Journal of Computational Physics 229 (2010) 4225–4246

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Handling solid–fluid interfaces for viscous flows: Explicit jump approximation vs. ghost cell approaches

Qinghai Zhang [a,*], Philip L.-F. Liu [b,c]

[a] Applied Numerical Algorithms Group, Lawrence Berkeley National Laboratory, Berkeley, CA 94720, USA
[b] School of Civil and Environmental Engineering, Cornell University, Ithaca, NY 14853, USA
[c] Institute of Hydrological and Oceanic Sciences, National Central University, Jhongli, Taiwan

ARTICLE INFO

Article history:
Received 18 February 2009
Received in revised form 7 February 2010
Accepted 8 February 2010
Available online 14 February 2010

Keywords:
Explicit jump approximation
Ghost cell approach
Viscous flow
Diffusion equation
Heat equation
Jump correction
The immersed interface method
Irregular solid boundary
The polygonal area mapping method
HyPAM

ABSTRACT

The ghost cell approaches (GCA) for handling stationary solid boundaries, regular or irregular, are first investigated theoretically and numerically for the diffusion equation with Dirichlet boundary conditions. The main conclusion of this part of investigation is that the approximation for the diffusion term has to be second-order accurate everywhere in order for the numerical solution to be rigorously second-order accurate. Violating this principle, the linear and quadratic GCAs have the following shortcomings: (1) restrictive constraints on grid size when the viscosity is small; (2) susceptibleness to instability of a time-explicit formulation for strongly transient flows; (3) convergence deterioration to zeroth- or first-order for solutions with high-frequency modes. Therefore, the widely-used linear extrapolation for enforcing no-slip boundary conditions should be avoided, even for regular solid boundaries. As a remedy, a simple method based on explicit jump approximation (EJA) is proposed. EJA hinges on the idea that a velocity-derivative jump at the boundary reduces to the value of the velocity-derivative at the fluid side because the velocity of the stationary boundary is zero. Although the time-marching linear system of EJA is not symmetric, it is strictly diagonal dominant with positive diagonal entries. Numerical results show that, over a large range of viscosity and grid sizes, EJA performs much better than GCAs in terms of stability and accuracy. Furthermore, the second-order convergence of EJA does not depend on viscosity and the spectrum of the solution, as those of GCAs do. This paper is written with enough details so that one can reproduce the numerical results.

1. Introduction

For any irregular domain with smooth boundaries, a smooth function can be extended across a boundary with a bound on the relative increase in the error norms. This is the essential idea behind the ghost cell approach (GCA). With the ghost cell values updated by extrapolating from inside the fluid phase, the boundary conditions at the solid–fluid interface are implicitly fulfilled. This approach dates back to Mayo [13] in 1980s and has been widely adopted by researchers; one group of examples [18,1] concerns the immersed boundary (IB) method with the finite difference formulation.

There exist numerous ways of obtaining the ghost cell values, most of them are variants of two formulas: the linear extrapolation via image points, hereafter referred to as GC1, and the quadratic extrapolation via polynomial fitting, hereafter referred to as GC2. Both GC1 and GC2 have been widely used in treating irregular boundaries, see [20,18] for two examples and [16] for a wide perspective.

* Corresponding author. Tel.: +1 510 4867473; fax: +1 510 4866900.
E-mail address: QHZhang@lbl.gov (Q. Zhang).

0021-9991/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2010.02.007

This paper considers the nonhomogeneous diffusion equation,

$$\frac{\partial u}{\partial t} = \nu \nabla^2 u + f(\mathbf{x}, t), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^{\mathbf{D}}$ is the location vector, t the temporal coordinate, $u = u(\mathbf{x}, t)$ a continuous scalar function with its value being a constant zero within the solid phase, ν the dynamic viscosity, and the forcing term $f(\mathbf{x}, t)$ is known *a priori*. In this and the next sections, we will focus on the one-dimensional version of (1):

$$\frac{\partial u}{\partial t} = \nu \frac{\partial^2 u}{\partial y^2} + f(y, t), \tag{2}$$

where y denotes the vertical coordinate. The solid–fluid interface is located at $y_B = bh$, with h being the uniform mesh spacing, as shown in Fig. 1. The regular boundary case in Fig. 1(a) can be considered as a special case of the irregular boundary case in Fig. 1(b) with $b = 0$. Without loss of generality, it is assumed that $b \in \left[-\frac{1}{2}, \frac{1}{2}\right)$ and the discretization of u is cell-centered. The value of u in the jth cell is represented by u_j, located at $y_j = (j + \frac{1}{2})h$.

Referring to Fig. 1(b), the image point of y_{-1} with respect to the interface is located at $y_{\text{image}} = (2b + \frac{1}{2})h$, where a linear interpolation yields $u_{\text{image}} = 2bu_1 + (1 - 2b)u_0$. GC1 sets the ghost cell value by

$$u_{-1}^{\text{GC1}} = 2u_B - u_{\text{image}}. \tag{3}$$

When $b = 0$ (i.e. regular boundary) and $u_B = 0$, (3) reduces to the well-known no-slip condition $u_{-1} = -u_0$ for regular boundaries, as shown in Fig. 1(a). Since (3) is linear, using a higher-order interpolation for u_{image} does not improve the overall accuracy unless more image points are introduced.

In GC2, the ghost cell value is evaluated by fitting a quadratic polynomial near the interface:

$$u_{-1}^{\text{GC2}} = \tilde{u}\left(-\frac{h}{2}\right), \tag{4}$$

$$\tilde{u}(y) = \frac{(y - \frac{3}{2}h)(y - \frac{5}{2}h)}{(b - \frac{3}{2})(b - \frac{5}{2})h^2} u_B + \frac{(y - \frac{5}{2}h)(y - bh)}{h^2(b - \frac{3}{2})} u_1 - \frac{(y - \frac{3}{2}h)(y - bh)}{h^2(b - \frac{5}{2})} u_2,$$

where the irregular cell value u_0 is excluded to prevent instabilities from $b \approx \frac{1}{2}$.

Taylor expansions of (3) and (4) at y_0 yield

$$\frac{u_{-1}^{\text{GC}k} + u_1 - 2u_0}{h^2} - \left.\frac{\partial^2 u}{\partial y^2}\right|_0 = T^{\text{GC}k} \left.\frac{\partial^{k+1} u}{\partial y^{k+1}}\right|_0 + O(h^k), \tag{5}$$

where

$$T^{\text{GC1}} = \frac{-1 - 8b + 4b^2}{4} \tag{6a}$$

$$T^{\text{GC2}} = \frac{1 + 2b}{2} h \tag{6b}$$

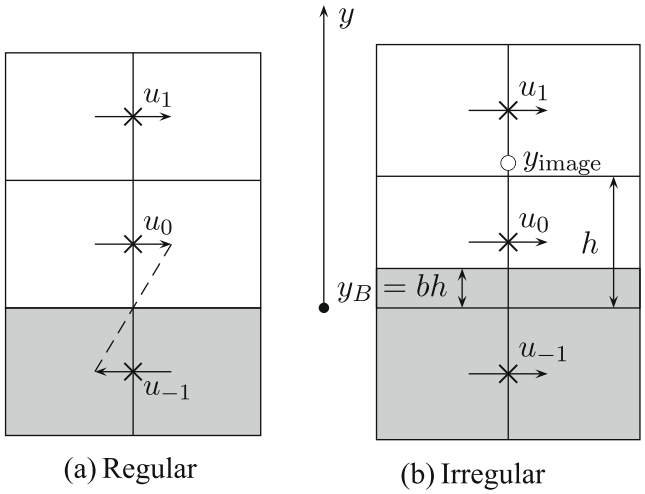


Fig. 1. Cells in the vicinity of the solid–fluid interface. Light gray area represents the solid phase and white area is occupied by the fluid phase. u_{-1} is a boundary condition to be specified. '$\circ$' represents the image point of y_{-1}.

It is clear from (5) that GCk introduces error of $O(h^{k-1})$ in approximating the diffusion term near the solid–fluid interface. For an explicit second-order method, one can expect that the errors near the solid–fluid interface propagate to the interior of the fluid phase by numerical diffusion and the accuracy is less than second-order, at least in the ∞-norm sense. However, explicit treatment of the diffusion term incurs a restrictive time step constraint $\Delta t \sim O(h^2)$ and the diffusion term is usually handled implicitly to avoid this constraint. This immediately raises a question on how a solution of (1) is influenced by perturbations of codimension one.

For Poisson's equation, it is well-known that the perturbations from GCA are negligible for solvers with second-order convergence. By examining the error equation for Poisson's equation, Johansen and Colella [6] showed that although GC2 generates a first-order truncation error on the boundary, the solution error induced is third-order; thus the second-order convergence rate is not influenced. McCorquodale et al. [14] also confirmed that both GC1 and GC2 are second-order accurate and even the solution gradient of GC2 is second-order accurate. As for the diffusion equation, Gibou and Fedkiw [4] claimed that a cubic extrapolation yields fourth-order accuracy of the solution, their supporting numerical tests only consisting of single Fourier modes with exponentially decaying amplitudes. However, we believe that effects of boundary perturbations on the diffusion equation is very different from those on Poisson's equation. As will be proved in Section 2, **a diffusion equation solver has to approximate the diffusion term to *k*th order accuracy everywhere inside the domain in order to be rigorously *k*th order accurate. This requires that the extrapolation near the boundary be (*k*+2)th order accurate.** Although sometimes a lower-order extrapolation may achieve super-convergence for solutions with a finite spectrum, this super-convergence does not apply to solutions with infinite spectrum, especially when viscosity is small.

In particular, GC1 and GC2 has asymptotic convergence of second- and third-order for solutions with low-frequency modes; but they are only zeroth and first-order accurate for some test cases in Section 4. Even when they do exhibit second-order convergence, their errors are much larger than those of the proposed explicit jump approximation (EJA) method in a wide range of grid sizes, as will be shown in Section 4.3. Furthermore, the asymptotic convergence of GCA requires the grid size to be smaller than a 'critical' grid size, which can be prohibitively small as viscosity is reduced. This coupling of convergence rate to viscosity is undesirable. In addition to the accuracy issues, instability of an explicit formulation of GCA is also reported in Section 4.2.

A straightforward way to improve the accuracy of GCA in one-dimensional space is to fit a cubic or even higher-order polynomial near the solid–fluid interface so that the diffusion term is uniformly approximated to second-order everywhere inside the fluid phase. However, in multi-dimensional spaces, extrapolations from different dimensions yield different values, as shown in Fig. 4. There are three ways to address this issue: (1) using a linear combination of these conflicting values [1]; (2) storing **D** ghost values at each ghost cell; (3) resort to multi-dimensional polynomial-based interpolation. In the last choice, the local qth order polynomial must have $\sum_{\ell=0}^{q} C_{D+\ell-1}^{\ell}$ terms including all the coordinates and their cross terms to preserve $(q-1)$th order-of-accuracy of the Laplacian term. For example, a fourth-order solver in three-dimensional space involves fitting a fifth-order polynomial, requiring solving at least a 56×56 linear system for each ghost cell value.

An alternative method to overcome the above two difficulties is to treat u as a function with discontinuities at the solid–fluid interface and incorporate the jump conditions into the discretization stencils. The immersed interface method (IIM) [10] is an example. Although the IIM was originally designed for elliptic problems, it has been extended to Stokes flows [11], Poisson's equation [3], parabolic equations [21,5], and incompressible Navier–Stokes equations [12,9]. Applications to rigid, flexible and moving boundaries [23,8] were also reported. One difficulty of the IIM, however, is the lack of jump conditions for high-order derivatives. To avoid this challenge, Zhong [29] and Zhou [30] have proposed their own high-order methods for solving elliptic equations by using two jump conditions for the value of the variable and its first-order derivative. Mathematically, these methods are equivalent to explicitly approximating high-order derivatives by either Taylor expansion or local polynomial fitting. However, in the case of a rigid solid phase, even the jump conditions on the first-order derivatives are unavailable.

Fortunately, the velocity and its derivatives for stationary solid–fluid interface are all zero. Thus, the velocity-derivative jumps reduce to the values of velocity-derivatives on the fluid side. This fact leads to the explicit jump approximation (EJA) method formulated in Section 3. Unlike the GCA, the extrapolation target of EJA is the interface, not the ghost cell. As will be shown in Fig. 4 in Section 3.2, the extrapolation of EJA is essentially one-dimensional because intersections of the interface to different directions are distinct. This permits a simple generalization to multi-dimensional spaces in a dimension-by-dimension way.

Similar to the IIM, EJA is based on the generalized Taylor expansion and discretization stencils are locally modified near the interface; unlike IIM, EJA approximates the jump conditions explicitly by extrapolation instead of deriving it from physical constraints. Also, the jumps in EJA are with respect to the coordinates, not with respect to the normal direction of the interface. These result in a simpler formulation without requiring jump condition derivation and local-global coordinate transformations. Roughly EJA can be viewed as an IIM specialized for stationary solid–fluid interfaces.

The rest of this paper is organized as follows. Section 2 answers the question on how a solution of (1) is influenced by perturbations near the solid–fluid interface, motivating the development of EJA in Section 3, where the time-marching linear system of EJA is shown to be strictly diagonally dominant. Section 4 evaluates the stability and accuracy of EJA and GCA via extensive tests in one-, two- and three-dimensional spaces. Finally, Section 5 outlines future work for coupling EJA to the polygonal area mapping method (PAM) [26] and HyPAM [28] to form a three-phase model including gas, fluid and solid.

2. An error analysis

Consider the one-dimensional homogeneous diffusion equation

$$\frac{\partial u}{\partial t} = \nu \frac{\partial^2 u}{\partial y^2}, \quad (u,t) \in [0,1] \times [0,\infty) \tag{7}$$

with initial and boundary conditions as

$$u(y,0) = \varphi(y), \quad u(0,t) = u(1,t) = 0. \tag{8}$$

Handling the solid–fluid interface at $y = 0$ by GCA adds into (7) a nonhomogeneous term that is zero everywhere except near the interface:

$$\frac{\partial u}{\partial t} = \nu \frac{\partial^2 u}{\partial y^2} + T(y,t), \tag{9}$$

where $T(y,t)$ can be deduced from (5) as

$$T(y,t) \approx T(t) = \begin{cases} \nu T^{\mathrm{GCk}}(t)\ u^{(k+1)}|_{\frac{h}{2}} & y \in [0,h] \\ 0 & y \in (h,1] \end{cases}, \tag{10}$$

with $u^{(k+1)}|_{\frac{h}{2}}$ as the value of the $(k+1)$th spatial partial derivative at $\frac{h}{2}$. $T(y,t)$ is assumed to be independent of y within the computational cell abutting the solid–fluid interface.

In the following we shall derive the exact solutions of the above homogeneous and nonhomogeneous problems; their difference yields the solution error caused by $T(y,t)$. The derivation can also be generalized to diffusion equations with Robin boundary conditions [2].

If x and t are separable, the solution of (7) and (8) can be derived as

$$u(y,t) = \sum_{m=1}^{\infty} \varphi_m e^{\lambda_m \nu t} \sin(m\pi y), \tag{11}$$

where $\lambda_m = -m^2\pi^2$ is the mth eigenvalue of the second-order spatial derivative and

$$\varphi_m = 2\int_0^1 \varphi(z) \sin(m\pi z)\ \mathrm{d}z. \tag{12}$$

To accommodate the nonhomogeneous term, we seek a solution of (9) in the form

$$u(x,t) = \sum_{m=1}^{\infty} u_m(t) \sin(m\pi y), \tag{13}$$

by setting

$$T_m(t) = 2\int_0^1 T(z,t) \sin(m\pi z)\ \mathrm{d}z = 2\nu T^{\mathrm{GCk}} u^{(k+1)}|_{\frac{h}{2}} \int_0^h \sin(m\pi z)\ \mathrm{d}z = 4\nu T^{\mathrm{GCk}} u^{(k+1)}|_{\frac{h}{2}} \frac{1}{m\pi} \sin^2\left(\frac{m\pi h}{2}\right), \tag{14}$$

so that

$$T(y,t) = \sum_{m=1}^{\infty} T_m(t) \sin(m\pi y). \tag{15}$$

Substitution of (15) and (13) into (9) yields a series of initial value problems, from which the solution of (9) can be obtained as

$$u(y,t) = \sum_{m=1}^{\infty} \varphi_m e^{\lambda_m \nu t} \sin(m\pi y) + \mathcal{E}, \tag{16}$$

where the contribution of the nonhomogeneous term is

$$\mathcal{E}(y,t) = \sum_{m=1}^{\infty} \left(\int_0^t e^{\lambda_m \nu (t-s)} T_m(s)\ \mathrm{d}s \right) \sin(m\pi y). \tag{17}$$

Since (11) and (16) are solutions to (7) and (9), respectively, (17) is indeed caused by the leading truncation error of GCA.

Substituting (14) into (17) yields

$$\mathcal{E}(y,t) = \sum_{m=1}^{\infty} \left(\beta_m^{\mathrm{GCk}} \frac{\sin(m\pi y)}{m\pi} \int_0^t \nu e^{\lambda_m \nu (t-s)} u^{(k+1)}|_{\frac{h}{2}}\ \mathrm{d}s \right), \tag{18}$$

where $\beta_m^{GCk} = O(h^{k-1})$. In particular, if $b = 0$ in (6), then

$$\beta_m^{GC1} = -\sin^2\left(\frac{m\pi h}{2}\right); \tag{19a}$$

$$\beta_m^{GC2} = 2h\sin^2\left(\frac{m\pi h}{2}\right). \tag{19b}$$

The integral in (18) represents the coupling of perturbations from GCA and the temporal error accumulation. Because $u^{(k+1)}|_{\frac{h}{2}}\,ds$ is independent of the wave number m, the summation with respect to m in general does not cancel. This nonlinear coupling of time and spatial Fourier modes accounts for the subtle difference between Poisson's equation and the diffusion equation on how the solution reacts to the perturbations at the boundary.

Based on (18) and (19), the following can also be deduced:

(I) For a single Fourier mode m, given y, t, GC1 and GC2 are asymptotically second-order and third-order accurate, respectively, i.e.,

$$h \to 0 \Rightarrow \begin{cases} \beta_m^{GC1} \sim O(h^2) \\ \beta_m^{GC2} \sim O(h^3) \end{cases} \Rightarrow \begin{cases} \mathcal{E}_m^{GC1} \sim O(h^2) \\ \mathcal{E}_m^{GC2} \sim O(h^3) \end{cases}. \tag{20}$$

(II) Let the solution (11) have a finite number of Fourier modes and $m_{\max}$ be the largest wave number. Define a critical grid size as

$$h_{crit} = \frac{2}{\pi m_{\max}}. \tag{21}$$

Within the range of grid sizes such that $h > h_{crit}$, GC1 and GC2 are only zeroth-order and first-order accurate, respectively. Depending on the spectrum of the solution, h_{crit} might be small and the range of lower-order convergence is then large.

(III) In practice, one often observes second-order convergence for GC1 and GC2 in the numerical tests of the diffusion equation because either the solution contains only low-frequency modes (e.g. [4]) or high-frequency modes are quickly damped out. The 'damping capacity' of the physical system relates to viscosity: the higher the viscosity, the faster high frequency modes are damped. When viscosity is very small, high-frequency errors persist due to slow damping. For GC1 and GC2, their second-order convergence rates thus depend on a large viscosity to damp out their low-order truncation errors, see also the argument after (38) and (39) for a precise explanation. As another way to understand this, let $t' = \nu t$, (7) can be viewed as another heat equation with a changed time variable and unit viscosity; a small ν effectively changes a transient solution in t' to a steady solution in t.

(IV) Due to the presence of the forcing term $f(y,t)$, the nonhomogeneous diffusion Eq. (1) might have high-frequency modes persisting (or even increasing) over time. Examples include the pressure-driven steady flow and impinging jet on a solid–fluid interface, in both scenarios the pressure gradient and the advection term serve as the combined forcing term. Therefore, coupling convergence rate to viscosity has a negative impact on the solution accuracy of many practical applications because high-frequency modes do not *always* die out.

(V) If the diffusion term is approximated to second-order near the boundary, then $\mathcal{E} \sim O\left(h^2\sin^2\left(\frac{m\pi h}{2}\right)\right)$ and the second-order convergence rate does not depend on the spectrum of the solution. This is the case with cubic GCA and EJA.

Numerical experiments in Section 4 will confirm these remarks. Together with the instability of time-explicit GCA shown in Section 4.2, they motivate EJA formulated in the next section.

3. The explicit jump approximation method

In this section, the EJA method is fully detailed for the nonhomogeneous diffusion Eq. (1) with irregular solid–fluid interfaces. Within its framework, simply changing the jump correction form also implements GCA. It is important to point out that many projection methods for the Navier–Stokes equations use this procedure to obtain intermediate results. Thus, the algorithm described here is not only pertinent to the diffusion equation, but also applicable to a wider context.

3.1. Mathematical foundation

Like the IIM, the theoretical foundation of the EJA method is the generalized Taylor expansion, which has already been proved in the literature [22,24].

Theorem 1 (Generalized Taylor expansion). *Assume function $g(y)$ has m discontinuity points of the first kind at $y_1, y_2, \ldots, y_m$ in (y_0, y_{m+1}), $y_0 < y_1 < \cdots < y_{m+1}$ and $g(y) \in C^\infty(y_0, y_1) \cup (y_1, y_2) \cup \cdots \cup (y_m, y_{m+1})$, as shown in Fig. 2. $g(y)$ can be either continuous or discontinuous at y_0 and y_{m+1}. Let*

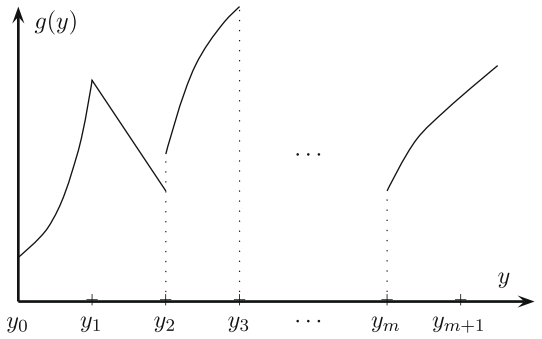


Fig. 2. A function $g(y)$ with finite discontinuities. $g(y)$ is continuous at y_1 while its derivatives are not. In contrast, $g(x)$ and all its derivatives are discontinuous at y_2.

$$\left[\left[g^{(n)}(y_i)\right]\right] = g^{(n)}(y_i^+) - g^{(n)}(y_i^-) \tag{22}$$

denote the jump of the nth derivative where $n = 0, 1, 2, \ldots$ *and* $i = 1, 2, \ldots, m$, *then*

$$g(y_{m+1}^-) = \sum_{n=0}^{\infty} \frac{g^{(n)}(y_0^+)}{n!}(y_{m+1} - y_0)^n + \sum_{i=1}^{m}\sum_{n=0}^{\infty} \frac{\left[\left[g^{(n)}(y_i)\right]\right]}{n!}(y_{m+1} - y_i)^n, \tag{23a}$$

$$g(y_0^+) = \sum_{n=0}^{\infty} \frac{g^{(n)}(y_{m+1}^-)}{n!}(y_0 - y_{m+1})^n - \sum_{i=1}^{m}\sum_{n=0}^{\infty} \frac{\left[\left[g^{(n)}(y_i)\right]\right]}{n!}(y_0 - y_i)^n. \tag{23b}$$

Theorem 1 gives rise to a generalized finite differencing that can be applied to functions with discontinuities.

Proposition 2 (Generalized central finite difference). *Let* $y_{j+1} - y_j = y_j - y_{j-1} = h > 0$ *and* $y_{j-1} < \alpha < y_j \leqslant \beta < y_{j+1}$. $u(y) \in C^\infty(y_{j-1}, \alpha) \cup (\alpha, \beta) \cup (\beta, y_{j+1})$. *Then*

$$\begin{aligned}\frac{\mathrm{d}\, u\left(y_j^-\right)}{\mathrm{d}\, y} &= \frac{u\left(y_{j+1}^-\right) - u\left(y_{j-1}^+\right)}{2h} \\ &\quad - \frac{1}{2h}\left(\sum_{n=0}^{2} \frac{\left[\left[u^{(n)}(\alpha)\right]\right]}{n!}(y_{j-1} - \alpha)^n + \sum_{n=0}^{2} \frac{\left[\left[u^{(n)}(\beta)\right]\right]}{n!}\left(y_{j+1} - \beta\right)^n\right) + O(h^2)\end{aligned} \tag{24}$$

$$\begin{aligned}\frac{\mathrm{d}^2 u(y_j^-)}{\mathrm{d}y^2} &= \frac{u\left(y_{j+1}^-\right) + u\left(y_{j-1}^+\right) - 2u(y_j)}{h^2} \\ &\quad + \frac{1}{h^2}\left(\sum_{n=0}^{3} \frac{\left[\left[u^{(n)}(\alpha)\right]\right]}{n!}(y_{j-1} - \alpha)^n - \sum_{n=0}^{3} \frac{\left[\left[u^{(n)}(\beta)\right]\right]}{n!}(y_{j+1} - \beta)^n\right) + O(h^2)\end{aligned} \tag{25}$$

Proof. Using (23) to expand $u(y_{j+1})$ and $u(y_{j-1})$ at y_j^- yields (24) and (25). □

According to Proposition 2, when there is only one jump between y_{j-1} and y_{j+1}, the location of the jump has to be compared with y_j in order to select the proper form of a velocity derivative jump. Thus, for the configuration in Fig. 1(b), (25) reduces to

$$\left.\frac{\partial^2 u}{\partial y^2}\right|_0 = \frac{u_1 + u_{-1} - 2u_0}{h^2} + \frac{1}{h^2}\left(\sum_{n=0}^{3} \frac{\left[\left[u^{(n)}(y_B)\right]\right]}{n!}(y_{-1} - y_B)^n\right) + O(h^2), \tag{26}$$

where y_B is the location of the interface between y_{-1} and y_0, assuming no interface exists between y_0 and y_1.

Identifying u as velocity for physical variable, the continuity of u implies

$$[[u]]_B = 0 \Rightarrow u(y_B^+) = u(y_B^-) = u_B,$$

where the speed of the solid phase, u_B, is assumed to be identically zero, which further implies that the velocity-derivatives of the solid phase are zero. Hence, for $n \geqslant 1$,

$$\left[\left[u^{(n)}\right]\right]_B = u^{(n)}(y_B^+) - u^{(n)}(y_B^-) = \begin{cases} u^{(n)}(y_B^+), & y_B^+ \text{ is in fluid} \\ -u^{(n)}(y_B^-), & y_B^- \text{ is in fluid} \end{cases}.$$

In other words, jumps of the derivatives are reduced to the derivatives at the fluid side, due to the assumption of u being zero in the solid phase and continuous in the whole domain.

In Appendix A the jumps of derivatives in (25) are approximated so that the second-order accuracy is preserved. Utilizing the results there, (25) can be written as

$$h^2 \frac{\mathrm{d}^2 u(y_j^-)}{\mathrm{d}y^2} = u_{j+1} + u_{j-1} - 2u_j + V^{\mathrm{EJA}}(u_j, u_{j+\mathsf{s}}, u_{j+2\mathsf{s}})^T + O(h^4), \tag{27}$$

where $\mathsf{s} = 1$ if the solid–fluid interface lies at the negative side of the fluid; otherwise $\mathsf{s} = -1$.

$$V^{\mathrm{EJA}}(\epsilon) = (\epsilon - 1)\left(\frac{3}{\epsilon},\ -\frac{3}{\epsilon+1},\ \frac{1}{\epsilon+2}\right), \tag{28}$$

where ϵ is the normalized distance from the cell center to the solid–fluid interface, $\epsilon = \frac{1}{2} - b$ in Fig. 1 (b). In the next section, these definitions are generalized to multi-dimensional space as in (33) and (34) with $\epsilon = \min\left(\epsilon^+_{\mathsf{i},d}, \epsilon^-_{\mathsf{i},d}\right)$.

Approximation of the second-order derivative by GCA can be put into similar forms:

$$h^2 \frac{\mathrm{d}^2 u(y_j^-)}{\mathrm{d}y^2} = u_{j+1} + u_{j-1} - 2u_j + V^{GCk}(u_j, u_{j+\mathsf{s}}, u_{j+2\mathsf{s}})^T + O(h^{k+1}), \tag{29}$$

where

$$V^{\mathrm{GC1}}(\epsilon) = (-2\epsilon,\ 2\epsilon - 1,\ 0); \tag{30a}$$

$$V^{\mathrm{GC2}}(\epsilon) = \left(0,\ 3\frac{\epsilon-1}{\epsilon+1},\ -2\frac{\epsilon-1}{\epsilon+2}\right). \tag{30b}$$

3.2. Laplacian discretization

Hereafter we expand our notation to **D**-dimensional space. For exposition convenience, the computational domain is assumed to have N cells in each dimension with uniform grid size h. A cell is identified by a multi-index $\mathsf{i} = (i_0, i_1, \ldots, i_{\mathbf{D}-1}) \in \mathbb{Z}^{\mathbf{D}}$. A bijective mapping $M : [0, N-1]^{\mathbf{D}} \to [0, N^{\mathbf{D}} - 1]$ sends a multi-index to a scalar index:

$$M(\mathsf{i}) = \sum_{d=0}^{\mathbf{D}-1} N^d\ i_d. \tag{31}$$

The stencil to discretize the Laplacian operator is a set of $2\mathbf{D} + 1$ multi-indices:

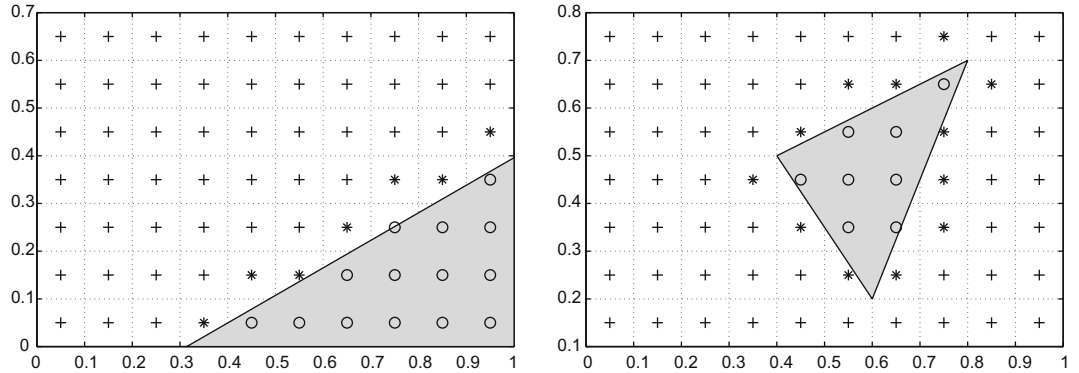
$$\mathcal{S}(\mathsf{i}) = \left\{\mathsf{i} + a\mathbf{e}^d : a = -1, 0, 1;\ d = 0, 1, \ldots, \mathbf{D} - 1\right\}, \tag{32}$$

where the dth component of $\mathbf{e}^d$ is one and all other components are 0.

Let $\mathcal{B}$ denote the compact set representing the solid phase, and $\mathbf{x}_\mathsf{i} = (\mathsf{i} + \frac{1}{2})h$ the center of cell i, cell i is

- a *fluid cell* iff $\forall \mathsf{j} \in \mathcal{S}(\mathsf{i}), \mathbf{x}_\mathsf{j} \notin \mathcal{B}$;
- an *interface cell* iff $\mathbf{x}_\mathsf{i} \notin \mathcal{B}$, but $\exists \mathsf{j} \in \mathcal{S}(\mathsf{i}), \mathbf{x}_\mathsf{j} \in \mathcal{B}$;
- a *solid cell* iff $\mathbf{x}_\mathsf{i} \in \mathcal{B}$.

This classification is illustrated in Fig. 3. Discretizing the Laplacian operator in the interface cells has to incorporate the jump conditions, which necessitate the definition of a pair of vectors $\boldsymbol{\epsilon}^\pm_\mathsf{i}$, whose dth components are the normalized distance between the cell center and the boundary of $\mathcal{B}$ along the dth axis:



(a) a slope as the irregular boundary. (b) a triangle solid inside the domain.

Fig. 3. Cell classification. Shaded area represents the solid phase. A solid cell marked by '○' has its center inside the solid; an interface cell marked by '∗' is close to the solid–fluid interface; a fluid cell marked by '+' does not need jump correction. To capture the geometry, each interface cell also has a labeling vector s_i indicating the orientation of the nearby solid and another pair of vectors $\boldsymbol{\epsilon}^\pm_\mathsf{i}$ recording the normalized distance from its center to the interface, as shown in Fig. 4.

$$\epsilon^{\pm}_{\mathrm{i},d} = \begin{cases} +\infty, & \text{if } \{\mathbf{x}_{\mathrm{i}} \pm ch\mathbf{e}^d : c \in \mathbb{R}^+\} \cap \mathcal{B} = \emptyset \\ \inf\{c \in \mathbb{R}^+ : \mathbf{x}_{\mathrm{i}} \pm ch\mathbf{e}^d \in \mathcal{B}\}, & \text{otherwise} \end{cases}, \tag{33}$$

and a sign vector s_{i} whose dth component is

$$\mathrm{s}_{\mathrm{i},d} = \begin{cases} 1, & \text{if } \epsilon^-_{\mathrm{i},d} \leqslant \epsilon^+_{\mathrm{i},d} \\ -1, & \text{if } \epsilon^+_{\mathrm{i},d} < \epsilon^-_{\mathrm{i},d} \end{cases}. \tag{34}$$

As shown in Fig. 4, $\mathrm{s}_{\mathrm{i},d}$ indicates the relative orientation of the center of cell i and the nearby solid along the dth axis while $\epsilon^{\pm}_{\mathrm{i},d}$ stores the normalized distances. For example, cell i in Fig. 4 has $\epsilon^+_{\mathrm{i},1} = +\infty$, $\epsilon^-_{\mathrm{i},1} < 1$, and $s_{\mathrm{i},1} = 1$; similarly, cell i' has $\epsilon^+_{\mathrm{i}',0} = +\infty$, $\epsilon^-_{\mathrm{i}',0} < 1$, and $s_{\mathrm{i}',0} = 1$.

Starting from the geometry information as above, the Laplacian operator can be discretized as

$$h^2\nabla^2 U = \mathbf{A}^L U + \Phi + O(h^4), \tag{35}$$

where $\mathbf{A}^L \in \mathbb{R}^{N^{\mathbf{D}} \times N^{\mathbf{D}}}$ includes the jump corrections and $\Phi \in \mathbb{R}^{N^{\mathbf{D}} \times 1}$ contains the domain boundary conditions. $U \in \mathbb{R}^{N^{\mathbf{D}}}$ is the unknown vector of u at cell centers ordered by (31).

Algorithm 1. Discretizing Laplacian operator by the EJA method. V^{EJA} is defined in (28). Alternatively, one can replace V^{EJA} with V^{GCk} in (30) for a GCA discretization. $\mathbf{A}^L_{m_1,m_2}$ refers to the element of $\mathbf{A}^L$ at the m_1th row and m_2th column. Similarly, Φ_{m_1} refers to the m_1th element of Φ.

Input: $\mathscr{B}$: a **D**-dimensional compact set representing the solid phase;
N: number of cells in each dimension;
h: grid size;
$\epsilon_{\min}$: a small number for controlling the condition number of $\mathbf{A}^L$;
M: a bijective mapping from multi-indices to scalar indices;
$u_{\partial\Omega}(\mathbf{x})$: boundary condition at the *domain boundary*;

Output: $\mathbf{A}^L$ and Φ : $h^2\nabla^2 U = \mathbf{A}^L U + \Phi + O(h^4)$.

$\mathbf{A}^L \leftarrow \mathbf{0}^{N^{\mathbf{D}} \times N^{\mathbf{D}}}$, $\Phi \leftarrow \mathbf{0}^{N^{\mathbf{D}} \times 1}$
foreach cell i inside the domain **do**
  if $\mathbf{x}_{\mathrm{i}} = (\mathrm{i} + \frac{1}{2})h \notin \mathscr{B}$ **then**
    compute $\epsilon^{\pm}_{\mathrm{i}}$ and s_{i} using (33) and (34)
    if $\min_{\pm,d}\epsilon^{\pm}_{\mathrm{i},d} < \epsilon_{\min}$ **then**
      continue
    end
    for $d = 0, 1, \ldots, \mathbf{D} - 1$ **do**
      $A_{M(\mathrm{i}),M(\mathrm{i})} \leftarrow A_{M(\mathrm{i}),M(\mathrm{i})} - 2$
      foreach $\mathrm{j} = \mathrm{i} \pm \mathbf{e}^d$ **do**
        if j *is inside the domain* **then**
          $A_{M(\mathrm{i}),M(\mathrm{j})} \leftarrow A_{M(\mathrm{i}),M(\mathrm{j})} + 1$
        else
          $\Phi_{M(\mathrm{j})} \leftarrow \Phi_{M(\mathrm{j})} + u_{\partial\Omega}(\mathbf{x}_{\mathrm{j}})$
        end
      end
      $\epsilon \leftarrow \min\left(\epsilon^+_{\mathrm{i},d}, \epsilon^-_{\mathrm{i},d}\right)$
      if $\epsilon < 1$ **and** $\max\left(\epsilon^+_{\mathrm{i},d}, \epsilon^-_{\mathrm{i},d}\right) \geqslant 2$ **then**
        $$\begin{pmatrix} \mathbf{A}^L_{M(\mathrm{i}),M(\mathrm{i})} \\ \mathbf{A}^L_{M(\mathrm{i}),M(\mathrm{i}+\mathrm{s}_{\mathrm{i},d}\mathbf{e}^d)} \\ \mathbf{A}^L_{M(\mathrm{i}),M(\mathrm{i}+2\mathrm{s}_{\mathrm{i},d}\mathbf{e}^d)} \end{pmatrix}^T \leftarrow \begin{pmatrix} \mathbf{A}^L_{M(\mathrm{i}),M(\mathrm{i})} \\ \mathbf{A}^L_{M(\mathrm{i}),M(\mathrm{i}+\mathrm{s}_{\mathrm{i},d}\mathbf{e}^d)} \\ \mathbf{A}^L_{M(\mathrm{i}),M(\mathrm{i}+2\mathrm{s}_{\mathrm{i},d}\mathbf{e}^d)} \end{pmatrix}^T + V^{\mathrm{EJA}}(\epsilon)$$
      else if $\epsilon < 1$ **then**
        use a smaller stencil out of the available data
      end
    end
  end
end

Because of the absence of cross differentiation terms, the jump corrections can be applied dimension by dimension. Algorithm 1 formalizes this idea by assembling $\mathbf{A}^L$ and Φ from the geometrical information and boundary conditions. In this algo-

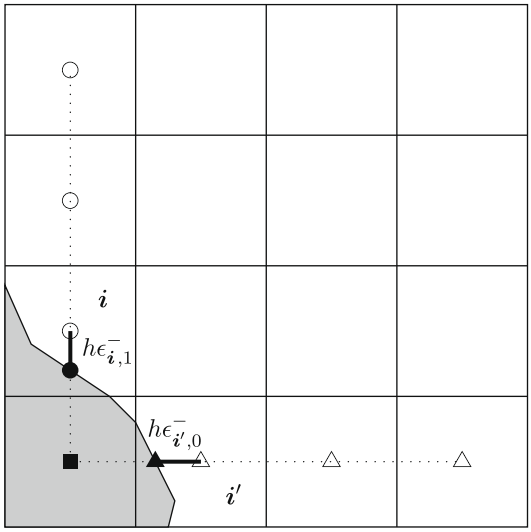


Fig. 4. Capturing geometry of interface cells for Laplacian discretization. The thick solid line segments represent the distances between the cell center and the interface along a certain axis in cell i and i′. Intersecting the solid–fluid interface to dotted lines yield the sites of extrapolation targets, represented by '•'s and '▲'s. '○'s and '△'s represent physical quantities involved in approximating $\left[\left[\frac{\partial^n u}{\partial y^n}\right]\right]$ and $\left[\left[\frac{\partial^n u}{\partial x^n}\right]\right]$ at these extrapolation targets, respectively. The fact of '•'s and '▲'s being distinct enables EJA to be locally one-dimensional. In contrast, one-dimensional GCA extrapolation will yield conflicting values at the location represented by '■'.

rithm, each cell in the domain corresponds to one row in $\mathbf{A}^L$. Lines 17–22 add the jump correction (28) or (30) into the standard Laplacian discretization in lines 9–16. Lines 17–22 operate only on interface cells while lines 9–16 operate on both interface cells and fluid cells. As for the solid cells, their corresponding rows are left as zeros to reduce the number of elements to be stored in the sparse matrix $\mathbf{A}^L$, which is justified by the assumption that the value of the unknown and its Laplacian is identically zero in the solid phase.

Proposition 3. *For an irregular solid–fluid interface, the matrix* $\mathbf{A}^L$ *produced by EJA in Algorithm 1 is not symmetric but diagonally dominant.*

Proof. The asymmetry of $\mathbf{A}^L$ follows from the one-sidedness of the jump correction form (28). In Algorithm 1, a solid cell results in a zero row and a fluid cell a row with $-2\mathbf{D}$ on the diagonal and $2\mathbf{D}$ 1's on other non-diagonal columns. As for an interface cell, $\epsilon \in (0,1)$ for some d. Since the jump corrections are applied dimension by dimension, it suffices to show that the coefficient of u_j in (28) is dominant:

$$\left|2+\frac{3(1-\epsilon)}{\epsilon}\right| > \left|1+\frac{3(1-\epsilon)}{\epsilon+1}\right| + \left|\frac{1-\epsilon}{2+\epsilon}\right| + 1,$$

which simplifies to $(1-\epsilon)(6+2\epsilon-\epsilon^2) > 0$. □

When $\epsilon \approx 0$, the first jump correction term is very large and consequently $\mathbf{A}^L$ might have a very large condition number. One possible solution is to shift the jump corrections away from the boundary to avoid large condition number of $\mathbf{A}^L$. However, this method destroys the diagonal dominance of $\mathbf{A}^L$ and increases the width of the jump correction stencil. Numerical experiments such as the 2-D sphere test in Section 4 show that this method can reduce the convergence rate of EJA by up to 0.8. Consequently, it is not adopted in Algorithm 1; instead, if the distance from an interface cell center to the solid–fluid interface is less than $\epsilon_{\min}h$, we simply treat the interface cell as a solid cell, as shown in lines 5–7 in Algorithm 1. Using $\epsilon_{\min} = 10^{-4}$, the second-order convergence rate of EJA is not influenced for all tests in Section 4.

Algorithm 1 is general in the sense that replacing V^{EJA} with V^{GCK} recovers a *time-implicit* GCA discretization that stores multiple ghost values in a single ghost cell. Since this yields the best accuracy of GCA and encourages module reuse, the results of GCA in Section 4 are also obtained through Algorithm 1.

To preserve the symmetry of $\mathbf{A}^L$, another type of *time-explicit* GCA formulation uses the standard Laplacian discretization without jump corrections and incorporates the effect of solid phase into the discretization by adding to Φ the ghost cell values $u_{-1}^{GCk}(t_{n+1})$ as defined in either (3) or (4). However, as is obvious in Section 3.3, when the integral effect of Φ is approximated, $u_{-1}^{GCk}(t_{n+1})$, the ghost value at the end of the time step, is needed whereas this formulation can only give $u_{-1}^{GCk}(t_n)$, the ghost value at the beginning of the time step, because the solution at t_{n+1} is not available for extrapolation yet. An ad hoc workaround sets

$$u_{-1}^{GCk}(t_{n+1}) = u_{-1}^{GCk}(t_n). \tag{36}$$

We shall denote this time-explicit formulation of GCA by 'GCkE' to distinguish it from the time-implicit formulation of GCA defined by (29) and (30). At first sight the symmetry of $\mathbf{A}^L$ is attractive because the time-marching matrices $\mathbf{B}(r_1)$ and $\mathbf{B}(r_2)$ in (44) are symmetric positive-definite. However, as will be demonstrated in Section 4.2, this formulation is susceptible to instabilities, especially in the GC1E case.

3.3. Time integration

Algorithm 1 transforms (1) into an ODE system as

$$\frac{\mathrm{d}U(t)}{\mathrm{d}t} = \mathbf{A}U(t) + \Psi(t), \tag{37}$$

where $\mathbf{A} = \frac{\nu}{h^2}\mathbf{A}_L$ and $\Psi(t) = f(t) + \frac{\nu}{h^2}\Phi(t)$. It is well-known that (37) satisfies the following recurrence relation

$$U(t_{n+1}) = \exp(\Delta t\mathbf{A})U(t_n) + \int_{t_n}^{t_{n+1}} \exp\left((t_{n+1}-s)\mathbf{A}\right)\Psi(s)\ \mathrm{d}s. \tag{38}$$

For the standard $3^{\mathbf{D}}$ Laplacian stencil, the eigenvalues of $\Delta t\mathbf{A}$ are

$$\lambda_{\Delta t\mathbf{A}} = -\frac{\nu\Delta t}{h^2}\sum_{d=0}^{\mathbf{D}-1}\sin^2\frac{\zeta_d}{2}, \tag{39}$$

where $\zeta_d \in (0,\pi)$. In the asymptotic range of $h \to 0$, $\lambda_{\Delta t\mathbf{A}} \ll 0$, and the truncation error of discretizing the diffusion term are damped very quickly through $\exp(\Delta t\mathbf{A})$ in (38). However, when ν is very small, so are the absolute values of $\lambda_{\Delta t\mathbf{A}}$. Consequently, the damping of truncation errors takes a much longer time. This argument on the difference system (37) formalizes remark (III) in Section 2.

For time integration, a family of widely-used methods is based on the Padé approximation of the exponential function in the RHS of (38). The well-known Crank–Nicolson method (C–N) is such an example. However, since C–N has a symbol that tends asymptotically to -1, it is only neutrally stable and the numerical solution exhibits oscillatory behaviors for discontinuities in initial and boundary conditions [7]. McCorquodale et al. [15] showed the instability of C–N by a moving boundary calculation. Section 4.2 shows that C–N coupled with GCkE is not stable for a strongly transient flow. One reason of this instability is that C–N does not effectively damp out the errors when the grid size is small.

In contrast, the method proposed by Twizell, Gumel, and Arigu (TGA) [19] is L_0-stable, i.e. its symbol tends asymptotically to zero. It is based on a (2,1) Padé approximation of the exponential function

$$\exp(\Delta t\mathbf{A}) = \mathbf{B}^{-1}(r_1)\mathbf{B}^{-1}(r_2)\mathbf{B}(\alpha-1) + O(\Delta t^3), \tag{40}$$

where

$$\mathbf{B}(\chi) = I - \chi\Delta t\mathbf{A}, \tag{41}$$

$$r_1 = \frac{\alpha - (\alpha^2-4\alpha+2)^{1/2}}{2}, \quad r_2 = \frac{\alpha + (\alpha^2-4\alpha+2)^{1/2}}{2}, \tag{42}$$

and α is in the range of $\left(\frac{1}{2}, 2-\sqrt{2}\right)$. This choice of α ensures second-order accuracy, L_0 stability and the use of only real arithmetic. In practice, α is chosen to be as large as possible to minimize the truncation error. In this work, $\alpha = 0.58$ is used for TGA. Note that setting $\alpha = 0.5$ reduces TGA to C–N. In this work, C–N is only used in Section 4.2 to demonstrate the instability of time-explicit GCA; elsewhere TGA is always used.

Approximating the integral in the RHS of (38) by a trapezoidal rule [17,19]:

$$\int_{t_n}^{t_{n+1}} \exp\left((t_{n+1}-s)\mathbf{A}\right)\Psi(s)\ \mathrm{d}s = \frac{\Delta t}{2}\mathbf{B}^{-1}(r_1)\mathbf{B}^{-1}(r_2)(\Psi(t_n) + \mathbf{B}(2\alpha-1)\Psi(t_{n+1})) + O(\Delta t^3), \tag{43}$$

the ODE (37) can be solved by

$$\mathbf{B}(r_2)\mathbf{B}(r_1)U^{n+1} = \mathbf{B}(\alpha-1)U^n + \frac{\Delta t}{2}(\Psi(t_n) + \mathbf{B}(2\alpha-1)\Psi(t_{n+1})). \tag{44}$$

At each time step, advancing (37) requires solving two linear systems to ensure L_0 stability. It is clear from (38), (40), and (43) that TGA is second-order accurate in time.

Proposition 4. *For EJA, $\mathbf{B}(\chi)$ with $\chi \geqslant 0$ is strictly diagonally dominant for both regular and irregular solid–fluid interfaces.*

Proof. The regular case holds because of $\chi \geqslant 0$, $\nu > 0$, the definition (41), and the fact that $\mathbf{A}^L$ becomes the standard discretization with the diagonal entries as $-2\mathbf{D}$. The irregular case follows from Proposition 3. □

By the Gershgorin circle theorem, a strictly diagonally dominant matrix is non-singular. Furthermore, since all the diagonal entries of $\mathbf{B}(r_1)$ and $\mathbf{B}(r_2)$ are positive, the real parts of the eigenvalues of them are non-negative.

3.4. Solution procedure

In summary, EJA solves the diffusion Eq. (1) with *stationary* solid–fluid interface as follows:

Step 1. Assemble $\mathbf{A}^L$ by Algorithm 1 from the geometry of the solid phase.
Step 2. Set initial condition $U(t_0)$.
Step 3. Compute $\Phi(t)$, $f(t)$ to evaluate the RHS of (44).
Step 4. Solve the two linear systems in (44) to advance the solution.
Step 5. Repeat Step 3 and Step 4 until the final time t_e is reached.

The solid–fluid interface being stationary implies that $\mathbf{A}^L$ is time-independent and hence should be assembled before the time loop; in contrast, Φ is time-dependent and should be updated at each time step.

4. Numerical experiments

In a computational domain $[0,1]^{\mathbf{p}}$, the diffusion Eq. (1) is numerically solved following the procedures outlined in Section 3.4 with $\Psi(t)$ and the initial condition $u(\mathbf{x},t_0)$ set by exact solutions. As discussed in Section 3.2, the boundary condition is also set by exact solutions through the assembling of Φ in Algorithm 1. The jump conditions at the solid–fluid interface are fulfilled differently by EJA and GCAs, as formulated in Section 3. Comparing these methods on this particular issue is exactly the main purpose of this section.

Hundreds of test cases are performed on successively refined uniform grids with varying viscosity $\log_{10}(1/\nu) = 0,1,2,3,4,5,6$. The number of cells along each dimension is $N = 10,20,40,80,160$ in 1-D and 2-D tests and $N = 8,16,32$ in 3-D tests. Spatial and temporal grids are $h = \frac{1}{N}$, $\Delta t = \frac{1}{5N}$. The initial time $t_0 = 20$ and the final time $t_e = 21$.

The convergence rate is defined as

$$\mathcal{O} = \log_2 \frac{\|\mathbf{E}(N)\|}{\|\mathbf{E}(2N)\|}, \tag{45}$$

where $\mathbf{E}(N)$ denotes the error vector between the numerical results and the exact solution at the end of the calculation t_e. The error ratios of GC1 and GC2 to EJA is measured by

$$\mathcal{R}_1^{GCk} = \log_{10} \frac{\|\mathbf{E}\|_1^{GCk}}{\|\mathbf{E}\|_1^{EJA}}, \quad \mathcal{R}_\infty^{GCk} = \log_{10} \frac{\|\mathbf{E}\|_\infty^{GCk}}{\|\mathbf{E}\|_\infty^{EJA}}. \tag{46}$$

They also indicates the order-of-magnitude by which EJA is more accurate than a GCA.

4.1. Setup

Numerical tests are performed in one-, two- and three-dimensional spaces. The setup of one-dimensional tests is shown in Fig. 1(b). For two-dimensional tests, one setup is the simple slope shown in Fig. 3(a) and the other is the circular shape shown in Fig. 5(a). In the three-dimensional tests, the solid phase is a sphere $\{\mathbf{x} : \sqrt{\mathbf{x}\cdot\mathbf{x}} \leqslant \frac{\pi}{5}\}$. Cell classification on a 5^3 grids for this setup is shown in Fig. 5(b).

As explained in Section 2, single Fourier mode solutions are insufficient for examining the converging behavior of GCA, thus we choose exponential function forms as exact solutions. For 1-D tests, the exact solution is,

$$u(y,t) = \begin{cases} \exp\left(a(y-y_B)(1+ct)\right) - 1 & y \geqslant y_B \\ 0 & y < y_B \end{cases}. \tag{47}$$

The 2-D slope tests have the exact solution as

$$u(x,y,t) = \begin{cases} x' \exp\left(a(1+ct)y'\right) - x' & y' \geqslant 0 \\ 0 & y' \leqslant 0 \end{cases}, \tag{48}$$

where the coordinates (x',y') relate to the original coordinates (x,y) as

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x - x_s \\ y \end{pmatrix}, \tag{49}$$

with $x_s = \frac{\pi}{10}$ and $\theta = \frac{\pi}{6}$ as shown in Fig. 3(a).

The exact solution of the 2-D and 3-D sphere tests is

$$u(\mathbf{x},t) = \begin{cases} \exp\left(a(1+ct)(\mathbf{x}\cdot\mathbf{x} - r^2)\right) - 1, & \mathbf{x}\cdot\mathbf{x} > r^2 \\ 0 & \mathbf{x}\cdot\mathbf{x} \leqslant r^2 \end{cases}, \tag{50}$$

with $r = \frac{\pi}{5}$. The forcing terms for the above exact solutions can be easily derived from the governing Eq. (1).

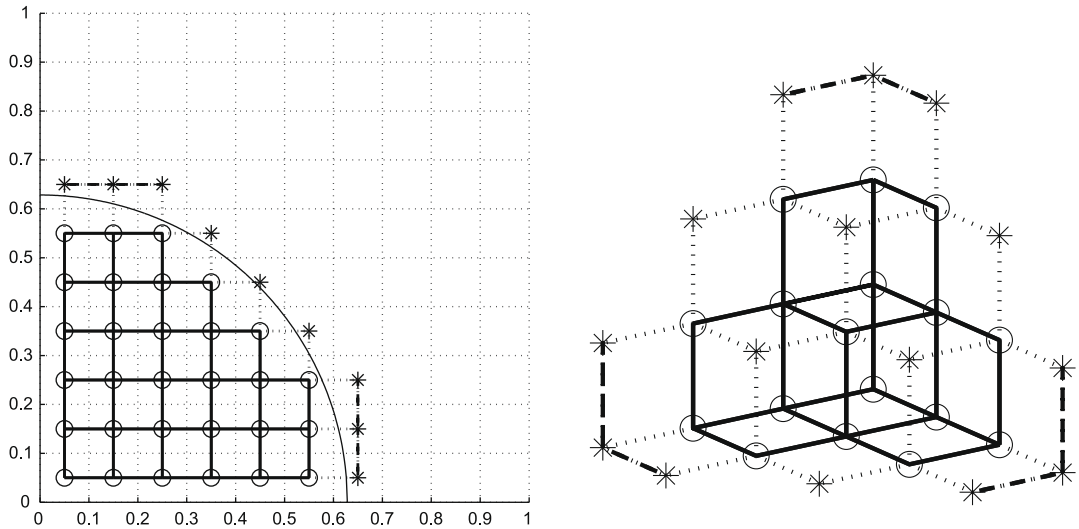


Fig. 5. Spherical setup for 2D and 3D tests. The **D**-balls are centered at the origin with a radius as $\frac{\pi}{5}$. The circular curve in (a) represents the fluid–solid interface. The solid cells are marked by '∘' and the interface cells by '∗'. Two solid cells within each other's stencils are connected with solid line segments; two interface cells with dashdot segments; an interface cell and a solid cell with dotted segments.

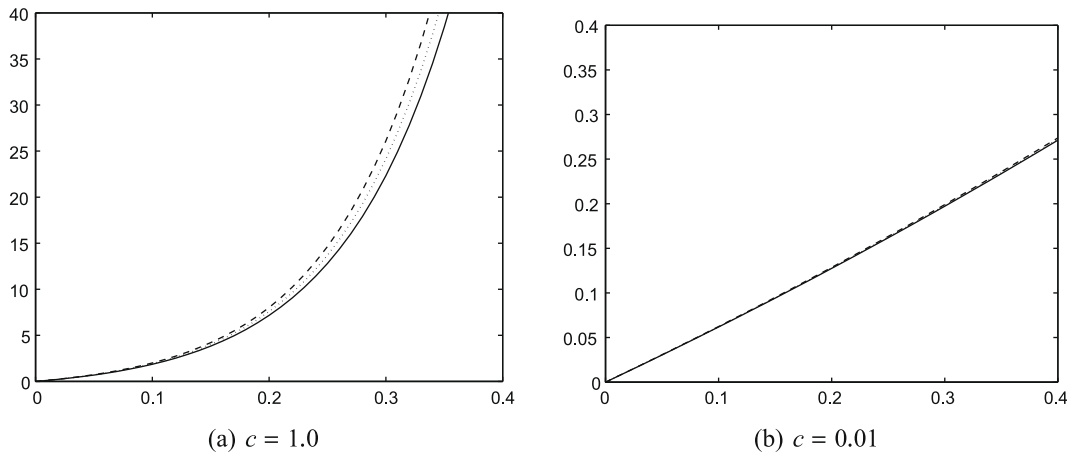


Fig. 6. Plots of (47) with $a = 0.5$ and $y_B = 0$. The horizontal and vertical axis are y and u, respectively. The solid, dotted and dashed lines corresponds to $t = 20, 20.5, 21.0$, respectively.

In Fig. 6, (47) is plotted with $a = 0.5,\ y_B = 0$, and $c = 1,\ 0.01$ at three time instances. As c increases, the time scale of the flow decreases and the flow becomes more transient. $c = 1$ is used in Section 4.2 to show that time-explicit GCA might be unstable for strongly transient flows. In Sections 4.3 and 4.4, when comparing the accuracies of EJA and GCA, a small value $c = 0.01$ is chosen for all tests to make the temporal change of the flow slow so that the errors come mainly from discretizing the spatial diffusion term. In all tests $a = 0.5$ is fixed. These parameters are carefully chosen to make both the velocity scale and the length scale unit size so that a Reynolds number can be conveniently defined as

$$\mathrm{Re} = \frac{1}{\nu}. \tag{51}$$

4.2. Instability of time-explicit GCA

When (36) is used in the time-explicit GCA, a temporal error of $O(\Delta t^2)$ in addition to the spatial error is introduced into the solver. Although each of them is of a lower dimension, when coupled together, they can cause instability for strongly transient flows.

Table 1
Error norms of 1-D tests with $b = 0,\ v = 1,\ c = 1$ and Crank–Nicolson ($\alpha = 0.5$).

Method	$\|\mathbf{E}(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(160)\|_1$
GC1E	3.95e+03	−30.98	8.37e+12	−117	1.49e+48	−259	1.06e+126	−552	1.65e+292
GC2E	3.95e+03	2.35	7.75e+02	2.20	1.69e+02	−45	4.71e+16	−406	7.50e+138
EJA	4.15e+03	2.38	7.95e+02	2.21	1.72e+02	2.11	3.97e+01	2.06	9.54e+00
	$\|\mathbf{E}(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(160)\|_\infty$
GC1E	6.62e+03	−34.20	1.31e+14	−118	3.53e+49	−259	3.75e+127	−553	8.55e+293
GC2E	6.62e+03	2.34	1.31e+03	2.19	2.88e+02	−53	2.83e+18	−406	5.04e+140
EJA	6.69e+03	2.35	1.32e+03	2.19	2.89e+02	2.10	6.74e+01	2.05	1.63e+01

Table 2
Error norms of 1-D tests with $b = 0,\ v = 1,\ c = 1$ and TGA ($\alpha = 0.58$).

Method	$\|\mathbf{E}(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(160)\|_1$
GC1E	3.95e+03	2.35	7.75e+02	2.11	1.79e+02	−37.58	3.68e+13	−69.44	2.96e+34
GC2E	3.95e+03	2.35	7.75e+02	2.20	1.69e+02	2.10	3.94e+01	2.05	9.49e+00
EJA	4.15e+03	2.38	7.95e+02	2.21	1.72e+02	2.11	3.97e+01	2.06	9.54e+00
	$\|\mathbf{E}(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(160)\|_\infty$
GC1E	6.62e+03	2.34	1.31e+03	2.19	2.88e+02	−41.49	8.87e+14	−69.89	9.73e+35
GC2E	6.62e+03	2.34	1.31e+03	2.19	2.88e+02	2.10	6.73e+01	2.05	1.63e+01
EJA	6.69e+03	2.35	1.32e+03	2.19	2.89e+02	2.10	6.74e+01	2.05	1.63e+01

In Table 1, it is shown that both GC1 and GC2 are unstable for the transient flow case with $c = 1$ if time discretization uses C–N. When C–N is replaced by TGA in Table 2, GC2 is able to generate stable solutions while GC1 is still unstable.

The symmetry of the linear solver is often considered as an advantage of the time-explicit GCA, however, this might make the solver susceptible to instabilities for transient flows. Thus, a more sophisticated approximation for the ghost cell value at the next time step is needed in order to simultaneously retain stability and the symmetry of the linear solver. Hereafter only TGA is used in time integration.

4.3. Accuracy and convergence: Mid-range Reynolds numbers

We first show the existence of velocity jumps in Fig. 7, where the final solutions of EJA for a 2-D slope test and a 2-D sphere test are plotted as surfaces over 40×40 grids. The lighting clearly illustrates the derivative jumps of the solution at the solid–fluid interface.

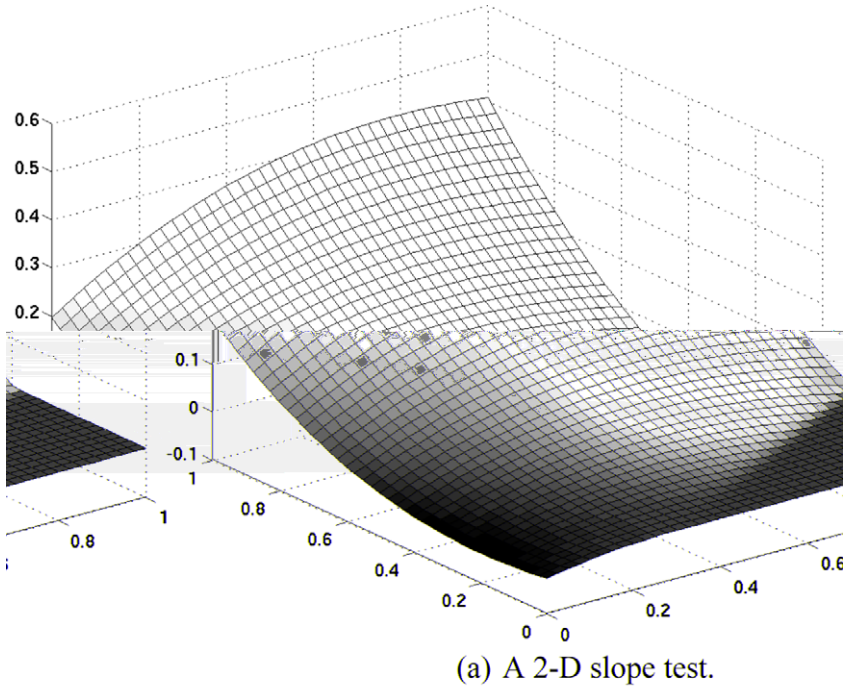
For the 1-D tests, Table 3 shows the results for the regular boundary case $b = 0$ with $v = 10^{-3}$. It is clear that GC1 is much less accurate than GC2, which is in turn less accurate than EJA. In the case of GC1, the grid size has to be reduced to the finest in order for it to reach second-order convergence. The convergence rate of GC2 also varies with the grid size and is close to 3 on the two finest grids. In contrast, the convergence rate of EJA is steadily at 2 regardless of the grid size. These observations confirm the remarks (I) and (V) in Section 2.

In Table 4, the error norms of the same test cases of Table 3 are re-evaluated within a distance to the solid–fluid interface instead of over the whole domain. Comparing Table 4 to Table 3, it is clear that *the maximum error of GC1 and GC2 always happens near the interface* because $\|\mathbf{E}\|_\infty$ remains the same for GC1 and GC2 and the averaged errors of GC1 and GC2 near the interface are roughly five times as large as those over the whole domain. This is due to the fact that within each time step, GC1 and GC2 commit errors of $O(1)$ and $O(h)$, respectively, near the interface, as shown in (6). Apart from deteriorating the accuracy away from the boundary by perturbing the linear systems in (44), these errors remain near the solid–fluid interface. Therefore, if the flow near the solid–fluid interface is of interests, EJA is an much better choice than GC1 and GC2.

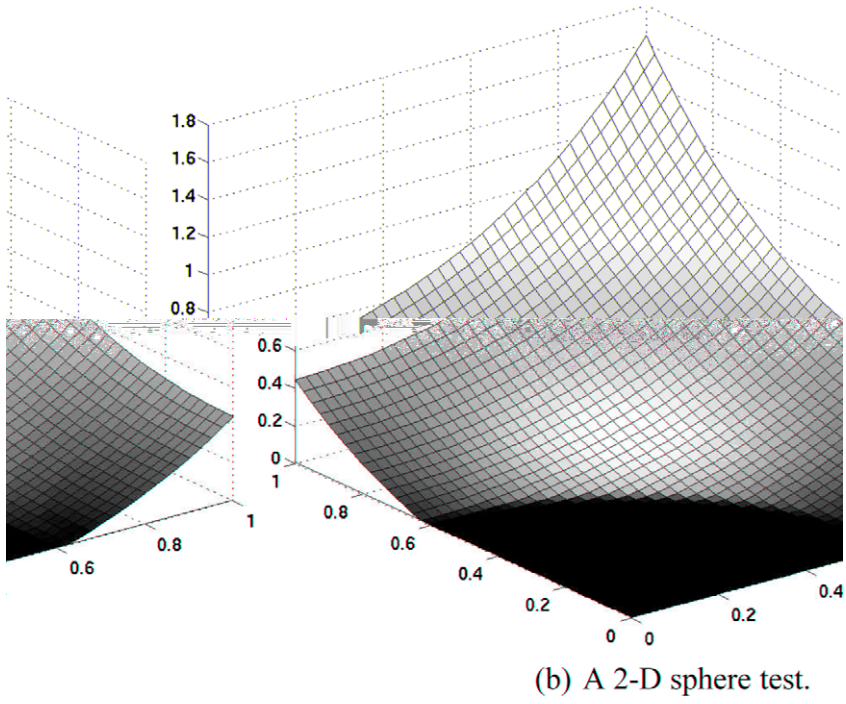
Table 5 shows the results for an irregular boundary case of $b = 0.2$ with otherwise the same configuration as Table 3. Comparing Table 3 to Table 5, the errors of EJA remain almost the same while those of GC1 roughly triple when the solid–fluid interface changes from regular to irregular. Thus, EJA is more independent to the location of the interface than GC1 and GC2.

Let $\mathbf{E}_\tau$ denote the error of the sum of the derivatives $\sum_{d=0}^{\mathbf{D}-1} \frac{\partial u}{\partial x_d}$, evaluated from the solution of u. Tables 6–8 list $\mathbf{E}_\tau$ for the 2-D slope tests, the 2-D sphere tests and the 3-D sphere tests, respectively. In all three tests, GC2 has roughly the same accuracy with EJA while GC1 is much less accurate, especially in the 3-D sphere tests.

Aside from a better accuracy, EJA is a better choice from an efficiency viewpoint, since EJA prevents local errors near the boundary from being distributed to the whole domain, i.e., a lower-dimension operation helps to reduce errors in a higher-dimension space.



(a) A 2-D slope test.



(b) A 2-D sphere test.

Fig. 7. Surface plots of the final solution $u(\mathbf{x}, t_e)$ of EJA on 40×40 grids. $\nu = 10^{-3}$. The parallel lighting shows the jump of $u(\mathbf{x}, t_2)$ at the solid–fluid interface.

4.4. The accuracy and convergence of GCA depend on viscosity

For practical applications, it is necessary to examine the error dependence on viscosity, or equivalently, the Reynolds number defined in (51). To this end, the error norms of $\mathrm{Re} = 1, 10^3, 10^6$ are shown in Tables 9–11 for the 2-D slope tests; Tables 12–14 for the 2-D sphere tests; Tables 15–17 for the 3-D sphere tests. The error ratios as defined in (46) are plotted in Fig. 8 and in Fig. 9 for the 2-D slope tests. Qualitatively the same as Figs. 8 and 9 are the error ratio plots of the 1-D tests

Table 3
Error norms of 1-D tests with $b = 0,\ v = 10^{-3},\ c = 0.01$.

Method	$\|\mathbf{E}(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(160)\|_1$
GC1	8.33e−06	1.34	3.29e−06	1.75	9.78e−07	1.95	2.53e−07	1.99	6.37e−08
GC2	1.25e−06	2.21	2.69e−07	2.53	4.67e−08	2.72	7.06e−09	2.60	1.16e−09
EJA	1.41e−07	1.99	3.56e−08	1.98	9.01e−09	2.00	2.25e−09	2.00	5.62e−10
	$\|\mathbf{E}(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(160)\|_\infty$
GC1	7.85e−05	0.54	5.40e−05	1.31	2.18e−05	1.78	6.33e−06	1.91	1.69e−06
GC2	1.06e−05	1.44	3.90e−06	2.18	8.58e−07	2.82	1.22e−07	2.92	1.60e−08
EJA	1.85e−07	1.98	4.70e−08	2.00	1.18e−08	2.00	2.94e−09	2.00	7.35e−10

Table 4
Error norms of 1-D tests near the interface ($y \in [0, 0.2]$) with $b = 0,\ v = 10^{-3},\ c = 0.01$.

Method	$\|\mathbf{E}(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(160)\|_1$
GC1	4.11e−05	1.33	1.63e−05	1.75	4.85e−06	1.95	1.25e−06	1.99	3.16e−07
GC2	5.59e−06	2.23	1.19e−06	2.61	1.94e−07	2.92	2.57e−08	2.91	3.40e−09
EJA	7.56e−08	1.84	2.11e−08	1.79	6.12e−09	1.94	1.60e−09	1.98	4.04e−10
	$\|\mathbf{E}(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(160)\|_\infty$
GC1	7.85e−05	0.54	5.40e−05	1.31	2.18e−05	1.78	6.33e−06	1.91	1.69e−06
GC2	1.06e−05	1.44	3.90e−06	2.18	8.58e−07	2.82	1.22e−07	2.92	1.60e−08
EJA	1.13e−07	1.89	3.05e−08	1.99	7.69e−09	1.99	1.93e−09	2.00	4.83e−10

Table 5
Error norms of 1-D tests with $b = 0.2,\ v = 10^{-3},\ c = 0.01$.

Method	$\|\mathbf{E}(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(160)\|_1$
GC1	2.08e−05	1.29	8.47e−06	1.71	2.58e−06	1.99	6.51e−07	2.03	1.60e−07
GC2	1.67e−06	2.23	3.56e−07	2.63	5.75e−08	2.85	8.00e−09	2.67	1.25e−09
EJA	1.43e−07	2.01	3.55e−08	1.99	8.96e−09	2.00	2.25e−09	2.00	5.61e−10
Method	$\|\mathbf{E}(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(160)\|_\infty$
GC1	1.97e−04	0.49	1.40e−04	1.27	5.80e−05	1.84	1.62e−05	1.94	4.21e−06
GC2	1.46e−05	1.45	5.35e−06	2.30	1.09e−06	2.93	1.42e−07	2.96	1.83e−08
EJA	1.83e−07	1.97	4.67e−08	1.99	1.17e−08	2.00	2.94e−09	2.00	7.35e−10

Table 6
Derivative error norms of 2-D slope tests with $v = 10^{-3},\ c = 0.01$.

Method	$\|\mathbf{E}_\tau(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}_\tau(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}_\tau(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}_\tau(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}_\tau(160)\|_1$
GC1	5.93e−03	0.87	3.25e−03	0.57	2.18e−03	1.00	1.09e−03	1.09	5.12e−04
GC2	2.06e−03	1.96	5.29e−04	2.05	1.28e−04	2.07	3.05e−05	2.02	7.54e−06
EJA	2.02e−03	2.08	4.79e−04	2.01	1.19e−04	2.05	2.88e−05	2.00	7.20e−06
	$\|\mathbf{E}_\tau(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}_\tau(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}_\tau(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}_\tau(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}_\tau(160)\|_\infty$
GC1	1.24e−02	0.70	7.67e−03	0.54	5.26e−03	0.96	2.70e−03	1.11	1.25e−03
GC2	2.71e−03	1.81	7.73e−04	2.09	1.81e−04	1.85	5.03e−05	2.08	1.19e−05
EJA	2.69e−03	2.06	6.45e−04	2.03	1.58e−04	2.03	3.88e−05	2.00	9.69e−06

Table 7
Derivative error norms of 2-D sphere tests with $v = 10^{-3},\ c = 0.01$.

Method	$\|\mathbf{E}_\tau(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}_\tau(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}_\tau(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}_\tau(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}_\tau(160)\|_1$
GC1	1.13e−02	−1.03	2.32e−02	1.67	7.28e−03	0.60	4.80e−03	0.98	2.43e−03
GC2	2.01e−02	2.00	5.01e−03	2.30	1.02e−03	2.32	2.04e−04	2.18	4.49e−05
EJA	1.50e−02	2.26	3.15e−03	2.15	7.11e−04	2.10	1.65e−04	2.02	4.06e−05
	$\|\mathbf{E}_\tau(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}_\tau(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}_\tau(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}_\tau(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}_\tau(160)\|_\infty$
GC1	3.95e−02	−0.09	4.19e−02	0.74	2.52e−02	0.39	1.92e−02	1.03	9.40e−03
GC2	2.76e−02	1.97	7.04e−03	2.24	1.49e−03	2.15	3.36e−04	2.05	8.11e−05
EJA	1.80e−02	2.19	3.97e−03	2.09	9.34e−04	2.06	2.23e−04	2.04	5.42e−05

Table 8
Derivative error norms of 3-D sphere tests with $v = 10^{-3},\ c = 0.01$.

Method	$\|\mathbf{E}_\tau(8)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}_\tau(16)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}_\tau(32)\|_1$
GC1	3.07e−02	0.32	2.45e−02	0.57	1.65e−02
GC2	3.20e−02	1.98	8.09e−03	2.29	1.65e−03
EJA	2.62e−02	2.36	5.09e−03	2.19	1.11e−03
	$\|\mathbf{E}_\tau(8)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}_\tau(16)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}_\tau(32)\|_\infty$
GC1	5.59e−02	−0.48	7.81e−02	−0.20	8.95e−02
GC2	5.04e−02	1.81	1.44e−02	2.26	3.00e−03
EJA	3.51e−02	2.21	7.61e−03	2.10	1.77e−03

Table 9
Error norms of 2-D slope tests with $v = 1,\ c = 0.01$.

Method	$\|\mathbf{E}(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(160)\|_1$
GC1	1.59e−04	3.32	1.59e−05	2.32	3.17e−06	2.40	5.99e−07	2.36	1.17e−07
GC2	2.91e−05	2.73	4.37e−06	2.55	7.48e−07	2.33	1.49e−07	2.19	3.26e−08
EJA	9.45e−06	2.18	2.08e−06	2.10	4.85e−07	2.06	1.17e−07	2.03	2.86e−08
	$\|\mathbf{E}(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(160)\|_\infty$
GC1	1.32e−03	2.52	2.30e−04	1.50	8.15e−05	1.90	2.19e−05	2.15	4.95e−06
GC2	7.37e−05	2.76	1.09e−05	3.03	1.33e−06	2.25	2.79e−07	2.12	6.41e−08
EJA	1.74e−05	2.09	4.08e−06	2.05	9.83e−07	2.03	2.41e−07	2.01	5.96e−08

Table 10
Error norms of 2-D slope tests with $v = 10^{-3},\ c = 0.01$.

Method	$\|\mathbf{E}(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(160)\|_1$
GC1	2.56e−05	2.21	5.53e−06	2.44	1.02e−06	2.78	1.48e−07	2.67	2.34e−08
GC2	1.45e−06	2.27	3.02e−07	2.73	4.55e−08	2.61	7.46e−09	2.46	1.35e−09
EJA	2.26e−07	2.01	5.61e−08	2.01	1.40e−08	2.02	3.45e−09	2.01	8.54e−10
	$\|\mathbf{E}(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(160)\|_\infty$
GC1	4.97e−04	1.38	1.91e−04	1.33	7.62e−05	1.83	2.15e−05	2.13	4.92e−06
GC2	2.03e−05	1.71	6.22e−06	2.56	1.06e−06	2.79	1.53e−07	2.85	2.11e−08
EJA	2.97e−07	1.98	7.54e−08	2.00	1.89e−08	2.00	4.71e−09	2.00	1.18e−09

Table 11
Error norms of 2-D slope tests with $v = 10^{-6},\ c = 0.01$.

Method	$\|\mathbf{E}(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(160)\|_1$
GC1	3.34e−08	1.47	1.21e−08	0.91	6.45e−09	1.15	2.89e−09	1.14	1.31e−09
GC2	1.59e−09	1.91	4.21e−10	2.12	9.70e−11	1.97	2.47e−11	2.01	6.11e−12
EJA	2.39e−10	1.99	6.02e−11	2.00	1.50e−11	2.00	3.75e−12	1.98	9.49e−13
	$\|\mathbf{E}(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(160)\|_\infty$
GC1	6.74e−07	0.50	4.75e−07	−0.49	6.69e−07	−0.07	7.03e−07	0.21	6.07e−07
GC2	2.46e−08	0.78	1.43e−08	1.02	7.08e−09	0.98	3.58e−09	1.01	1.78e−09
EJA	3.19e−10	1.72	9.69e−11	2.11	2.24e−11	2.13	5.12e−12	1.92	1.35e−12

Table 12
Error norms of 2-D sphere tests with $v = 1,\ c = 0.01$.

Method	$\|\mathbf{E}(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(160)\|_1$
GC1	8.44e−04	2.50	1.49e−04	1.85	4.13e−05	2.59	6.88e−06	2.15	1.55e−06
GC2	9.44e−04	2.61	1.55e−04	2.34	3.07e−05	2.16	6.89e−06	2.09	1.62e−06
EJA	5.64e−04	2.24	1.20e−04	2.14	2.73e−05	2.08	6.44e−06	2.04	1.57e−06
	$\|\mathbf{E}(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(160)\|_\infty$
GC1	6.22e−03	1.95	1.62e−03	2.49	2.88e−04	1.37	1.12e−04	1.81	3.18e−05
GC2	1.43e−03	2.34	2.83e−04	2.18	6.25e−05	2.07	1.48e−05	2.04	3.60e−06
EJA	1.15e−03	2.17	2.55e−04	2.09	5.99e−05	2.05	1.45e−05	2.02	3.56e−06

Table 13
Error norms of 2-D sphere tests with $\nu = 10^{-3},\ c = 0.01$.

Method	$\|\mathbf{E}(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(160)\|_1$
GC1	1.49e−04	1.54	5.14e−05	2.75	7.61e−06	1.68	2.38e−06	2.27	4.93e−07
GC2	5.25e−05	2.37	1.02e−05	2.46	1.85e−06	2.24	3.90e−07	2.18	8.62e−08
EJA	2.08e−05	1.99	5.23e−06	2.02	1.29e−06	2.03	3.14e−07	2.02	7.75e−08
	$\|\mathbf{E}(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(160)\|_\infty$
GC1	2.11e−03	0.72	1.28e−03	2.16	2.87e−04	1.36	1.12e−04	1.81	3.18e−05
GC2	3.68e−04	2.02	9.07e−05	2.74	1.36e−05	2.72	2.06e−06	2.93	2.71e−07
EJA	5.44e−05	2.04	1.32e−05	1.99	3.33e−06	2.02	8.23e−07	2.01	2.04e−07

Table 14
Error norms of 2-D sphere tests with $\nu = 10^{-6},\ c = 0.01$.

Method	$\|\mathbf{E}(10)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(20)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(40)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(80)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(160)\|_1$
GC1	1.76e−07	0.94	9.21e−08	1.62	3.01e−08	0.60	1.98e−08	1.00	9.89e−09
GC2	5.60e−08	2.16	1.25e−08	2.21	2.70e−09	1.84	7.53e−10	2.05	1.82e−10
EJA	2.22e−08	1.97	5.65e−09	2.00	1.41e−09	2.00	3.54e−10	2.00	8.87e−11
	$\|\mathbf{E}(10)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(20)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(40)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(80)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(160)\|_\infty$
GC1	2.61e−06	−0.01	2.63e−06	0.28	2.17e−06	−0.40	2.86e−06	0.01	2.84e−06
GC2	4.50e−07	1.29	1.85e−07	1.12	8.47e−08	0.72	5.13e−08	1.03	2.50e−08
EJA	6.08e−08	1.86	1.68e−08	1.93	4.41e−09	1.97	1.13e−09	2.01	2.80e−10

Table 15
Error norms of 3-D sphere tests with $\nu = 1,\ c = 0.01$.

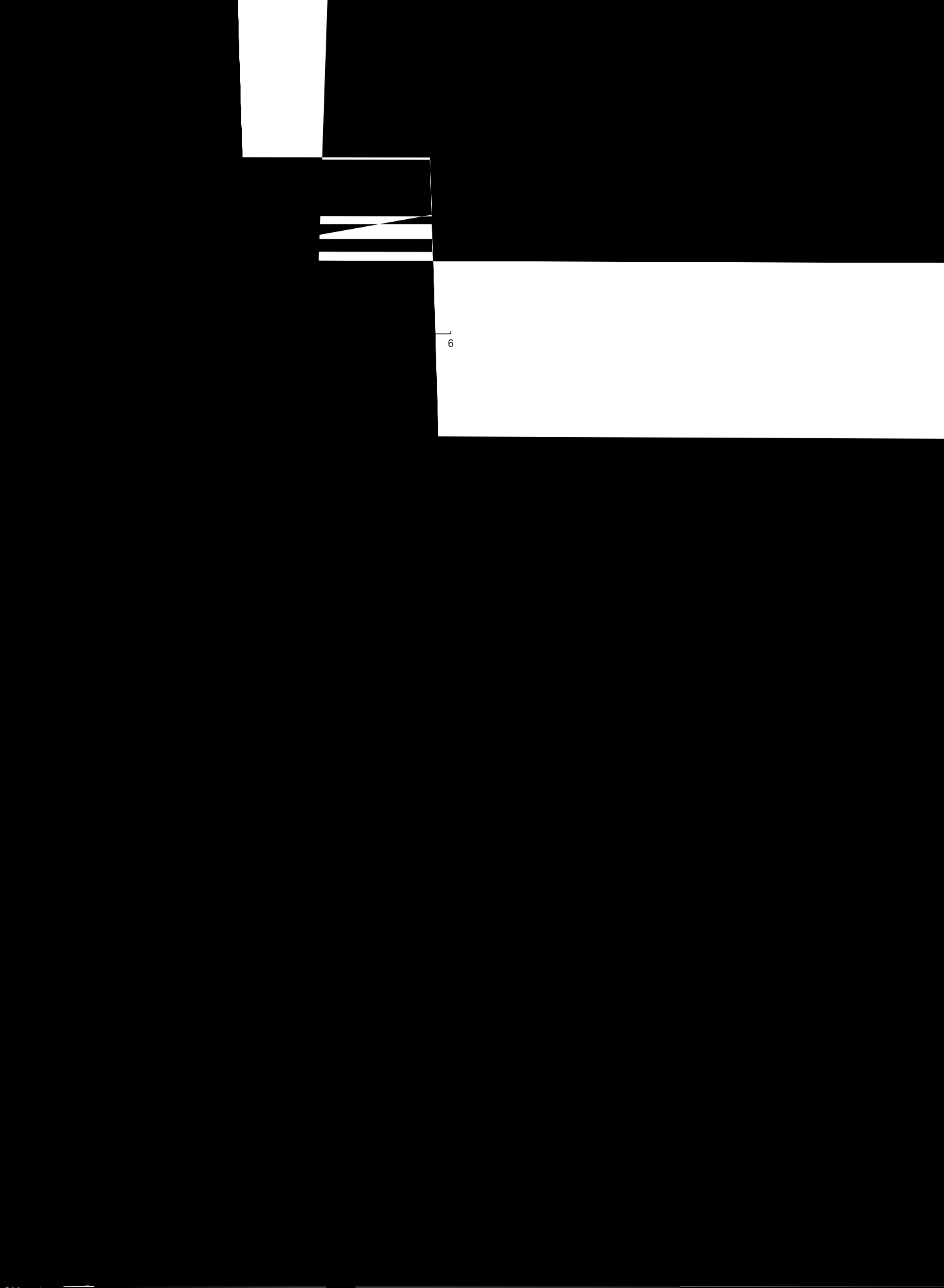
Method	$\|\mathbf{E}(8)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(16)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(32)\|_1$
GC1	2.17e−03	2.73	3.27e−04	2.33	6.52e−05
GC2	1.78e−03	2.45	3.25e−04	2.27	6.74e−05
EJA	1.53e−03	2.36	2.98e−04	2.21	6.46e−05
	$\|\mathbf{E}(8)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(16)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(32)\|_\infty$
GC1	5.33e−03	1.33	2.11e−03	1.69	6.55e−04
GC2	3.58e−03	2.22	7.70e−04	2.13	1.76e−04
EJA	3.47e−03	2.20	7.54e−04	2.11	1.75e−04

Table 16
Error norms of 3-D sphere tests with $\nu = 10^{-3},\ c = 0.01$.

Method	$\|\mathbf{E}(8)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(16)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(32)\|_1$
GC1	1.44e−04	1.91	3.85e−05	2.18	8.52e−06
GC2	7.41e−05	2.11	1.72e−05	2.19	3.76e−06
EJA	5.55e−05	2.03	1.36e−05	2.05	3.29e−06
	$\|\mathbf{E}(8)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(16)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(32)\|_\infty$
GC1	1.68e−03	0.01	1.66e−03	1.35	6.53e−04
GC2	4.67e−04	1.68	1.46e−04	2.60	2.39e−05
EJA	2.07e−04	2.11	4.78e−05	1.92	1.26e−05

Table 17
Error norms of 3-D sphere tests with $\nu = 10^{-6},\ c = 0.01$.

Method	$\|\mathbf{E}(8)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(16)\|_1$	$\mathcal{O}_1$	$\|\mathbf{E}(32)\|_1$
GC1	1.67e−07	1.44	6.17e−08	1.13	2.83e−08
GC2	7.74e−08	2.01	1.93e−08	2.02	4.76e−09
EJA	5.75e−08	1.99	1.45e−08	2.00	3.63e−09
	$\|\mathbf{E}(8)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(16)\|_\infty$	$\mathcal{O}_\infty$	$\|\mathbf{E}(32)\|_\infty$
GC1	2.27e−06	−0.62	3.48e−06	−0.23	4.08e−06
GC2	5.62e−07	0.89	3.03e−07	0.91	1.61e−07
EJA	2.31e−07	1.77	6.76e−08	1.88	1.83e−08


6

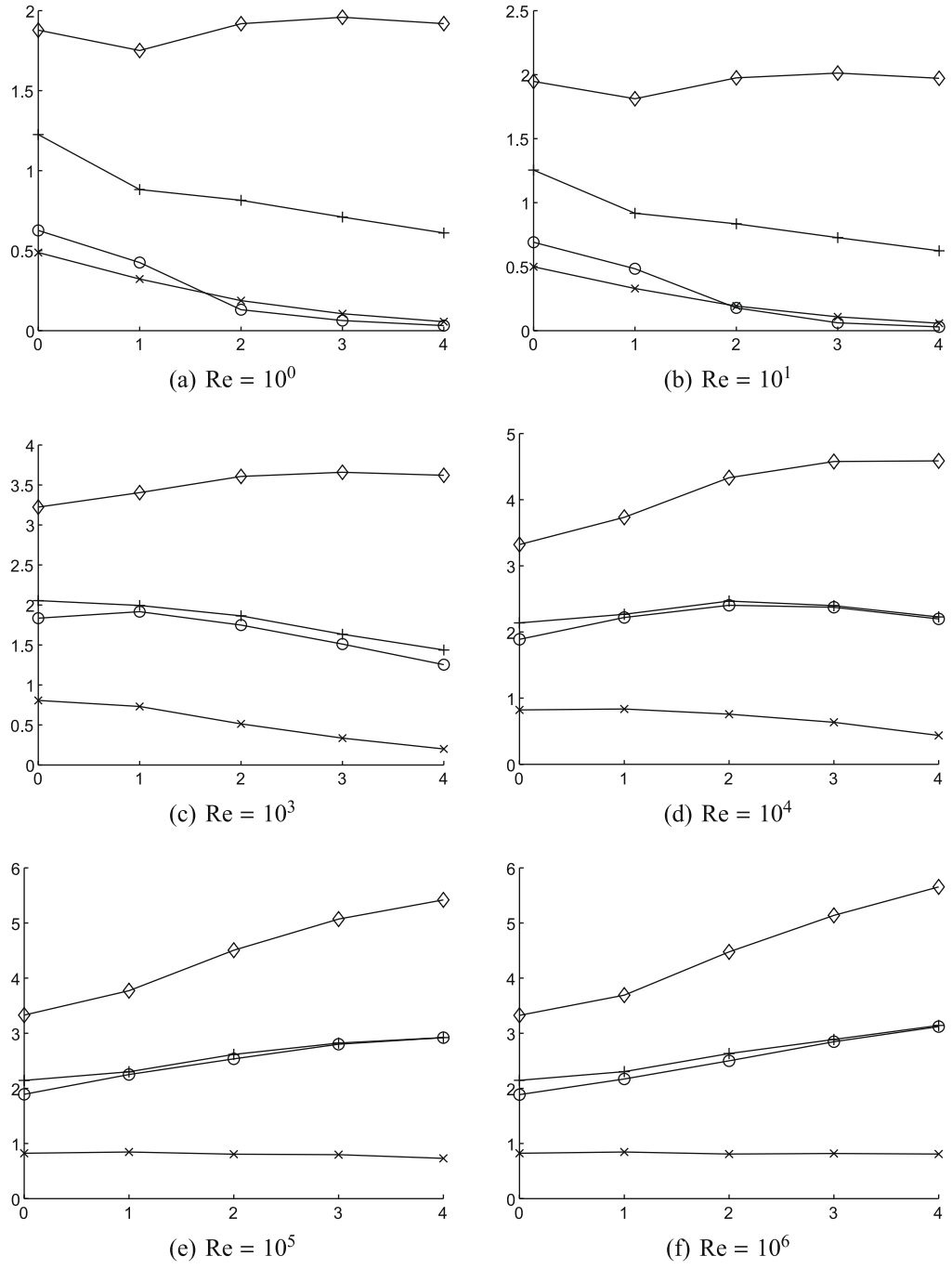


Fig. 9. 2D slope test convergence rates of GC1 and GC2 with respect to the Reynolds number as defined in (51). The horizontal axis is $\log_2 \frac{N}{N_0}$ with $N_0 = 10$; the vertical axis is $\mathcal{R}^{GCk}$ as defined in (46). '+', '◇', '*', and '○' represent $\mathcal{R}_1^{GC1}$, $\mathcal{R}_\infty^{GC1}$, $\mathcal{R}_1^{GC2}$, and $\mathcal{R}_\infty^{GC2}$, respectively. An increasing curve implies convergence worse than second-order while a decreasing curve implies convergence better than second-order. The value of a vertical coordinate is the order-of-magnitude by which the GCA is less accurate than EJA.

curves have no descending parts, implying that even the smallest grid size $\frac{1}{2^4 N_0}$ is still bigger than h_{crit}. Note that at this resolution, the accuracy of EJA in the 1-D tests and the 2-D slope tests is already close to machine precision for high Reynolds numbers.

Although GCAs can be second-order accurate or even better given small enough grid sizes, their errors are much larger than that of EJA in a wide range of grid sizes. This is particularly true for GC1. Furthermore, because of limited computational

resources, h_{crit} can be unrealistically small, depending on the particular type of the viscous flow. As an example, for $\nu = 10^{-6}$, second-order convergence rates are impossible to show for the ∞-norms of GC1 and GC2, given the length of the mantissa of a double-precision floating number.

Aside from the analysis, an interpretation based on physical intuition can also explain Figs. 8, and 9: when the fluid phase is strongly diffusive (corresponding to small Re), lower-dimensional perturbations can be effectively damped out. However, when viscosity becomes smaller and smaller, the same perturbations will cause larger and larger errors for the whole system. Worse, the errors are committed and accumulated within each time step. This time-accumulation of errors at high Reynolds numbers is the most important reason that GC1 and GC2 should be avoided for handling solid–fluid interfaces. This statement also holds for general practical viscous flows because the advection terms in the Navier–Stokes equations are usually treated explicitly and they correspond to the nonhomogeneous term $f(\mathbf{x}, t)$ in (1).

5. Concluding remarks

Although the error equations are the same for Poisson's equation and the diffusion equation, the solution error of the diffusion equation might accumulate to an unacceptable degree over $O(1/\Delta t) = O(1/h)$ time steps, especially when the truncation error commited to the difference system at each time step is of a larger orde-of-magnitude than that of the intended solution error. We have shown, theoretically and numerically, that the truncation error of the diffusion equation has to be $O(h^2)$ to achieve rigorous second-order accuracy for the solution. Despite the wide usage of the linear extrapolation of GC1 for regular or irregular boundaries (i.e. $u_{-1} = -u_0$), it has serious shortcomings when coupled to any method aiming for second-order accuracy. GC1 ignores all the jumps of velocity-derivatives, whereas *these jumps exist anyway regardless of the boundary being regular or not*. The numerical results in this paper confirm that GC1 is far less than satisfactory even for regular boundaries. For inviscid flows, GC1 generates first-order truncation error near the boundary, thus a second-order method should still avoid using it.

We have proposed the explicit jump approximation method as a more accurate and more stable way to handle solid–fluid interfaces for viscous flows. Although the resulting linear system is not symmetric, it is strictly diagonally dominant. Currently, EJA only applies to stationary boundaries; the generalization of EJA to moving boundary problems will be reported in a future paper.

EJA finds the extrapolation locations by intersecting the solid–fluid interface to lines parallel to coordinate axes. As such, accurate interface tracking is desirable, particularly when the solid–fluid interface is moving. The polygonal area mapping (PAM) method [26] represents material areas with piecewise polygons and tracks the interface by polygon-clipping algorithms from computational geometry. One attractive advantage of PAM is its high accuracy, particularly for sharp corners and singularity points; another is its rigorous second-order convergence [25]. These make PAM the ideal candidate to be coupled with EJA for moving boundary problems.

PAM has already been coupled to a projection method to form a hybrid continuum-particle model (HyPAM) [28] for incompressible free-surface flows. In this model, a single-phase decomposition algorithm based on graph theory is used to decompose the water phase into a continuum zone and a particle zone, where different governing equations are applied. It is shown that the accuracy for practical free-surface flows, such as dam-break generated bores and swash-zone hydrodynamics [27], can be improved dramatically by replacing volume-of-fluid methods with PAM and by changing how the velocity field near the free-surface is calculated. A long term goal is to combine these three elements together to form an air–fluid–solid model, which could be of much value for practical applications of free-surface flows and flow-structures interactions.

Acknowledgements

The first author, Qinghai Zhang, would like to thank Phillip Colella for his helpful discussions. The second author, Philip L.-F. Liu, would like to acknowledge the support from the Humboldt Foundation, Germany NSF and NY Sea Grant Research Institute during the preparation of this paper. Both authors also thank the referees for their helpful suggestions and comments.

Appendix A. This appendix derives formulas for one-dimensional jump approximations. It is important to emphasize that the '+' and '−' superscripts in EJA represent the positive and negative sides of the jump with respect to a coordinate, not with respect to the two phases.

Referring to Fig. 1(b), Taylor expansions at y_B in the fluid phase result in a linear system

$$\begin{pmatrix} u_0 \\ u_1 \\ u_2 \end{pmatrix} = u(y_B^+) + \begin{pmatrix} \epsilon & \epsilon^2 & \epsilon^3 \\ (\epsilon+1) & (\epsilon+1)^2 & (\epsilon+1)^3 \\ (\epsilon+2) & (\epsilon+2)^2 & (\epsilon+2)^3 \end{pmatrix} \begin{pmatrix} h & & \\ & \frac{h^2}{2} & \\ & & \frac{h^3}{6} \end{pmatrix} \begin{pmatrix} \frac{\partial u}{\partial y}(y_B^+) \\ \frac{\partial^2 u}{\partial y^2}(y_B^+) \\ \frac{\partial^3 u}{\partial y^3}(y_B^+) \end{pmatrix} + O(h^4),$$

where $\epsilon = \frac{1}{2} - b$. Hence the jumps of the first derivatives are

$$\mathbf{J}_y^{1,3} = \left[\left[\begin{array}{c}\frac{\partial u}{\partial y}(y_B)\\ \frac{\partial^2 u}{\partial y^2}(y_B)\\ \frac{\partial^3 u}{\partial y^3}(y_B)\end{array}\right]\right] = \left(\begin{array}{c}\frac{\partial u}{\partial y}(y_B^+)\\ \frac{\partial^2 u}{\partial y^2}(y_B^+)\\ \frac{\partial^3 u}{\partial y^3}(y_B^+)\end{array}\right) = \mathbf{H}\mathbf{W}_3\left(\begin{array}{c}u_0 - u(y_B^+)\\ u_1 - u(y_B^+)\\ u_2 - u(y_B^+)\end{array}\right) + \left(\begin{array}{c}O(h^3)\\ O(h^2)\\ O(h)\end{array}\right),$$

where the diagonal matrix is

$$\mathbf{H} = \mathrm{diag}\left(\frac{1!}{h}, \frac{2!}{h^2}, \frac{3!}{h^3}\right), \tag{52}$$

and the extrapolation matrix is

$$\mathbf{W}_3 = \left(\begin{array}{ccc}\frac{(\epsilon+1)(\epsilon+2)}{2\epsilon} & -\frac{\epsilon(\epsilon+2)}{\epsilon+1} & \frac{\epsilon(\epsilon+1)}{2(\epsilon+2)}\\ -\frac{2\epsilon+3}{2\epsilon} & 2 & -\frac{2\epsilon+1}{2(\epsilon+2)}\\ \frac{1}{2\epsilon} & -\frac{1}{\epsilon+1} & \frac{1}{2(\epsilon+2)}\end{array}\right). \tag{53}$$

The case that the solid phase lies at the positive side of the fluid phase can be derived similarly to reach a general result that do not change the forms of (52) and (53):

$$\mathbf{J}_y^{1,3} = \left[\left[\frac{\partial u}{\partial y}, \frac{\partial^2 u}{\partial y^2}, \frac{\partial^3 u}{\partial y^3}\right]\right]_B^T = \mathbf{S}\mathbf{H}\mathbf{W}_3\delta\mathbf{u}_y + O(h^3, h^2, h)^T, \tag{54}$$

where $\mathbf{S}$ = diag(s,1,s), $s = 1$ if the solid–fluid interface lies at the negative side of the fluid; otherwise $s = -1$. The extrapolation vector is

$$\delta\mathbf{u}_y = \left(u_j, u_{j+s}, u_{j+2s}\right)^T - u(y_B^+). \tag{55}$$

In the case when not enough data are available for $\mathbf{W}_3$, the extrapolation matrix is reduced to $\mathbf{W}_1 = 1$ or

$$\mathbf{W}_2 = \left(\begin{array}{cc}\frac{\epsilon+1}{\epsilon} & -\frac{\epsilon}{\epsilon+1}\\ -\frac{1}{\epsilon} & \frac{1}{\epsilon+1}\end{array}\right). \tag{56}$$

Applying (54) to (25) yields (28), where $\mathbf{S}$ is canceled due to the selecting property of (25).

References

[1] P.A. Berthelsen, O.M. Faltinsen, A local directional ghost cell approach for incompressible viscous flow problems with irregular boundaries, J. Comput. Phys. 227 (2008) 4354–4397, doi:10.1016/j.jcp.2007.12.022.
[2] S.J. Farlow, Partial Differential Equations for Scientists and Engineers, Dover Publications, 1993, ISBN-13: 978-0486676203.
[3] A.L. Fogelson, J.P. Keener, Immersed interface methods for Neumann and related problems in two and three dimensions, SIAM J. Sci. Comput. 22 (5) (2000) 1630–1654.
[4] F. Gibou, R. Fedkiw, A fourth-order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problem, J. Comput. Phys. 202 (2005) 577–601, doi:10.1016/j.jcp.2004.07.018.
[5] K. Ito, Z. Li, Y. Kyei, Higher-order, Cartesian grid based finite difference schemes for elliptic equations on irregular domains, SIAM J. Sci. Comput. 27 (1) (2005) 346–367.
[6] H. Johansen, P. Colella, A Cartesian grid embedded boundary method for Poisson's equation on irregular domains, J. Comput. Phys. 147 (1998) 60–85.
[7] J.D. Lawson, J.L. Morris, The extrapolation of first-order methods for parabolic partial differential equations. I, SIAM J. Numer. Anal. 15 (6) (1978) 1212–1224.
[8] D.V. Le, B.C. Khoo, J. Peraire, An immersed interface method for viscous incompressible flows involving rigid and flexible boundaries, J. Comput. Phys. 220 (2006) 109–138, doi:10.1016/j.jcp.2006.05.004.
[9] L. Lee, R.J. Leveque, An immersed interface method for incompressible Navier–Stokes equations, SIAM J. Sci. Comput. 25 (3) (2003) 832–856, doi:10.1137/S1064827502414060.
[10] R.J. Leveque, Z. Li, Immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (4) (1994) 1019–1044.
[11] R.J. Leveque, Z. Li, Immersed interface methods for Stokes flow with elastic boundaries or surface tension, SIAM J. Sci. Comput. 18 (3) (1997) 709–735.
[12] Z. Li, M.-C. Lai, The immersed interface method for the Navier–Stokes equations with singular forces, J. Comput. Phys. 171 (2) (2001) 822–842, doi:10.1006/jcph.2001.6813.
[13] A. Mayo, The fast solution of Poisson's and the biharmonic equations on irregular regions, SIAM J. Numer. Anal. 21 (1984) 285–299.
[14] P. McCorquodale, P. Colella, D.P. Grote, J.-L. Vay, A node-centered local refinement algorithm for Poisson's equation in complex geometries, J. Comput. Phys. 201 (2004) 34–60, doi:10.1006/jcph.2004.04.022.
[15] P. McCorquodale, P. Colella, H. Johansen, A Cartesian grid embedded boundary method for the heat equation on irregular domains, J. Comput. Phys. 173 (2001) 620–635, doi:10.1006/jcph.2001.6900.
[16] R. Mittal, G. Iaccarino, Immersed boundary methods, Ann. Rev. Fluid Mech. 37 (2005) 239–261, doi:10.1146/annurev.fluid.37.061903.175743.
[17] D.A. Swayne, Time-dependent boundary and interior forcing in locally one-dimensional schemes, SIAM J. Sci. Stat. Comput. 8 (5) (1987) 755–767.
[18] Y.-H. Tseng, J.H. Ferziger, A ghost cell immersed boundary method for flow in complex geometry, J. Comput. Phys. 192 (2003) 593–623, doi:10.1016/j.jcp.2003.07.024.
[19] E.H. Twizell, A.B. Gumel, M.A. Arigu, Second-order, L_0-stable methods for the heat equation with time-dependent boundary conditions, Adv. Comput. Math. 6 (1996) 333–352.
[20] H.S. Udaykumar, R. Mittal, W. Shyy, Computation of solid–fluid phase fronts in the sharp interface limit on fixed grids, J. Comput. Phys. 153 (1999) 535–574.
[21] A. Wiegmann, K.P. Bube, The immersed interface method for nonlinear differential equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 35 (1) (1998) 177–200.

[22] A. Wiegmann, K.P. Bube, The explicit jump immersed interface method: finite difference methods for PDEs with piecewise smooth solutions, SIAM J. Numer. Anal. 37 (3) (2000) 827–862.
[23] S. Xu, Z.J. Wang, An immersed interface method for simulating the interaction of a fluid with moving boundaries, J. Comput. Phys. 216 (2006) 454–493, doi:10.1016/j.jcp.2005.12.016.
[24] S. Xu, Z.J. Wang, Systematic derivation of jump conditions for the immersed interface method in three-dimensional flow simulation, SIAM J. Sci. Comput. 27 (6) (2006) 1948–1980, doi:10.1137/040604960.
[25] Q. Zhang, On explicit interface tracking, SIAM J. Numer. Anal. (2010), submitted for publication.
[26] Q. Zhang, P.L.-F. Liu, A new interface tracking method: the polygonal area mapping method, J. Comput. Phys. 227 (8) (2008) 4063–4088, doi:10.1016/j.jcp.2007.12.014.
[27] Q. Zhang, P.L.-F. Liu, A numerical study of swash flows generated by bores, Coastal Eng. 55 (12) (2008) 1113–1134, doi:10.1016/j.coastaleng.2008.04.010.
[28] Q. Zhang, P.L.-F. Liu, HyPAM : A hybrid continuum-particle model for incompressible free-surface flows, J. Comput. Phys. 228 (4) (2009) 1312–1342, doi:10.1016/j.jcp.2008.10.029.
[29] X. Zhong, A new high-order immersed interface method for solving elliptic equations with imbedded interface of discontinuity, J. Comput. Phys. 225 (2007) 1066–1099, doi:10.1016/j.jcp.2007.01.017.
[30] Y.C. Zhou, S. Zhao, M. Feig, G.W. Wei, High order matched interface and boundary method for elliptic equations with discontinuous coefficients and singular sources, J. Comput. Phys. 213 (2006) 1–30, doi:10.1016/j.jcp.2005.07.022.