{yy} u]$.
- Analogously we find that discretization (7) requires
  - $[v]$, $[\partial_x v]$ and $[\partial_{xx} v]$ at x-intersections of the v-mesh and
  - $[v]$, $[\partial_y v]$, $[\partial_{yy} v]$, $[p]$ and $[\partial_y p]$ at the y-intersections of the v-mesh.
- Intersections of Γ with the p-mesh affect the derivatives in (8). For correcting the approximation $(1/h)(u_{i+1,j} - u_{i,j}) \approx \partial_x u(x_{i+1/2}, y_j)$, we would need $[u]$ and $[\partial_x u]$ at the x-intersections

DOI: 10.1002/fld

of the p-mesh. Note, however, that x-intersections of the p-mesh coincide with the x-intersections of u-mesh and that all necessary jumps are already available. In fact, it is not related with any extra cost to achieve even a second-order truncation error by adding one more term with $m=2$ in the corrected finite difference formula (14), as the $[\partial_{xx}u]$ jump is necessary for discretizing the $\partial_{xx}u$ term anyway.

Analogously the approximation $(1/h)(v_{i,j+1}-v_{i,j})\approx\partial_y v(x_i, y_{j+1/2})$ would require $[v]$ and $[\partial_y v]$ jumps at the y-intersections of the p-mesh. Also, here no additional variables need to be introduced thanks to the coincidence of the y-intersections on v- and p-meshes.

The corrected finite differences can be expressed in compact form as

$$\mathbf{A}S+\mathbf{\Psi}J=R \tag{18}$$

where J is the vector of jumps and $\mathbf{\Psi}$ contains the coefficients of the correction terms according to Lemma 1. Matrix $\mathbf{\Psi}$ is sparse and has non-zero entries only in the rows corresponding to the irregular grid points.

2.3.2. Imposing the compatibility condition. With a known jump vector J, the solvability of system (18) has to be assured, that is, $R-\mathbf{\Psi}J$ and homogeneous Dirichlét boundary conditions for the velocity along ∂B have to satisfy the discrete compatibility condition (12).

At first we rewrite $\mathbf{\Psi}$ in form

$$\mathbf{\Psi}=\begin{pmatrix}\mathbf{\Psi}^{\text{impulse}}\\ \mathbf{\Psi}^{\text{mass}}\end{pmatrix} \quad \text{so that } \mathbf{A}S+\mathbf{\Psi}J=\begin{pmatrix}\Delta_h W+\nabla_h P+\mathbf{\Psi}^{\text{impulse}}J\\ \nabla_h\cdot W+\mathbf{\Psi}^{\text{mass}}J\end{pmatrix}$$

Then the approximation of the incompressibility condition (2) according to the corrected finite differences described before is

$$\nabla_h\cdot W+\mathbf{\Psi}^{\text{mass}}J=0$$

System (18) is solvable under the condition that

$$\sum_{(x_{i_k},y_{j_k})\in\mathfrak{G}}(\mathbf{\Psi}^{\text{mass}}J)_k=\sum_{(x_{i_k},y_{j_k})\in\mathfrak{G}^{\text{irreg}}}(\mathbf{\Psi}^{\text{mass}}J)_k\overset{!}{=}0 \tag{19}$$

as the right-hand side of (12) is zero due to extension of the velocity by zero in Ω^c. The notation $\mathfrak{G}$ stands for the set of all p-grid points and $\mathfrak{G}^{\text{irreg}}$ is the set of all irregular points of the p-mesh. We have used that $\mathbf{\Psi}^{\text{mass}}J$ is zero at all regular points of the p-mesh.

Unfortunately, due to the truncation errors, condition (19) cannot be guaranteed to hold even for the exact solution of Stokes equations (1)–(3) and the related jumps. To overcome this difficulty, the operator $\mathbf{\Psi}^{\text{mass}}$ is replaced by $\tilde{\mathbf{\Psi}}^{\text{mass}}$ defined as

$$(\tilde{\mathbf{\Psi}}^{\text{mass}}J)_k=\begin{cases}(\mathbf{\Psi}^{\text{mass}}J)_k-m(J) & \text{if } (x_{i_k},y_{j_k})\in\mathfrak{G}^{\text{irreg}}\\ 0 & \text{else}\end{cases}$$

for $k=1,2,\ldots,(n-1)(n-1)$ with

$$m(J)=\frac{1}{N}\sum_{(x_{i_k},y_{j_k})\in\mathfrak{G}^{\text{irreg}}}(\mathbf{\Psi}^{\text{mass}}J)_k,\quad N:=\#(\mathfrak{G}^{\text{irreg}})$$

DOI: 10.1002/fld

Instead of (18) in further calculations we use

$$AS+\tilde{\mathbf{\Psi}}J=R \quad \text{where } \tilde{\mathbf{\Psi}}=\begin{pmatrix}\mathbf{\Psi}^{\text{impulse}}\\ \tilde{\mathbf{\Psi}}^{\text{mass}}\end{pmatrix} \tag{20}$$

The truncation error after this replacement can be shown to be $\mathcal{O}(h)$ at all irregular points of the p-mesh.

Lemma 2
Let W^{ex} be the exact velocity components of the solution to (1)–(3) (extended by zero in Ω^c and restricted to the grid points). Let J^{ex} be the vector of the exact jumps restricted to the intersection points. Then the truncation error of the discrete mass conservation equation is

$$-\nabla_h\cdot W^{\text{ex}}+\tilde{\mathbf{\Psi}}^{\text{mass}}J^{\text{ex}}=\begin{cases}\mathcal{O}(h^2) & \text{at all regular grid points}\\ \mathcal{O}(h) & \text{at irregular grid points}\end{cases}$$

Proof
By using also the jumps in second-order derivatives of the velocity components, we have the second-order approximation at all points of the p-mesh:

$$-\nabla_h\cdot W^{\text{ex}}+\mathbf{\Psi}^{\text{mass}}J^{\text{ex}}=\mathcal{O}(h^2) \tag{21}$$

Now we sum the above expression over all grid points. Owing to homogeneous Dirichlét boundary conditions on $\mathbf{u}$, we have

$$\sum_{(x_{i_k},y_{j_k})\in\mathfrak{G}}(\nabla_h\cdot W^{\text{ex}})_k=0$$

implying

$$Nm(J)=\sum_{(x_{i_k},y_{j_k})\in\mathfrak{G}^{\text{irreg}}}(\mathbf{\Psi}^{\text{mass}}J^{\text{ex}})_k=\sum_{(x_{i_k},y_{j_k})\in\mathfrak{G}}(\mathbf{\Psi}^{\text{mass}}J^{\text{ex}})_k=\sum_{(x_{i_k},y_{j_k})\in\mathfrak{G}}\mathcal{O}(h^2)=\mathcal{O}(1)$$

In the expression above we have used that $\mathbf{\Psi}^{\text{mass}}J^{\text{ex}}$ is non-zero only at the irregular grid points. Thus, $m(J)=(1/N)\mathcal{O}(1)=\mathcal{O}(h)$ as $N=\mathcal{O}(1/h)$. □

Remark 5
In [21, p. 74], a very similar approach is used by subtracting the mean value of the correction term. The essential difference is that, here, we perturb the approximation *only* at the irregular grid points, whereas in [21] a small $\mathcal{O}(h^2)$ constant is added at *all* grid points. Our approach allows to preserve the local influence of the operator $\mathbf{\Psi}$.

2.3.3. Jump conditions. The corrected finite differences (20) involve the unknown jump vector J. To close the system, additional conditions relating the primary variables (W, P) and the jump vector J are necessary. They are constructed in two steps.

- Derive the so-called *jump conditions* by expressing all necessary jumps in terms of the known quantities and one-sided values of the unknown solution and its derivatives.

DOI: 10.1002/fld

- The necessary one-sided values at the intersections are approximated by (unknown) solution values at the grid points using an appropriate *extrapolation*.

We start by deriving the jump conditions for the Stokes equations.

Jumps in the velocity.

- *Zeroth-order jumps*. Thanks to the Dirichlét boundary condition, the jump in the velocity is known explicitly

$$[\mathbf{u}] = -\mathbf{u}_{\mathrm{D}} \tag{22}$$

- *First-order jumps*. We start by introducing the local velocity $(\xi, \eta) := (\mathbf{u}\cdot\mathbf{n}, \mathbf{u}\cdot\mathbf{t})$, where $\mathbf{n} = (n_1, n_2)^{\top}$ and $\mathbf{t} = (-n_2, n_1)^{\top}$ are normal and tangent vectors of the interface Γ. The incompressibility condition (2) implies that $[\partial_n \xi] + [\partial_t \eta] = 0$.

  For the local velocities, the following set of jump conditions is used:

$$[\partial_n \xi] = -\partial_n \xi, \quad [\partial_n \eta] = -[\partial_n \xi] = \partial_n \xi$$

$$[\partial_t \xi] = -\partial_t \xi, \quad [\partial_t \eta] = -\partial_t \eta$$

  Transforming the above expressions back to (u, v) components leads to

$$[\partial_x u] = -n_1^2 \partial_x u^- + n_2^2 \partial_y v^- \tag{23}$$

$$[\partial_y u] = -\partial_y u^- \tag{24}$$

$$[\partial_x v] = -\partial_x v^- \tag{25}$$

$$[\partial_y v] = n_1^2 \partial_x u^- - n_2^2 \partial_y v^- \tag{26}$$

- *Second-order jumps*. Here we just set

$$[\partial_{xx} \mathbf{u}] = -\partial_{xx} \mathbf{u}^-, \quad [\partial_{yy} \mathbf{u}] = -\partial_{yy} \mathbf{u}^- \tag{27}$$

  Jumps in the mixed derivatives $[\partial_{xy} \mathbf{u}]$ are not necessary for the Stokes equations.

Jumps in the pressure

- *Zeroth-order-jumps*. We use the trivial jump condition

$$[p] = -p^- \tag{28}$$

- *First-order jumps*. Exploiting the impulse conservation equations (1) yields

$$[\partial_x p] = [f^1] + [\Delta u] = [f^1] - \Delta u^- \tag{29}$$

$$[\partial_y p] = [f^2] + [\Delta v] = [f^2] - \Delta v^- \tag{30}$$

- *Second-order jumps* are not necessary for preserving the first-order truncation error at the interface.

In the next step, the one-sided values appearing in the jump conditions (22)–(30) have to be expressed with the function values at the grid points by an appropriate extrapolation. Here we use

DOI: 10.1002/fld

two approaches: the classical polynomial fit as described in [12] and a modified version where the Dirichlét boundary is explicitly imposed by following the ideas in [22].

Then, system (20) can be closed by additional equations

$$\mathbf{D}S + J = T \tag{31}$$

with an appropriate matrix $\mathbf{D}$ and the right-hand side T. For more details we refer to the first works on EJIIM [11, 12].

2.3.4. Solving the EJIIM system. System (20), (31) is solved in a similar manner as other boundary value problems [11, 12, 19]. At first we write the Schur complement for the jump variable J:

$$(\mathbf{I} - \mathbf{D}\mathbf{A}^{-1}\tilde{\mathbf{\Psi}})J = T - \mathbf{D}\mathbf{A}^{-1}R \tag{32}$$

The matrix $\mathbf{B} := \mathbf{I} - \mathbf{D}\mathbf{A}^{-1}\tilde{\mathbf{\Psi}}$ is dense and cannot be stored explicitly. The way out is offered by Krylov-space methods (see, for example, [23]) such as GMRES, which only require matrix–vector products with $\mathbf{B}$. In each step a standard discrete Stokes problem of type (9) has to be solved when applying the operator $\mathbf{A}^{-1}$. This is done using the pressure equation method from Section 2.2.

Remark 6

Already in [11, Section 3.6.1], it has been pointed out for the Poisson problems that EJIIM approach does not have a unique solution if the domain Ω^c is not connected, an example of which arises when computing flows around obstacles in two dimensions.

In particular, extra work is required to assure the zero solution in the components of Ω^c not sharing the boundary ∂B. Here we consider several possible solutions:

1. If nothing is done to impose zeros in Ω^c, we call it '*standard extension*' or '*SE*', which is expected to work in case Ω^c is connected.
2. Impose zero explicitly everywhere in Ω^c by using $u_{ij} = 0$ and $v_{ij} = 0$ instead of (6) and (7). This approach has the best approximation properties; unfortunately, the FFT solver is disabled. We name this '*exact extension*' or '*EE*'.
3. We follow the suggestion in [11, Section 3.8], and introduce additional variables for function values at few points in Ω^c. In our case, we enforce zero values of the velocity at the irregular points of Ω^c. This approach allows one to use FFT for inverting the Stokes matrix $\mathbf{A}$. We refer to it as '*zero ghost extension*' or '*ZGE*'.
4. Instead of the 'classical' EJIIM extrapolation of the one-sided function values as in [19] we use the modified version following [22]. In addition to the least-squares fit for the polynomial coefficients (see [19, 22] for details) we require the one-sided approximations of $\mathbf{u}^-$ to satisfy the Dirichlét boundary conditions. In our computations demonstrated below we have weighted this condition by a factor 10. This will be called '*boundary condition extension*' or '*BCE*'.

Remark 7

Usage of different numerical tools can make the practical realization of the algorithm relatively sophisticated, and a modular approach can be extremely helpful.

DOI: 10.1002/fld

An example of the EJIIM implementation for the Poisson equation by the author and A. Wiegmann is available online at `http://www.math.uni-konstanz.de/~rutka/EjiimEx/DOC/documentation/`.

The routines posted there form the basis also of the programs used in this article for the Stokes equations.

Remark 8
There is a certain similarity between the immersed interface and the boundary integral methods. In fact, the article by Mayo [24] where fast solvers for Poisson and biharmonic equations have been constructed based on integral equation formulations is one of the milestones in the development of the IIMs (see, e.g. the introduction of [11]). Some details and comparison of the EJIIM with the boundary integral methods are discussed in [25; 20, p. 75]

An extension of Mayo's method (embedded boundary integral (EBI) method) to the Stokes system has been offered in [26]. If we compare EBI with EJIIM, we see that solving Equation (32) corresponds to approximately solving an integral equation. In the EJIIM approach, the corresponding matrix on the left-hand side, $\mathbf{I}-\mathbf{D}\mathbf{A}^{-1}\tilde{\mathbf{\Psi}}$, is never computed and stored explicitly. On the one hand, this makes an application of preconditioners difficult as the matrix entries are difficult to obtain. On the other hand, the practical experience (see the following section) shows that iteration counts in solving (32) increase very slowly under the grid refinement (Tables I and II). Thanks to the fast Stokes solver, the application of $\mathbf{I}-\mathbf{D}\mathbf{A}^{-1}\tilde{\mathbf{\Psi}}$ is cheap and the method allows a straightforward extension to three dimensions.

Table I. Number of GMRES iterations to achieve 10^{-8} stopping tolerance in Example 3.1 with different extensions.

	$n=80$	$n=120$	$n=160$	$n=200$	$n=240$	$n=280$	$n=320$
Circle geometry							
SE	21	21	23	23	24	24	24
EE	15	14	15	15	15	15	15
ZGE	83	96	109	120	128	134	143
BCE	22	23	26	27	29	28	32
'*Fish*' *geometry*							
EE	19	19	19	18	17	17	18
ZGE	168	170	205	188	186	207	216
BCE	35	39	40	41	43	45	46

Table II. Number of GMRES iterations needed to achieve 10^{-8} stopping tolerance for the Wannier flow example.

	$n=80$	$n=120$	$n=160$	$n=200$	$n=240$	$n=280$	$n=320$
EE	22	21	22	23	22	23	21
BCE	23	23	25	27	29	30	32

DOI: 10.1002/fld

3. NUMERICAL RESULTS

In this section we demonstrate the performance of the method on two examples with known analytic solutions. The error is computed in the maximum norm.

Example 3.1
We take the following functions as exact solutions to the Stokes equations (1)–(3):

$$u = \frac{1}{x+1}\sin(2\pi y)$$

$$v = -\frac{1}{2\pi}\frac{1}{(x+1)^2}\cos(2\pi y)$$

$$\hat{p} = \exp(xy), \quad p = \hat{p} - \text{mean}(\hat{p})$$

where $\text{mean}(\hat{p})$ is subtracted to ensure condition (4). The corresponding right-hand side $\mathbf{f}$ and the boundary condition $\mathbf{u}_\Gamma$ are found by inserting functions $(u, v)^\top$ and p in (1)–(3).

Here we use two geometries: the circle (Figure 3(a)) with center at $(0.5, 0.51)$ and radius 0.3, and the so-called 'fish' (Figure 3(b)), where the the boundary is given as the zero-level set of the following function F:

$$F(x, y) = ((x-0.5)^8 + 2(y-0.51)^8 - 2.5(x-0.5)^3(y-0.51)^5) \\ -0.002(2(x-0.5)^2 + (y-0.5)^2 - (x-0.5)(y-0.5) - 0.25^2 - 0.3^8)$$

The convergence results are summarized in Figure 4 using the mesh widths $h \in \{\frac{1.2}{60}, \frac{1.2}{70}, \ldots, \frac{1.2}{320}\}$ and the maximum norm of the error. As expected, the convergence is of second order for the velocity and slightly better than first order for the pressure. It is remarkable that the BCE produces results of the same accuracy as the EE.

Table I demonstrates how many GMRES iterations were needed to reduce the norm of the residual to 10^{-8} when solving the Schur complement system (32). As expected, the best iteration counts are achieved with the EE. However, it disables the usage of the FFT-based fast Poisson solver, thus making the whole procedure slow. Important here is the drastic drop of the iteration count for the BCE if compared with the ZGE. Given this performance together with a better solution quality as documented in Figure 4, we can conclude that the BCE approach is preferable.

The stopping tolerance for inverting the operator $\mathbf{A}$ with the pressure equation method was set to 10^{-12} and fast convergence has been observed in all cases (in about 20–30 iterations).

The next example is taken from [27, 28] and allows to test the algorithm in a situation closer to real flows.

Example 3.2 (*Wannier flow*)
The computational domain is situated between two non-concentric rotating cylinders as shown in Figure 5(a). As the expressions for the analytic solution are lengthy, we do not give them here and refer to [27] instead.

In our example, the inner cylinder rotates with angular velocity 2 in the clockwise direction, the outer one with angular velocity -2 in the anti-clockwise direction. Figure 5 shows the streamlines

DOI: 10.1002/fld

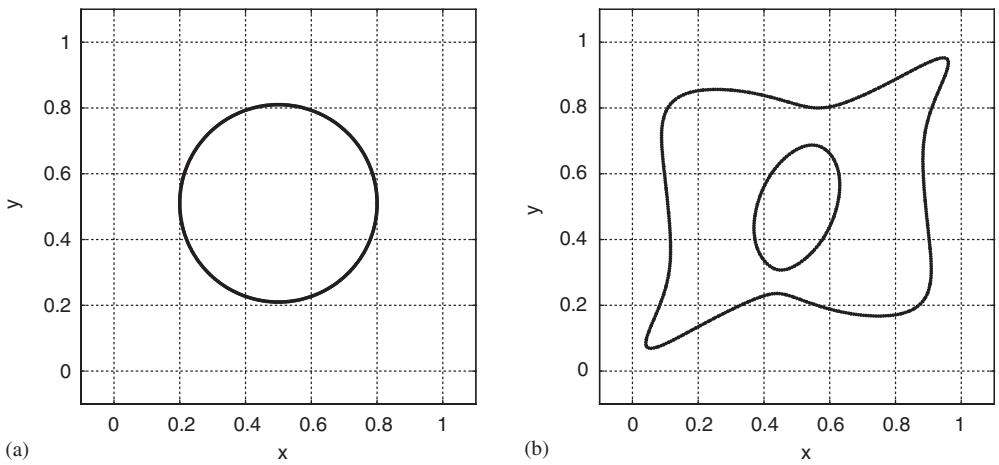

Figure 3. Geometries used for convergence studies in Example 3.1: (a) circle and (b) 'fish'.

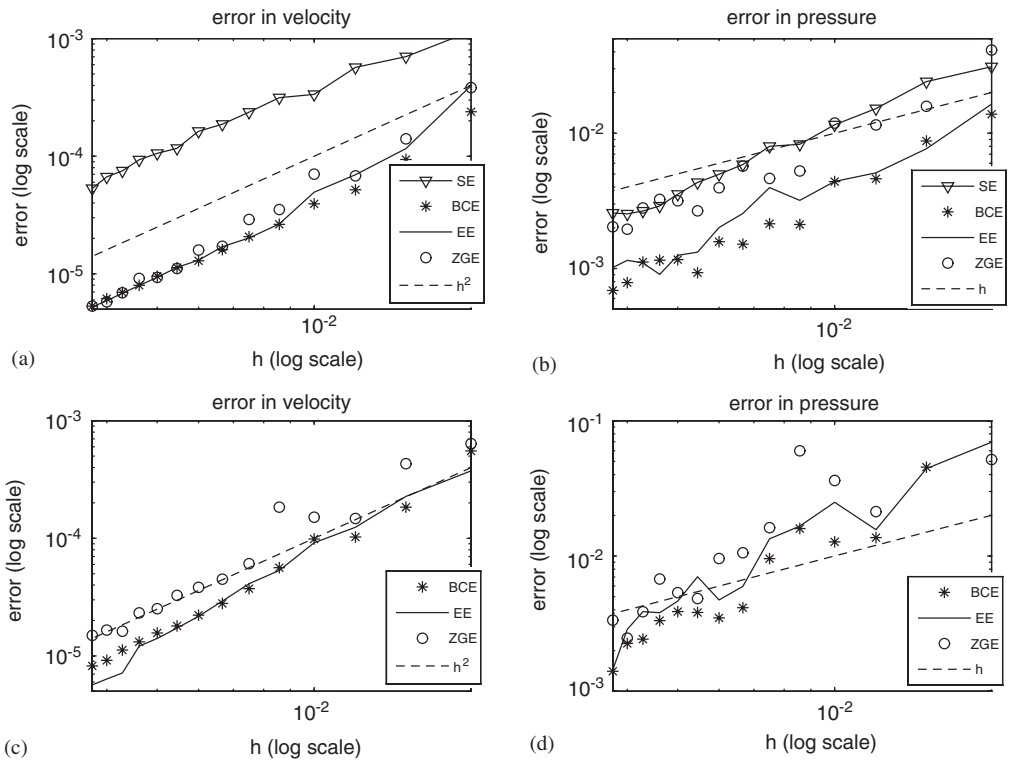

Figure 4. Convergence in Example 3.1 with different extensions in Ω^c: (a) circle geometry, velocity; (b) circle geometry, pressure; (c) 'fish' geometry, velocity; and (d) 'fish' geometry, pressure. Mesh widths $h \in \{\frac{1.2}{60}, \frac{1.2}{80}, \ldots, \frac{1.2}{320}\}$.

DOI: 10.1002/fld

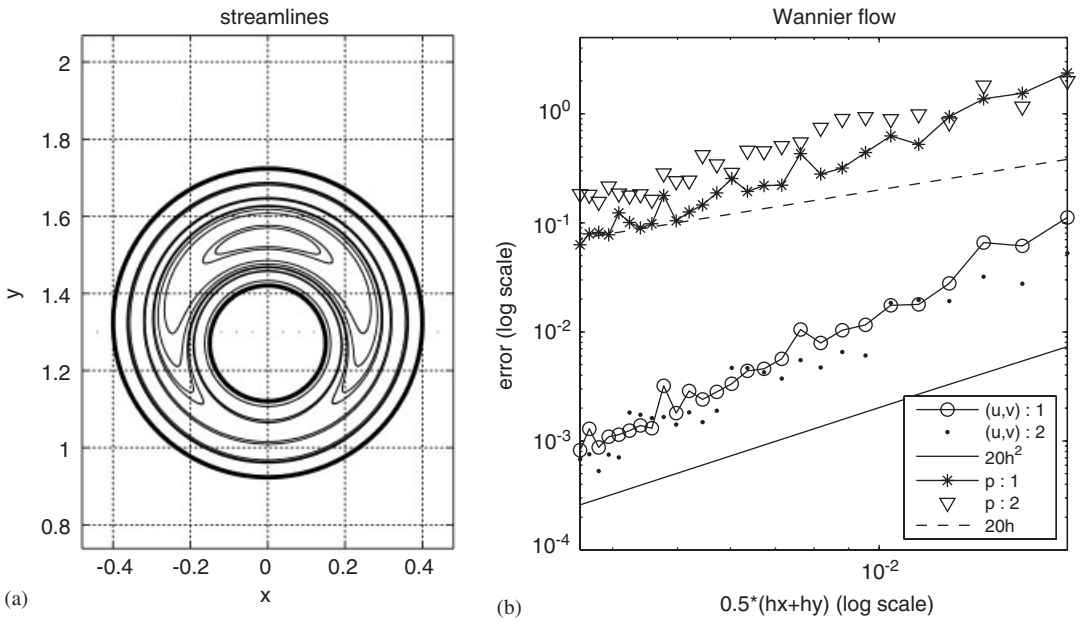

Figure 5. Wannier flow. In this case we take the same number n of grid points in the x and y directions, but the mesh widths are different $h_x \neq h_y$: (a) Geometry used in the convergence tests and streamlines of the numerical solution computed with $n=240$ grid points per direction. Note that the computational domain (marked in the figure) is not a square thus leading to different mesh widths in the x and y directions. (b) Convergence study using $n \in \{60, 70, \ldots, 320\}$ grid points per direction. $(u, v):1$, boundary condition extension, velocity component; (u, v): 2, exact extension, velocity component; $p:1$, boundary condition extension, pressure; $p:2$, exact extension, pressure. Note that the solution quality is the same for both types of approximation.

and convergence plot. The second-order convergence in velocity and first-order in pressure is observed.

Table II demonstrates how many GMRES iterations were needed to achieve a 10^{-8} tolerance when solving the Schur complement system (32). Summarizing the results in Figure 5 and Table II we see that also in this case the BCE has been an effective choice. The accuracy of the solution is the same as for the case with the EE but the cost is kept low by low iteration counts and the ability to use the FFT Poisson solver.

Example 3.3

The domain in this case is a bended tube with 11 circular inclusions (Figure 6). The inflow boundary is at the left top and the outflow boundary at the right top, where a parabolic inflow and outflow are set:

$$u_D = 0, \quad v_D = (x+3)(x+1) \text{ along the inflow boundary}$$

and

$$u_D = 0, \quad v_D = -(x-3)(x-1) \text{ along the outflow boundary}$$

DOI: 10.1002/fld

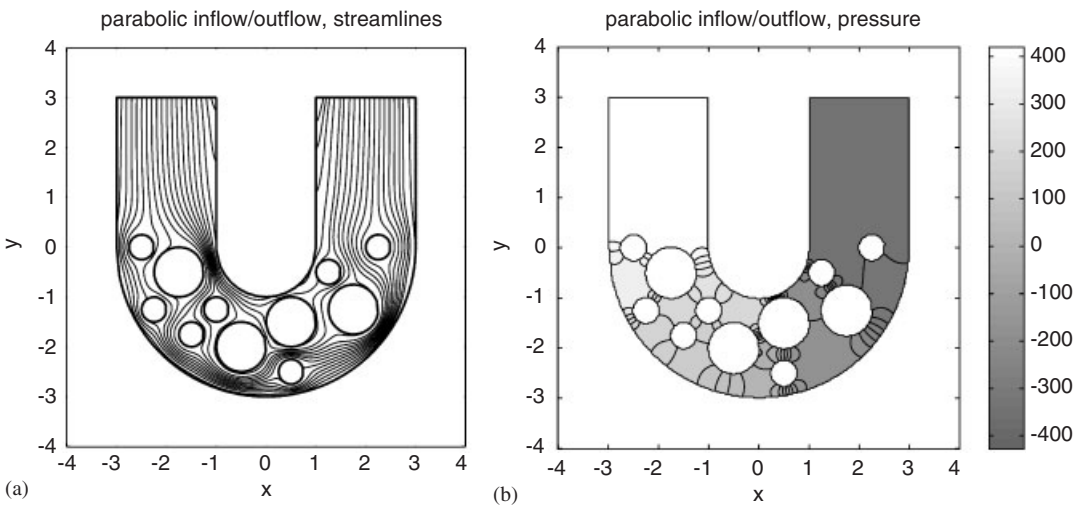

Figure 6. Numerical results in Example 3.3: (a) streamlines and (b) few level lines of the pressure with darker gray indicating lower values.

Everywhere else along the pipe boundary and the boundaries of the inclusions, we require the no-slip condition.

The computational costs using 320 grid points in each direction were 299 GMRES iterations to achieve 10^{-4} stopping tolerance, with a runtime of 112 min (Matlab 7 implementation on an Intel Pentium 4 CPU 2.80 GHz machine with 1 GB RAM). For a better quality of the solution, the boundary condition in the BCE approach has been weighted by a factor 40. The iteration count is high in this case; however, note that the geometry has several very thin structures, with only few grid points connecting different regions of Ω.

4. SUMMARY

In this paper we have applied the EJIIM to the stationary Stokes equations. Staggered grid finite differences form the basis of the new algorithm. The underlying fast Stokes solver is constructed using the pressure equation method together with an FFT-based Poisson solver. The expected convergence order (second order for velocity and first-order for pressure) has been confirmed by numerical experiments.

The BCE allows to overcome the problem of relatively high iteration counts, which EJIIM had in its standard version with ZGE. In addition, the solution quality is improved.

ACKNOWLEDGEMENTS

The author thanks A. Wiegmann, PhD (Fraunhofer ITWM Kaiserslautern) for the help in fixing the situations as in Remark 6. Also the hint of PD Dr O. Iliev (Fraunhofer ITWM Kaiserslautern) to Reference [22] turned out to be very useful. F. Brucker (Universität Konstanz) provided a code that was used to generate the geometry in Example 3.3.

DOI: 10.1002/fld

REFERENCES

1. Galdi GP. *An Introduction to the Mathematical Theory of the Navier–Stokes Equations*, *Volume I*: *Linearised Steady Problems* (2nd edn). Springer: Berlin, 1998.
2. Temam R. *Navier–Stokes Equations*: *Theory and Numerical Analysis*. North-Holland: Amsterdam, 1977.
3. Junk M, Illner R. A new derivation of Jeffery's equation. *Journal of Mathematical Fluid Mechanics* 2006; **8**:1–34.
4. Triebsch LK. A discrete boundary integral approach to 3D cell problems with complex geometry. *Master's Thesis*, University of Kaiserslautern, August 2001.
5. Wiegmann A. Computation of the permeability of porous materials from their microstructure by FFF-Stokes. *Technical Report 129*, Fraunhofer ITWM, Kaiserslautern, 2007.
6. LeVeque RJ, Li Z. The immersed interface method for elliptic equations with discontinuous coefficients and singular sources. *SIAM Journal on Numerical Analysis* 1994; **31**:1019–1044.
7. LeVeque RJ, Li Z. Immersed interface methods for Stokes flow with elastic boundaries or surface tension. *SIAM Journal on Scientific Computing* 1997; **18**(3):709–735.
8. Li Z, Lubkin SR. Numerical analysis of interfacial two-dimensional Stokes flow with discontinuous viscosity and variable surface tension. *International Journal for Numerical Methods in Fluids* 2001; **37**:525–540.
9. Li Z, Ito K, Lai M-C. An augmented approach for Stokes equations with a discontinuous viscosity and singular forces. *Technical Report CRSC-TR04-23*, North Carolina State University, 2004.
10. Li Z, Wan X, Ito K, Lubkin S. An augmented pressure boundary condition for a Stokes flow with a non-slip boundary condition. *Communications in Computational Physics* 2006; **1**:874–885.
11. Wiegmann A. The explicit-jump immersed interface method and interface problems for differential equations. *Ph.D. Thesis*, University of Washington, 1998.
12. Wiegmann A, Bube KP. The explicit-jump immersed interface method: finite difference methods for PDEs with piecewise smooth solutions. *SIAM Journal on Numerical Analysis* 2000; **37**(3):827–862.
13. Fletcher CAJ. *Computational Techniques for Fluid Dynamics*, *Volume II*, *Specific Techniques for Different Flow Categories*. Springer: Berlin, 1988.
14. Strikwerda JC. Finite difference methods for the Stokes and Navier–Stokes equations. *SIAM Journal on Scientific and Statistical Computing* 1984; **5**:56–68.
15. Shin D, Strikwerda JC. Inf-Sup conditions for finite difference approximations of the Stokes equations. *Journal of Australian Mathematical Society, Series B* 1997; **39**:121–134.
16. Shin D, Strikwerda JC. Fast solvers for finite difference approximations for the Stokes and Navier–Stokes equations. *Journal of Australian Mathematical Society, Series B* 1996; **38**:274–290.
17. Wiegmann A. Fast Poisson, fast Helmholtz and fast linear elastostatic solvers on rectangular parallelepipeds. *Technical Report LBNL-43565*, Lawrence Berkeley National Laboratory, Berkeley, CA, June 1999.
18. Iliash Y, Rossi T, Toivanen J. Two iterative methods for solving the Stokes problem. *Technical Report 2*, Laboratory of Scientific Computing, Department of Mathematics, University of Jyvaskyla, 1993.
19. Sethian JA, Wiegmann A. Structural boundary design via level set and explicit jump immersed interface methods. *Journal of Computational Physics* 2000; **163**(2):489–528.
20. Rutka V. Immersed interface methods for elliptic boundary value problems. *Ph.D. Thesis*, TU Kaiserslautern, 2005.
21. Li Z. The immersed interface method—a numerical approach for partial differential equations with interfaces. *Ph.D. Thesis*, University of Washington, 1994.
22. Iliev O, Tiwari S. A generalized (meshfree) finite difference discretization for elliptic interface problems. In *Lecture Notes in Computer Sciences*, vol. 2542/2003, Dimov I, Lirkov I, Margenov S, Zlatev Z (eds). Springer: Berlin, 2003; 488–497.
23. Kelley CT. *Iterative Methods for Linear and Nonlinear Equations*. SIAM: Philadelphia, PA, 1995.
24. Mayo A. The fast solution of Poisson's and the biharmonic equations on irregular regions. *SIAM Journal on Numerical Analysis* 1984; **21**(2):285–299.
25. Wiegmann A. The explicit jump immersed interface method and integral formulas. *Technical Report LBNL-43566*, Lawrence Berkeley National Laboratory, Berkeley, CA, June 1999.
26. Biros G, Ying L, Zorin D. A fast solver for Stokes equations with distributed forces in complex geometries. *Journal of Computational Physics* 2003; **193**:317–348.
27. Wannier GH. A contribution to the hydrodynamics of lubrication. *Quarterly of Applied Mathematics* 1950; **8**(1):1–32.
28. Calhoun D. A Cartesian grid method for solving the two-dimensional streamfunction–vorticity equations in irregular domains. *Journal of Computational Physics* 2002; **176**(2):231–275.

DOI: 10.1002/fld