Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 225 (2007) 2138–2174
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

A coupling interface method for elliptic interface problems ☆

I-Liang Chern [a,*], Yu-Chen Shu [b,c]

[a] Department of Mathematics, Taida Institute for Mathematical Sciences, National Center for Theoretical Science at Taipei, National Taiwan University, Taipei 106, Taiwan
[b] Department of Mathematics, National Taiwan University, Taipei, Taiwan
[c] Division of Mechanics, Research Center for Applied Sciences, Academia Sinica, Taipei, Taiwan

Received 6 July 2006; received in revised form 19 January 2007; accepted 9 March 2007
Available online 28 March 2007

Abstract

We propose a coupling interface method (CIM) under Cartesian grid for solving elliptic *complex* interface problems in arbitrary dimensions, where the coefficients, the source terms, and the solutions may be discontinuous or singular across the interfaces. It consists of a first-order version (CIM1) and a second-order version (CIM2).

In one dimension, the CIM1 is derived from a linear approximation on both sides of the interface. The method is extended to high dimensions through a dimension-by-dimension approach. To connect information from each dimension, a coupled equation for the first-order derivatives is derived through the jump conditions in each coordinate direction. The resulting stencil uses the standard 5 grid points in two dimensions and 7 grid points in three dimensions. Similarly, the CIM2 is derived from a quadratic approximation in each dimension. In high dimensions, a coupled equation for the principal second-order derivatives $u_{x_k x_k}$ is derived through the jump conditions in each coordinate direction. The cross derivatives are approximated by one-side interpolation. This approach reduces the number of grid points needed for one-side interpolation. The resulting stencil involves 8 grid points in two dimensions and 12–14 grid points in three dimensions.

A numerical study for the condition number of the resulting linear system of the CIM2 in one dimension has been performed. It is shown that the condition number has the same behavior as that of the discrete Laplacian, independent of the relative location of the interface in a grid cell. Further, we also give a proof of the solvability of the coupling equations, provided the curvature κ of the interface satisfies $\kappa h \leqslant \text{Const}$, where h is the mesh size.

The CIM1 requires that the interface intersects each grid segment (the segment connecting two adjacent grid points) at most once. This is a very mild restriction and is always achievable by refining meshes. The CIM2 requires basically that the interface does not intersect two adjacent grid segments simultaneously. In practice, we classify the underlying Cartesian grid points into interiors, normal on-fronts, and exceptionals, where a standard central finite difference method, the CIM2, and the CIM1 are adopted, respectively. This hybrid CIM maintains second-order accuracy in most applications due to the fact that usually in d dimensions, the number of normal on-front grid points is $O(h^{1-d})$ and the number of the exceptional points is O(1).

Numerical convergence tests for the CIM1 and CIM2 are performed. A comparison study with other interface methods is also reported. Algebraic multigrid method is employed to solve the resulting linear system. Numerical tests demonstrate that CIM1 and CIM2 are respectively first order and second order in the maximal norm with less error as compared with

☆ This work is supported by the National Science Council of the Republic of China under grant number NSC 95-2115-M-002-009.
* Corresponding author. Tel.: +886 2 23632858.
E-mail addresses: chern@math.ntu.edu.tw (I.-L. Chern), scc@math.ntu.edu.tw (Y.-C. Shu).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2007.03.012

other methods. In addition, this hybrid CIM passes many tests of complex interface problems in two and three dimensions. Therefore, we believe that it is a competitive method for complex interface problems.

MSC: 65F30

Keywords: Elliptic interface problems; Discontinuous coefficients; Singular sources; Cartesian grid; Coupling interface method; Immersed interface method; Algebraic multigrid method

1. Introduction

Interface problems are those physical problems whose solutions are composed of several components separated by interfaces. These interfaces could be material interfaces, phase boundaries, flame fronts, physical boundaries, etc. Sometimes, they may dynamically move. The interface problems appear in fluid dynamics, solid mechanics, electrodynamics, material science, biochemistry, etc. Among these applications, a fundamental problem is the following elliptic interface problem:

$$-\nabla\cdot(\varepsilon(\mathbf{x})\nabla u(\mathbf{x}))=f(\mathbf{x}),\quad \mathbf{x}\in\Omega\setminus\Gamma, \tag{1}$$

$$[u]=\tau,\quad [\varepsilon u_n]=\sigma\quad \text{on } \Gamma, \tag{2}$$

with the boundary condition

$$u=g\quad \text{on } \partial\Omega. \tag{3}$$

Here, the coefficient $\varepsilon(\mathbf{x})$ is assumed to be discontinuous across the interface Γ, the notation $[u]$ stands for the jump of u across the interface, and u_n represents the outer normal derivative of u.

In the application of electrostatics, u is the potential, ε is the dielectric coefficient, and f is the charge density. In biochemistry, a typical problem is to find the electric potential for a macromolecule immersed in an ionic solvent (e.g. water). The dielectric coefficient is about 2 inside the macromolecule and about 80 in water region [16,17,34]. In material science, the dielectric coefficients are about 1 for air and 12–13 for silicon. The jump conditions usually come from some balance laws across interfaces. For instance, $[u]$ represents the potential difference across a cell membrane, or the pressure difference in an immiscible two-phase flow; and $[\varepsilon u_n]$ is the surface charge in electrostatics. Sometimes, they are equivalent to some singular source terms, like those in fluid-solid interaction, see review articles [7,29].

A closely related problem is the elliptic irregular domain problem [12,13,15,21,39,40], where one embeds an irregular domain into a larger regular domain and treats the original boundary conditions as internal jump conditions.

In these applications, it is important to have an accurate numerical method for both u, the potential, and its gradient, the electric field (or the velocity in fluid problems). The studies of these interface problems have a long history. For body-fitting approaches, we refer readers to [34,48]. For finite element approaches, we refer readers to [6,14,18,32] and references therein.

This paper is concerned with finite difference approaches under structured grids. Such approaches enjoy simplicity and are popular for many practical problems, especially for those with moving interfaces. There are several approaches in this direction. We classify them into three: regularization, dimension un-splitting and dimension splitting approaches.

1.1. Regularization approach

Regularization approach is a smoothing technique applied to the coefficients and singular sources plus a post-processing (if needed). The smoothing by harmonic averaging [4,43,44] for the coefficient $\varepsilon(\mathbf{x})$ can achieve second-order accuracy in one dimension, but drops down to low-order accuracy in high dimensions [4,29,43,45]. It was shown that smoothing technique for coefficients cannot go beyond first-order accuracy [45]. For smooth coefficient cases, the celebrated immersed boundary method [7,41] regularizes the Dirac

source term by moment method. High-order methods can be derived for singular sources based on high-order moment methods [46], provided there is no discontinuity in the coefficient $\varepsilon(\mathbf{x})$.

1.2. Dimension un-splitting approach

These finite difference methods are derived from local Taylor expansion in multi-dimensions. The immersed interface method (IIM) proposed by LeVeque and Li [26] is a second-order method in this class. It requires quadratic approximation on both sides of the interface and a second-order match of jump conditions on the interface. For the latter, IIM obtains full second-order jump conditions by differentiating the prescribed jump conditions along tangential direction up to second order at just one interface point plus the difference of the original second-order equations on the two sides. In two dimensions, its stencil involves six grid points. This is the least number of grid points needed for a second-order scheme for interface problems. However, it is found that the resulting linear system may not be stable as employed by some iterative linear solvers [3,30].

An attempt to determine which grid point should be used in addition to the standard five-point stencil for stability consideration for the Neumann problems on irregular domain was proposed by Fogelson and Kenner [11]. Alternatively, a maximum principle preserving immersed interface method (MIIM) [30] was proposed which enforces the resulting coefficient matrix to be an M-matrix. Its stencil involves all 3^d neighboring grid points in d dimensions. The coefficients of the finite difference schemes are found by solving a constrained optimization in 3^d dimensions. This scheme enjoys stability and second-order accuracy. Moreover, it is compact, has less grid orientation effect, and has good convergence performance by geometric multigrid and algebraic multigrid (AMG) [1–3]. But the scheme involves solving 3^d coefficient equations for each grid point near the interface.

On the other hand, for speed consideration, Li also proposed a fast iterative immersed interface method (FIIM) [28] for elliptic interface problems with piecewise constant coefficients. The method preconditions Eq. (1) before applying IIM in order to take advantage of the standard fast Poisson solver. To do so, an auxiliary unknown (the jump $[u_n]$) and a corresponding interpolation equation are introduced via a weighted least square approach. The GMRES is employed to solve the corresponding Schur complement system. The computational time is essentially linear in the number of unknowns from numerical experiments. For other improvements and applications of IIM, please see the references [8,9,19,20,22–25,27,28,31,33,35,47,51,52].

1.3. Dimension splitting approach

These finite difference methods are derived from Taylor expansions in each dimension. The jump conditions are realized at some nearby interface points or at the intersections of the coordinate axes and the interface. This approach was originally designed for solving elliptic irregular domain problems [38]. In this case, many high-order accurate methods were available [12,13,21,39,40]. The ghost fluid method (GFM) [10] introduced by Fedkiw et al. falls into this dimension splitting approach for hyperbolic interface problems. One main idea there is to smoothly extend the function across over the other side of the interface. It was extended to solve elliptic interface problems by Liu et al. [36]. In [36], only three grid points are used in each dimension. In high dimensions, the jump data $[\varepsilon u_{x_k}]$ in each coordinate direction are projected onto the normal direction $[\varepsilon u_n]$ and tangential direction $[\varepsilon u_t]$. The tangential part is neglected in their method (see formulas (75), (76) of [36]). One may suspect that such GFM can result in zeroth-order accuracy in the maximum norm from the negligence of $[\varepsilon u_t]$. But in practice, it is found that GFM reaches first-order. Indeed, a convergence proof for GFM is provided by Liu and Sideris [37]. In Appendix A, we demonstrate by an interesting numerical example that the truncated error can grow like $O(1/h)$ yet the true error does converge like $O(h)$.

Three higher-order GFMs have been proposed for solving elliptic irregular domain problems [13,50,12]. They are second-order with symmetric coefficients [13], second-order for complex interface problems[50], and fourth-order method [12], respectively. For elliptic interface problems, a high-order method which combines the merits of GFM and FIIM is the explicit-jump immersed interface method (EJIIM) [49], where the auxiliary unknowns are the high-order jumps at the intersections of the interface and the coordinate directions. The interpolation equation for these high-order jumps is derived via a local polynomial on one side and the jump data.

Another high-order method for elliptic interface problems is the decomposed immersed interface method (DIIM) proposed by Berthelsen [5]. The main idea is to decompose the jump data into coordinate directions. The method uses the standard central finite difference scheme for the left-hand side and put all correction terms from jumps to the right-hand side, where high-order one-side interpolation is used on both sides of the interface. The advantage is that the left-hand side is easily to invert. The right-hand side needs a successive correction. However, due to the fact that the right-hand side is not small, a small parameter for the successive under-relaxation is required to reach convergence and thus the convergence is slow.

Another high-order approach in this direction is the matched interface and boundary method (MIB) proposed by Zhou et al. [53]. In each dimension, a high-order finite difference equation using grid data and jump data is derived through the help of fictitious points. In multi dimensions, the jump data $[\varepsilon u_{x_k}]$ in each coordinate direction is expressed in terms of $[\varepsilon u_n]$ and $[\varepsilon u_t]$. The former is the prescribed datum. The latter is obtained by a combination of the prescribed $[u_t]$ and one-side interpolation of nearby grid values. Since this one-side interpolation has to be second-order and would involve many grid points on one-side, it is limited to simple interfaces.

1.4. Present approach: coupling interface method

In the present work, we also take a dimension splitting approach as those of GFM [36,13,50,12], EJIIM [49] and MIB [53]. The new part of our approach is to derive a coupling equation for the principal derivatives to avoid unnecessary one-side interpolation. Our method consists of a first-order version (CIM1) and a second-order version (CIM2).

In one dimension, the CIM1 is derived from a linear approximation on both sides of the interface. The method is extended to high dimensions through a dimension-by-dimension approach. To connect information from each dimension, a coupled equation for the first-order partial derivatives is derived by realizing the jump conditions at the intersections of the interface and the grid segments, where the coupling is done through the expression of the one-side tangential derivative u_t in terms of the first-order partial derivatives. The resulting stencil uses the standard 5 grid points in two dimensions and 7 grid points in three dimensions.

Similarly, the CIM2 is derived from a quadratic approximation on each side of the interface in one dimension. This results in a scheme which expresses $u_{x_kx_k}$ in terms of two grid data from each sides and two low-order jump data on the interface. To connect information from each dimension, a one-side interpolation of tangential derivatives u_t is expressed in terms of $u_{x_jx_j}$ again to reduce the number of interpolation points needed. This will result in a $d\times d$ coupled equation for the principal second-order derivatives $u_{x_kx_k}, k = 1,\ldots,d$. The one-side interpolation is only applied to the cross derivatives. The resulting stencil involves 8 grid points in two dimensions and 12–14 grid points in three dimensions. The method is simple and easy to implement in any dimensions.

To analyze the stability of the resulting linear system for the CIM2, we demonstrate by numerical experiment that its condition number behaves as that of a discrete Laplacian, and is insensitive to the relative location of the interface in a grid cell. Further, we prove the solvability of the coupling equation, provided the curvature κ of the interface satisfies $\kappa h \leqslant \text{Const}$, where h is the mesh size. This can always be achievable if we refine the meshes.

The CIM1 requires that the interface intersects each grid segment (the segment connecting two adjacent grid points) at most once. This is a very mild restriction and is always achievable by refining meshes. The CIM2 requires basically that the interface does not intersect two adjacent grid segments simultaneously. This requirement may not be satisfied at some interface points, where we use the CIM1. In practice, we classify the underlying Cartesian grid points into interiors, normal on-fronts and exceptionals, where a standard central finite difference method, the CIM2, and the CIM1 are adopted, respectively. This method maintains second-order accuracy for most applications due to the fact that usually in d dimensions, the number of normal on-front grid points is $O(h^{1-d})$ and the number of the exceptional points is $O(1)$.

Algebraic multigrid method (AMG) [42] is employed to solve the resulting linear system. Gauss–Seidel is used for the smoother in the AMG. Numerical tests show that the AMG is very stable and fast with this system. The reduction rate for each V-cycle is about 0.1–0.7 in most test cases. The computational time grows essentially linear in the number of unknowns. Convergence tests for both CIM1 and CIM2 are performed.

Numerical results show that they are first-order and second-order accurate, respectively. Comparison studies with other existing second-order methods are performed. It is found that CIM2 produces less absolute errors despite using few stencil grid points. Moreover, we demonstrate by many numerical examples that our method is capable of handling complex interface problems in two and three dimensions.

This paper is organized as follows. Sections 2–4 are the coupling interface method (CIM1 and CIM2) in one, two and general d dimensions, respectively. Section 5 is the numerical tests. It includes convergence tests of the CIM1 and CIM2, a comparison study with other existing interface methods in two and three dimensions. Section 6 is the conclusion. In Appendix A, we show a numerical example to demonstrate that cancellation is responsible to the convergence of GFM; we also provide a proof of the solvability of the coupling equation; and a pseudocode of the CIM for readers' convenience.

2. One dimensional case

We consider the following elliptic interface problem on $[a,b]$ with an interface located at $\hat{x} \in (a,b)$:

$$-(\varepsilon(x)u'(x))' = f(x), \quad x \in (a,b) \setminus \{\hat{x}\}, \tag{1}$$

$$u(a) = u_a, \quad u(b) = u_b,$$

$$[u]_{\hat{x}} = \tau, \quad [\varepsilon u_x]_{\hat{x}} = \sigma. \tag{2}$$

Here, the functions $\varepsilon(x) > 0$ and $f(x)$ are assumed to be smooth on both sides of $\hat{x}$. To derive a finite difference method for the above problem, we partition $[a,b]$ into N subintervals evenly. Let $h = (b-a)/N$, $x_i = a + ih$, $0 \leqslant i \leqslant N$. A grid point x_i is called *on-front* if either $[x_{i-1}, x_i)$ or $[x_i, x_{i+1})$ contains an interface point. Otherwise, it is called an *interior* point. At an interior point x_i, a standard central finite difference scheme is adopted. Namely,

$$-(\varepsilon u')'(x_i) = -\frac{1}{h^2}(\varepsilon_{i+1/2}(u_{i+1} - u_i) - \varepsilon_{i-1/2}(u_i - u_{i-1})) + \mathrm{O}(h^2). \tag{3}$$

For an on-front grid point x_i, we propose a first-order coupling method (CIM1) and a second-order coupling method (CIM2) below.

2.1. CIM1 in one dimension

Our basic assumption is that *there is at most one interface point in each cell*. Suppose there is an interface point $\hat{x}$ situated in the interval $[x_i, x_{i+1})$. Here and after, We shall call the region where x_i is located the Ω^- side, whereas the other Ω^+ side. Let $\alpha := (\hat{x} - x_i)/h$, $0 \leqslant \alpha < 1$, $\beta = 1 - \alpha$ and let us use the abbreviations: $\varepsilon^- := \varepsilon(\hat{x}^-), \varepsilon^+ := \varepsilon(\hat{x}^+)$. In the interval $[x_i, x_{i+1})$, like the approach of the ghost fluid method [36], we approximate u by a piecewise linear function

$$u(x) \approx \begin{cases} u^-(x) := u_i + (u')^-_{i+1/2}(x - x_i) & \text{for } x_i \leqslant x < \hat{x}, \\ u^+(x) := u_{i+1} + (u')^+_{i+1/2}(x - x_{i+1}) & \text{for } \hat{x} < x < x_{i+1}. \end{cases}$$

Here, $(u')^{\pm}_{i+1/2}$ represents the approximation slopes of u in $\Omega^{\pm}$ respectively in the interval $[x_i, x_{i+1})$ (see Fig. 1).

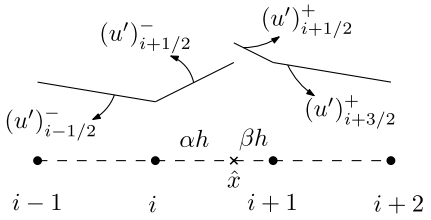


Fig. 1. Linear approximation of the CIM1: u is approximated by two linear functions in the cell $[x_i, x_{i+1})$ in which an interface point $\hat{x}$ is situated.

From the jump condition (2), we get

$$(u_{i+1} - \beta h (u')^+_{i+1/2}) - (u_i + \alpha h (u')^-_{i+1/2}) \approx \tau,$$
$$\varepsilon^+ (u')^+_{i+1/2} - \varepsilon^- (u')^-_{i+1/2} \approx \sigma.$$

This yields

$$\begin{aligned} (u')^-_{i+1/2} &= \frac{1}{h}\left(\bar{\rho}^+ (u_{i+1} - u_i) - \bar{\rho}^+ \tau - \beta h \frac{\sigma}{\bar{\varepsilon}}\right) + \mathrm{O}(h), \\ (u')^+_{i+1/2} &= \frac{1}{h}\left(\bar{\rho}^- (u_{i+1} - u_i) - \bar{\rho}^- \tau + \alpha h \frac{\sigma}{\bar{\varepsilon}}\right) + \mathrm{O}(h), \end{aligned} \tag{4}$$

where $\bar{\varepsilon} = \alpha\varepsilon^+ + \beta\varepsilon^-$, $\bar{\rho}^\pm = \varepsilon^\pm/\bar{\varepsilon}$. Similarly, we can define $(u')^-_{i-1/2}$ and $(u')^+_{i-1/2}$ if there is an interface in $[x_{i-1}, x_i)$. Otherwise, $(u')^-_{i-1/2} := (u_i - u_{i-1})/h$. We recall that the minus sign in $(u')^-_{i-1/2}$ represents the Ω^- side where x_i is situated. With these, we approximate (1) by

$$-(\varepsilon u')'(x_i) = -\frac{1}{h}\varepsilon_i ((u')^-_{i+1/2} - (u')^-_{i-1/2}) + \mathrm{O}(1). \tag{5}$$

The O(1) truncation error here produces an O(h) global error [26].

Notice that CIM1 allows interface points appear on both sides of x_i. Indeed, the derivation of $(u')^-_{i-1/2}$ is decoupled with that of $(u')^+_{i+1/2}$. This gives flexibility to use CIM1.

Below is a pseudocode of CIM1. In this code, $s = 1$ (resp. -1) represents the case where the interface $\hat{x}$ lies to the right (resp. left) of the grid point x_i.

Algorithm 1. CIM1 in one dimension

procedure CIM1-1D (x_i, h)
  for $s = -1, 1$ **do**
    $(\bar{D}_s u_i, \hat{x}_s, \bar{\rho}^\pm_s, \bar{\varepsilon}_s, \beta_s, \gamma_s) \leftarrow$ 1stDerivative-1D(x_i, h, s)
    if $\gamma = 1$ **then**
      $(\tau_s, \sigma_s) \leftarrow$ the jump data locate at $\hat{x}_s$ from x_i side to the other side
      $\bar{J}_s \leftarrow -(s\bar{\rho}^+_s \tau_s + \beta_s h \sigma_s/\bar{\varepsilon}_s)$
    else
      $\bar{J}_s \leftarrow 0$
    end If
    $u'_{i+s/2} \leftarrow \frac{1}{h}(\bar{D}_s u_i + \bar{J}_s)$
  end For
  $-(\varepsilon u')'(x_i) \leftarrow -\frac{1}{h}\varepsilon(x_i)(u'_{i+1/2} - u'_{i-1/2})$
end procedure

Algorithm 2. First-order derivative in one dimension

function $(\bar{D}u_i, \hat{x}, \bar{\rho}^\pm, \bar{\varepsilon}, \beta, \gamma) =$ 1stDerivative-1D(x_i, h, s)
  if There is an interface point $\hat{x}$ in x_i to $x_i + sh$ **then**
    $\gamma \leftarrow 1, \alpha \leftarrow |\hat{x}_s - x_i|/h, \beta \leftarrow 1 - \alpha.$
    $\varepsilon^- \leftarrow \lim_{t \leftarrow 0^-} \varepsilon(\hat{x}_s + st), \varepsilon^+ \leftarrow \lim_{t \leftarrow 0^+} \varepsilon(\hat{x}_s + st).$
  else
    $\gamma \leftarrow 0, \alpha \leftarrow 1, \beta \leftarrow 0, \varepsilon^\pm \leftarrow \varepsilon(x_i), \hat{x} = x_i + sh$
  end if
  $\bar{\varepsilon} \leftarrow \alpha\varepsilon^+ + \beta\varepsilon^-$
  $\bar{\rho}^\pm \leftarrow \varepsilon^\pm/\bar{\varepsilon}$
  $\bar{D}u_i \leftarrow \bar{s}\rho^+(u(x_i + sh) - u(x_i))$
end function

2.2. CIM2 in one dimension

Suppose there is an interface point $\hat{x}$ situated in $[x_i, x_{i+1})$. To derive CIM2, we shall assume there is no other interface points inside $[x_{i-1}, x_{i+2}]$. As in CIM1, let us call the side where x_i is localed the Ω^- side. We assume that the interface point $\hat{x}$ sits in $[x_i, x_{i+1})$. See Fig. 2.

At x_i, we introduce the following bias finite differencing:

$$-(\varepsilon u')'(x_i) = -\frac{1}{h^2}((\varepsilon_i - \varepsilon_{i-1})(u_i - u_{i-1}) + \varepsilon_i u_i'') + \mathrm{O}(h) \tag{6}$$

$$:= -\frac{1}{h^2}(D^{(1)}\varepsilon_i \cdot D^{(1)}u_i + \varepsilon_i u_i'') + \mathrm{O}(h). \tag{7}$$

where $D^{(1)}\varepsilon_i$ denotes for $\varepsilon_i - \varepsilon_{i-1}$. To maintain $\mathrm{O}(h)$ truncation error at x_i, we need a first-order approximation for the term u_i''. The idea is to approximate u by quadratic functions on both sides of $\hat{x}$. These involve six coefficients. They are determined by the two jump conditions and realizing u at x_{i-1}, x_i, x_{i+1} and x_{i+2}. To be precise, by Taylor expansion, let us express u on both sides of $\hat{x}$ as

$$\begin{cases} u^-(x) = u_i + \left(\frac{u_i - u_{i-1}}{h} + \frac{1}{2}hu_i''\right)(x - x_i) + \frac{1}{2}u_i''(x - x_i)^2 + \mathrm{O}(h^3), & \text{for } x \in [x_{i-1}, \hat{x}), \\ u^+(x) = u_{i+1} + \left(\frac{u_{i+2} - u_{i+1}}{h} - \frac{1}{2}hu_{i+1}''\right)(x - x_{i+1}) + \frac{1}{2}u_{i+1}''(x - x_{i+1})^2 + \mathrm{O}(h^3), & \text{for } x \in (\hat{x}, x_{i+2}), \end{cases}$$

where the conditions $u^-(x_{i-1}) = u_{i-1}, u^-(x_i) = u_i, u^+(x_{i+1}) = u_{i+1}$ and $u^+(x_{i+2}) = u_{i+2}$ have been used. From the jump conditions:

$$u^+(\hat{x}) - u^-(\hat{x}) = \tau, \quad (\varepsilon u')^+(\hat{x}) - (\varepsilon u')^-(\hat{x}) = \sigma, \tag{8}$$

we can solve for u_i'', u_{i+1}'' as

$$u_i'' = \frac{1}{h^2}(L^{(1)}u_i + J_i) + \mathrm{O}(h), \tag{9}$$

$$u_{i+1}'' = \frac{1}{h^2}(L^{(-1)}u_{i+1} + J_{i+1}) + \mathrm{O}(h), \tag{10}$$

where

$$L^{(1)}u_i := a_{i,-1}u_{i-1} + a_{i,0}u_i + a_{i,1}u_{i+1} + a_{i,2}u_{i+2},$$

$$L^{(-1)}u_{i+1} := a_{i+1,-2}u_{i-1} + a_{i+1,-1}u_i + a_{i+1,0}u_{i+1} + a_{i+1,1}u_{i+2},$$

$$J_i := -(1 + 2\beta)\rho^+\tau - (\beta + \beta^2)\frac{\sigma h}{\hat{\varepsilon}},$$

$$J_{i+1} := (1 + 2\alpha)\rho^-\tau - (\alpha + \alpha^2)\frac{\sigma h}{\hat{\varepsilon}}.$$

and the corresponding coefficients are listed as follows:

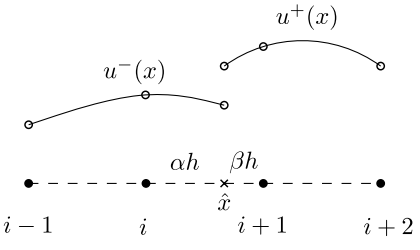


Fig. 2. Quadratic approximation of CIM2: u is approximated by two quadratic functions in $[x_{i-1}, x_{i+2}]$.

$$\hat{\varepsilon} = (\beta+\beta^2)\left(\frac{1}{2}+\alpha\right)\varepsilon^- + (\alpha+\alpha^2)\left(\frac{1}{2}+\beta\right)\varepsilon^+,$$
$$\rho^{\pm} = \frac{\varepsilon^{\pm}}{\hat{\varepsilon}},$$
$$a_{i,-1} = (\beta+\beta^2)\rho^- + \alpha(1+2\beta)\rho^+,$$
$$a_{i,0} = -(\beta+\beta^2)\rho^- - (1+\alpha)(1+2\beta)\rho^+,$$
$$a_{i,1} = (1+\beta)^2\rho^+,$$
$$a_{i,2} = -\beta^2\rho^+,$$
$$a_{i+1,1} = (\alpha+\alpha^2)\rho^+ + \beta(1+2\alpha)\rho^-,$$
$$a_{i+1,0} = -(\alpha+\alpha^2)\rho^+ - (1+\beta)(1+2\alpha)\rho^-,$$
$$a_{i+1,-1} = (1+\alpha)^2\rho^-,$$
$$a_{i+1,-2} = -\alpha^2\rho^-.$$

Notice that the coefficients a's are dimensionless. Furthermore,

$$\min\{\varepsilon^-,\varepsilon^+\} \leqslant \hat{\varepsilon} \leqslant (1+2\alpha\beta)\max\{\varepsilon^-,\varepsilon^+\} \leqslant \frac{3}{2}\max\{\varepsilon^-,\varepsilon^+\}.$$

Combining (7) and (9), we get

$$-(\varepsilon u')'(x_i) = -\frac{1}{h^2}(D^{(1)}\varepsilon_i D^{(1)}u_i + \varepsilon_i(L^{(1)}u_i + J_i)) + \mathrm{O}(h). \tag{11}$$

We can get a similar formula for $-(\varepsilon u')'(x_{i+1})$. Notice that the local truncation error is $\mathrm{O}(h)$ at these two on-front grid points. It contributes $\mathrm{O}(h^2)$ global error in the maximal norm.

By introducing an orientation indicator, we can unify the above finite difference formulae (9) and (10) into just one formula. Let x_i be an on-front point and $\hat{x}$ be the nearest interface point (see Fig. 3). As before, we shall call the region where x_i is situated the Ω^- region, and the other side the Ω^+ region. We define $\varepsilon^{\pm}$ to be the limit of $\varepsilon(x)$ at $\hat{x}$ from $\pm$ sides. The jump $[u]$ is defined to be the difference of $u(x)$ from $+$ side to $-$ side, that is $[u] = u^+ - u^-$. At x_i, let us define an orientation indicator s to be

$$s = \begin{cases} 1 & \text{if } \hat{x} \in [x_i, x_{i+1}), \\ -1 & \text{if } \hat{x} \in [x_{i-1}, x_i), \\ 0 & \text{if } x_i \text{ is an interior point.} \end{cases}$$

Let us define the following parameters:

$$\begin{cases} \alpha = |\hat{x} - x_i|/h, \\ \beta = 1-\alpha, \end{cases}$$
$$\begin{cases} \hat{\varepsilon} = \left(\frac{1}{2}+\alpha\right)(\beta+\beta^2)\varepsilon^- + \left(\frac{1}{2}+\beta\right)(\alpha+\alpha^2)\varepsilon^+, \\ \rho^{\pm} = \varepsilon^{\pm}/\hat{\varepsilon}, \end{cases}$$
$$\begin{cases} a_{-s} = (\beta+\beta^2)\rho^- + \alpha(1+2\beta)\rho^+, \\ a_0 = -(\beta+\beta^2)\rho^- - (1+\alpha)(1+2\beta)\rho^+, \\ a_s = (1+\beta)^2\rho^-, \\ a_{2s} = -\beta^2\rho^-. \end{cases} \tag{12}$$

When $s = 0$, the parameters are defined to be

$$\alpha = 1, \quad \beta = 0, \quad \varepsilon^+ = \varepsilon^- = \varepsilon(x_i).$$

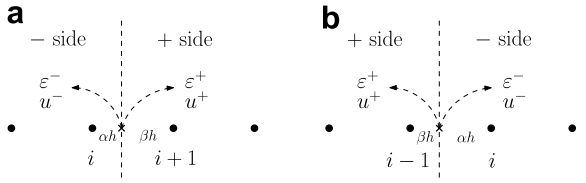


Fig. 3. Orientation indicator s is defined to be 1 if the interface $\hat{x}$ lies to the right of x_i (left subfigure) and -1 if the interface lies to the left of x_i (right subfigure): (a) $s = 1$ and (b) $s = -1$.

We extend the definitions of $D^{(1)}$ and $L^{(1)}$ to the follows:

$$D^{(s)}u_i = \frac{1}{2}((1-s)u_{i+1} + 2su_i - (1+s)u_{i-1}), \tag{13}$$
$$= \begin{cases} \frac{u_{i+1}-u_{i-1}}{2} & \text{if } s = 0, \\ u_i - u_{i-1} & \text{if } s = 1, \\ u_{i+1} - u_i & \text{if } s = -1, \end{cases}$$
$$L^{(s)}u_i = \begin{cases} a_{-s}u_{i-s} + a_0u_i + a_su_{i+s} + a_{2s}u_{i+2s} & \text{if } s = \pm 1, \\ u_{i-1} - 2u_i + u_{i+1} & \text{if } s = 0, \end{cases} \tag{14}$$

We also define the jump operator at $\hat{x}$ as

$$J_{\hat{x}} = -\left((1+2\beta)\rho^+[u] + s(\beta+\beta^2)h\frac{[\varepsilon u']}{\hat{\varepsilon}}\right). \tag{15}$$

With these notations, we can express u_i'' and $-(\varepsilon u')'$ as

$$u_i'' = \frac{1}{h^2}(L^{(s)}u_i + J_{\hat{x}}) + \mathrm{O}(h^{2-|s|}), \tag{16}$$
$$-(\varepsilon u')'(x_i) = -\frac{1}{h^2}(D^{(s)}\varepsilon_i \cdot D^{(s)}u_i + \varepsilon_i(L^{(s)}u_i + J_{\hat{x}})) + \mathrm{O}(h^{2-|s|}). \tag{17}$$

Here are the pseudocode of CIM2:

Algorithm 3. CIM2 in one dimension

procedure CIM2-1D (x_i,h)
  $(L^{(s)}u_i, \hat{x}, \rho, \hat{\varepsilon}, \beta, s) \leftarrow$ 2NDDERIVATIVE-1D(x_i, h)
  if $s \neq 0$ **then**
    $(\tau, \sigma) \leftarrow$ the jump data locate at $\hat{x}_s$ from x_i side to the other side
    $J \leftarrow -((1+2\beta)\tau + s(\beta+\beta^2)h\sigma/\hat{\varepsilon})$
  else
    $J = 0$
  end if
  $D^{(s)}u_i \leftarrow \frac{1}{2}((1-s)u_{i+1} + 2su_i - (1+s)u_{i-1})$
  $D^{(s)}\varepsilon_i \leftarrow \frac{1}{2}((1-s)\varepsilon_{i+1} + 2s\varepsilon_i - (1+s)\varepsilon_{i-1})$
  $u_i'' \leftarrow \frac{1}{h^2}(L^{(s)}u_i + J)$
  $-(\varepsilon u')'(x_i) \leftarrow -\frac{1}{h^2}(D^{(s)}\varepsilon_i D^{(s)}u_i + \varepsilon_i u_i'')$
end procedure

Algorithm 4. Second-order derivative in one dimension

function $(L^{(s)}u_i, \hat{x}, \rho, \hat{\varepsilon}, s) = 2\text{ND}\textsc{Derivative}\text{-1D}(x_i, h)$
  if There is an interface point $\hat{x}$ in $[x_i - h, x_i + h)$ **then**
    if $\hat{x} \geqslant x_i$ **then**
      $s \leftarrow 1$, ▷ There is no other interface point in $[x_{i-1}, x_{i+2})$
    else
      $s \leftarrow -1$, ▷ There is no other interface point in $[x_{i-2}, x_{i+2})$
    end if
    $\alpha \leftarrow |\hat{x} - x_i|/h, \beta \leftarrow 1 - \alpha$.
    $\varepsilon^- \leftarrow \lim_{t \leftarrow 0^-} \varepsilon(\hat{x} + st), \varepsilon^+ \leftarrow \lim_{t \leftarrow 0^+} \varepsilon(\hat{x} + st)$.
    $\hat{\varepsilon} \leftarrow (\beta + \beta^2)(\frac{1}{2} + \alpha)\varepsilon^- + (\alpha + \alpha^2)(\frac{1}{2} + \beta)\varepsilon^-$,
    $\rho^{\pm} \leftarrow \varepsilon^{\pm}/\hat{\varepsilon}$
    $a_{-s} \leftarrow (\beta + \beta^2)\rho^- + \alpha(1 + 2\beta)\rho^+$
    $a_s \leftarrow (1 + \beta)^2\rho^+$, $a_{2s} \leftarrow -\beta^2\rho^+$
    $a_0 \leftarrow -(a_{-s} + a_s + a_{2s})$
    $L^{(s)}u_i \leftarrow (a_{-s}u(x_i - sh) + a_0u(x_i) + a_su(x_i + sh) + a_{2s}u(x_i + 2sh))$
  else
    $s \leftarrow 0, L^{(0)}u_i \leftarrow (u(x_i - h) - 2u(x_i) + u(x_i + h)), J_{\hat{x}} \leftarrow 0$
  end if
end function

2.3. Stability issue for CIM2 in one dimension

We perform the following numerical investigation to analyze the stability of the coefficient matrix corresponding to the CIM2 in one dimension. We choose the domain to be $[0, 1]$. Assuming there is only one interface point $\hat{x}$ located in $(1/2, 1/2 + h)$. We shall vary the coefficient $\alpha := (\hat{x} - 1/2)/h$ from 0.1 to 0.9 with increment 0.1. The coefficient $\varepsilon(x)$ is assumed to be piecewise constants with $\varepsilon^- = 2$ and $\varepsilon^+ = 80$. The number of grid points N varies from 20 to 200 with increment 2. We use MATLAB to compute the eigenvalues and the condition numbers of the coefficient matrix A. The result is shown in Fig. 4. Next, we fix $\alpha = 0.5$ and $\varepsilon^- = 2$

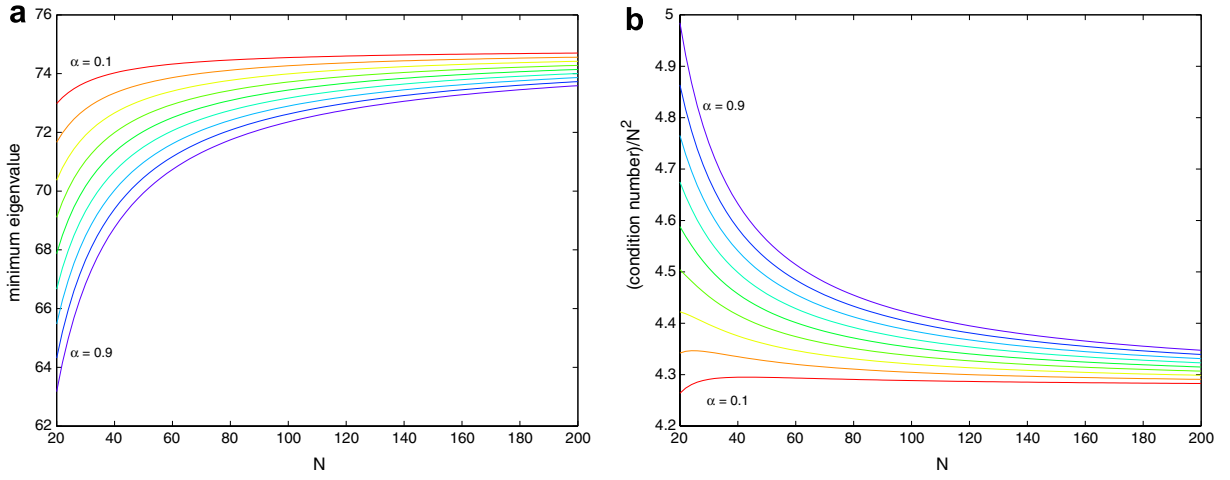


Fig. 4. (a) The minimal eigenvalue $\lambda_{\min}(N, \alpha)$ of the coefficient matrix for CIM2 is insensitive to N and α for large N. (b) The scaled condition number $\text{cond}A(N, \alpha)/N^2$ of the coefficient matrix A for CIM2. It is insensitive to N and α for large N. Here, $\varepsilon^- = 2$ and $\varepsilon^+ = 80$.

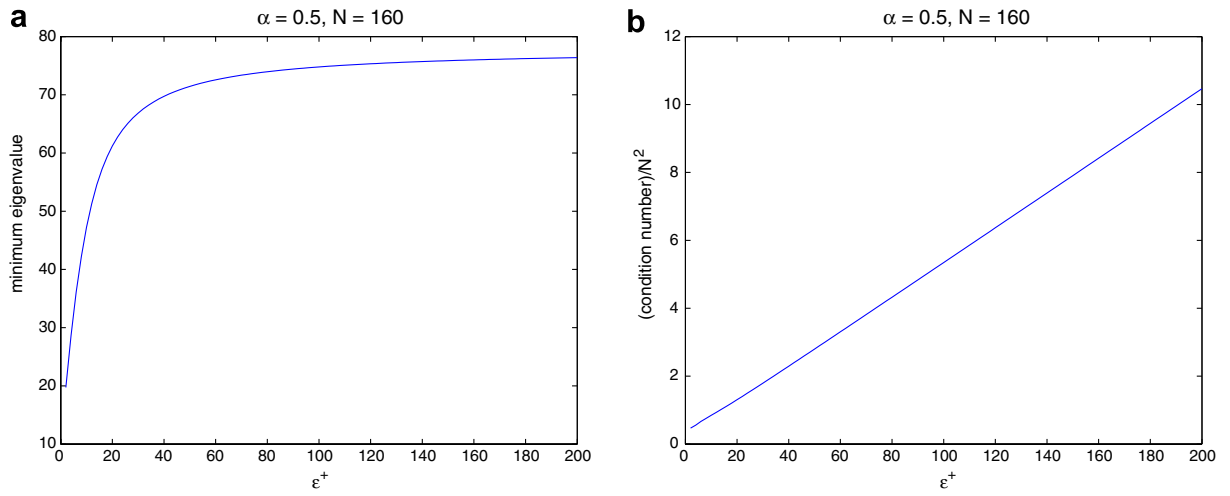


Fig. 5. The minimal eigenvalue (left) and the scaled condition number cond A/N^2 of the coefficient matrix of CIM2 with fixed $\alpha = 0.5$, $\varepsilon^- = 2$ and varying ε^+ from 2 to 200. (a) Minimal eigenvalue $\lambda_{\min}$ as a function of ε^+. (b) Scaled condition number cond A/N^2 as a function of ε^+.

and vary ε^+ from 2 to 200 with increment 2. The corresponding minimal eigenvalue and the scaled condition number are shown in Fig. 5. We summarize the results as the follows.

1. All eigenvalues are positive.
2. The minimal eigenvalues $\lambda_{\min}(N,\alpha)$ are insensitive to N and α for $N \geqslant 100$, see Fig. 4a.
3. The scaled condition number $\mathrm{cond}(A(N,\alpha))/N^2$ is asymptotically constant for large N as that of a standard discrete Laplacian, and is insensitive to α. See Fig. 4b.
4. The minimal eigenvalue is insensitive to the contrast of ε. See Fig. 5a.
5. The scaled condition number increases linearly in ε^+. See Fig. 5b.

In other words, the stability property of the coefficient matrix corresponding to CIM2 is the same as that of a discrete Laplacian.

3. Two dimensional cases

In two dimensions, let us assume our domain $\Omega = [0,1] \times [0,1]$. We partition $[0,1]$ into N subintervals. The mesh size $h := 1/N$. Let $x_i = ih, y_j = jh, 0 \leqslant i,j \leqslant N$. Let $\gamma_{i+1/2,j}$ (resp. $\gamma_{i,j+1/2}$) denote the number of intersections of the interface Γ and the grid segment $[x_i, x_{i+1}) \times \{y_j\}$ (resp. $\{x_i\} \times [y_j, y_{j+1})$). We assume $\gamma_{i+1/2,j}, \gamma_{i,j+1/2} \leqslant 1$ for all $0 \leqslant i,j \leqslant N-1$. This is always achievable by refining meshes. A grid point (x_i, y_j) is called *interior* if none of its four neighboring segments intersects Γ. Otherwise, we call it *on-front*. For an interior grid point, a standard central finite difference method is adopted. Below, we shall illustrate CIM discretization at an on-front grid point (x_i, y_j). As before, we shall call the region where (x_i, y_j) is situated the Ω^- region, whereas the other the Ω^+ region. The approximate solution of u in Ω^- will be denoted by u^-.

3.1. CIM1 in two dimensions

We first perform the following finite differencing:

$$-((\varepsilon u_x)_x + (\varepsilon u_y)_y)(x_i, y_j) = -\frac{\varepsilon_{i,j}}{h}((u_x^-)_{i+1/2,j} - (u_x^-)_{i-1/2,j} + (u_y^-)_{i,j+1/2} - (u_y^-)_{i,j-1/2}) + \mathrm{O}(1).$$

Here, the superscript $-$ means that the quantity is in the Ω^- region. Our goal is to provide a first-order approximation to $(u_x^-)_{i\pm1/2,j}$ and $(u_y^-)_{i,j\pm1/2}$. We shall only illustrate the discretization for $(u_x^-)_{i+1/2,j}$. Other cases are duplication. We assume $\gamma_{i+1/2,j}=1$, otherwise $(u_x^-)_{i+1/2,j}$ is approximated by the central finite difference. There are two sub cases in this case: (i) $\gamma_{i,j+1/2}=0$, (ii) $\gamma_{i,j+1/2}\neq 0$. The corresponding intersection points are denoted by p and q. See Fig. 6.

The discretization procedure for $(u_x^-)_{i+1/2,j}$ consists of the following three steps.

- Dimension-by-dimension discretization. We adopt the one-dimensional formula (4) on the segment $[x_i,x_{i+1})\times\{y_j\}$. Thus, up to an error $O(h)$,

$$(u_x^-)_{i+1/2,j}\approx\frac{1}{h}\left(\bar\rho_p^+(u_{i+1,j}-u_{i,j})-\bar\rho_p^+[u]_p-\beta_p h\frac{[\varepsilon u_x]_p}{\bar\varepsilon_p}\right). \tag{18}$$

Here, $\beta_p,\bar\varepsilon_p:=(1-\beta_p)\varepsilon_p^+ +\beta_p\varepsilon_p^-,\bar\rho_p^+:=\varepsilon_p^+/\bar\varepsilon_p$, etc. are the same coefficients in (4) with respect to the intersection point p.

- Decomposition of jump data. In (18), the jump datum $[\varepsilon u_x]_p$ is not available from the prescribed jump condition, therefore we decompose it into tangential and normal directions of Γ at p:

$$[\varepsilon u_x]_p=[\varepsilon u_n]_p n_p^x+[\varepsilon u_t]_p t_p^x=[\varepsilon u_n]_p n_p^x+(\varepsilon_p^+[u_t]_p+(\varepsilon_p^+-\varepsilon_p^-)(u_t^-))t_p^x. \tag{19}$$

Here, (t_p^x,t_p^y) and (n_p^x,n_p^y) the unit tangential and normal vectors of Γ at p;u_n and u_t are the normal and tangential derivatives of u at p;$(u_t^-)_p$ means the limit of u_t at p from Ω^- side. The quantities $[\varepsilon u_n]_p=\sigma(p)$ and $[u_t]_p=\tau_t(p)$ are obtained from the prescribed jump data. But $(u_t^-)_p$ is not available. To discretize $(u_t^-)_p$, we perform the following step.

- Coupling and one-side interpolation. First, we express $(u_t^-)_p$ by $(u_x^-)_p t_p^x+(u_y^-)_p t_p^y$. Next, we approximate $(u_x^-)_p$ and $(u_y^-)_p$ by the following formulae:
  - $(u_x^-)_p\approx(u_x)_{i+1/2,j}$
  - $(u_y^-)_p\approx\begin{cases}(u_{i,j+1}-u_{i,j})/h & \text{if } \gamma_{i,j+1/2}=0 \text{ (Case(i))},\\ (u_y)_{i,j+1/2} & \text{if } \gamma_{i,j+1/2}=1 \text{ (Case(ii))}.\end{cases}$

With this, $(u_t^-)_p$ can be approximated by

$$(u_t^-)_p\approx(u_x^-)_{i+1/2,j}\cdot t_p^x+(\gamma_{i,j+1/2})(u_y^-)_{i,j+1/2}\cdot t_p^y+\frac{1}{h}\bar T_x u_{i,j},$$

where

$$\bar T_x u_{i,j}=\begin{cases}0 & \text{if } \gamma_{i,j+1/2}=1,\\ (u_{i,j+1}-u_{i,j})t_p^y & \text{if } \gamma_{i,j+1/2}=0.\end{cases}$$

We plug this formula back to (18). For Case (i), we arrive the following equation for $(u_x^-)_{i+1/2,j}$:

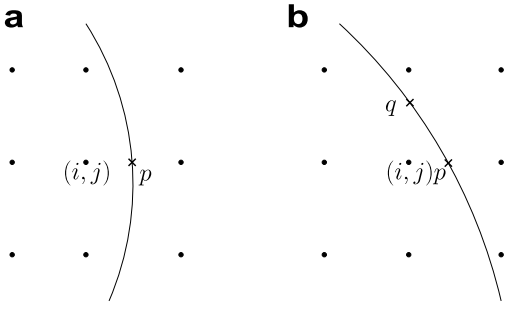


Fig. 6. When the interface intersects the x-axis, there are two subcases: either (i) $\gamma_{i,j+1/2}=0$, or (ii) $\gamma_{i,j+1/2}\neq 0$: (a) Case (i) and (b) Case (ii).

$$(1-\bar{b}_p t_p^x t_p^x)(u_x^-)_{i+1/2,j} = \frac{1}{h}\left(\bar{\rho}_p^+(u_{i+1,j}-u_{i,j}) + \bar{b}_p t_p^x \bar{T}_x u_{i,j} + \bar{J}_p\right), \tag{20}$$

where

$$\bar{b}_p = -\beta_p(\bar{\rho}_p^+ - \bar{\rho}_p^-),$$
$$\bar{J}_p = -\bar{\rho}_p^+[u]_p - \beta_p h\left(\frac{[\varepsilon u_n]_p}{\bar{\varepsilon}_p} n_p^x + \bar{\rho}_p^+[u_t]t_p^x\right).$$

For Case (ii), we obtain a 2×2 coupling equations for $(u_x^-)_{i+1/2,j}$ and $(u_y^-)_{i,j+1/2}$:

$$\bar{M}\begin{bmatrix}(u_x^-)_{i+1/2,j}\\ (u_y^-)_{i,j+1/2}\end{bmatrix} = \frac{1}{h}\begin{bmatrix}\bar{\rho}_p^+(u_{i+1,j}-u_{i,j}) + \bar{b}_p t_p^x \bar{T}_x u_{i,j} + \bar{J}_p\\ \bar{\rho}_q^+(u_{i,j+1}-u_{i,j}) + \bar{b}_q t_q^y \bar{T}_y u_{i,j} + \bar{J}_q\end{bmatrix},$$

where

$$\bar{M} = \begin{bmatrix}1-\bar{b}_p t_p^x t_p^x & -\bar{b}_p t_p^x t_p^y\\ -\bar{b}_q t_q^x t_q^y & 1-\bar{b}_q t_q^y t_q^y\end{bmatrix}.$$

By inverting $\bar{M}$, we obtain a finite difference approximation to $(u_x^-)_{i+1/2,j}$ and $(u_y^-)_{i,j+1/2}$. Similarly, we can derive a coupling equations for $(u_x^-)_{i-1/2,j}$ and $(u_y^-)_{i,j-1/2}$ if both $\gamma_{i-1/2,j},\gamma_{i,j-1/2}=1$.

Remarks.

- Notice that in our derivation, the equations for $(u_x^-)_{i+1/2,j}, (u_y^-)_{i,j+1/2}$ and the equations for $(u_x^-)_{i-1/2,j}$, $(u_y^-)_{i,j-1/2}$ are decoupled.
- In case (i) for $(u_y^-)_p$, we may also approximate it by $(u_{i,j+1}-u_{i,j-1})/(2h)$ if $\gamma_{i,j-1/2}=\gamma_{i,j+1/2}=0$.
- The CIM1 is essentially equivalent to the ghost fluid method [36] except the treatment of the term $[\varepsilon u_t]$ in formula (19). In GFM, this term is neglected. The effect of this neglect is shown in Appendix A.

3.2. CIM2 in two dimensions

There are some restrictions to a grid point to apply the CIM2. Roughly speaking, the basic requirement is to allow the one-dimensional formula (9) to be applicable in both x and y directions there. We call such a grid point a normal on-front grid point. A precise definition will be given in Section 4.

Let (x_i, y_j) be a normal on-front grid point. There are also two sub cases here: either Γ intersects one grid segment or two grid segments, just like the two sub cases in CIM1. See Fig. 7.

Case 1. *The interface intersects the grid segment in the x-direction only.* See Fig. 7, (a). In this case, we approximate $(\varepsilon u_y)_y(x_i,y_j)$ by a standard central finite difference. The term $(\varepsilon u_x)_x(x_i,y_j)$ is approximated by

$$-(\varepsilon u_x)_x(x_i,y_j) = -\frac{1}{h^2}((D_x^{(1)}\varepsilon)_{i,j}(D_x^{(1)}u)_{i,j} + \varepsilon_{i,j}h^2(u_{xx})_{i,j}) + O(h),$$

where the operators $D_x^{(1)}$ is defined by (13) in the x-direction. Our goal is to derive a first-order finite difference approximation for $(u_{xx})_{i,j}$. Below, let us drop the sub-index (i,j) from $(u_{xx})_{i,j}$ for notational simplicity. The discretization procedure for u_{xx} at (x_i,y_j) consists of the following three steps.

- Dimension-by-dimension discretization. We use the one-dimension formula for CIM2 (16) to get

$$u_{xx} = \frac{1}{h^2}\left(L_x^{(1)}u - (1+2\beta_p)\rho_p^+[u]_p - (\beta_p+\beta_p^2)h\frac{[\varepsilon u_x]_p}{\hat{\varepsilon}_p}\right) + O(h),$$

where $L_x^{(1)}$ is the extension of (14) in the x-direction and the coefficients are defined in (12) with respect to the intersection p.

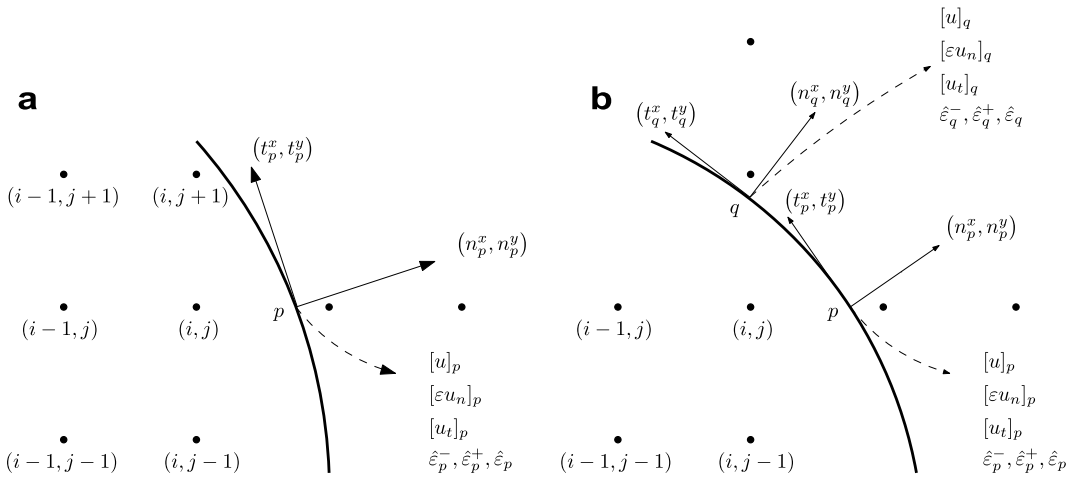


Fig. 7. Normal on-front points in two dimensions: (a) Case 1 and (b) Case 2.

- Decomposition of the one-dimensional jump data. We decompose the jump $[\varepsilon u_x]_p$ into normal and tangential directions:

$$[\varepsilon u_x]_p = [\varepsilon u_n]_p \cdot n_p^x + \left(\varepsilon_p^+ [u_t]_p + (\varepsilon_p^+ - \varepsilon_p^-)(u_t^-)_p\right) \cdot t_p^x.$$

- Coupling and one-side interpolation. We express $(u_t^-)_p$ in terms of u_{xx} again. The cross derivative is approximated by one-side interpolation.

$$\begin{aligned}(u_t^-)_p &= (u_x^-)_p t_p^x + (u_y^-)_p t_p^y \\ &= \left(\frac{u_{i,j} - u_{i-1,j}}{h} + \left(\frac{1}{2} + \alpha_p\right) h u_{xx}\right) t_p^x + \left((1+\alpha_p)\frac{u_{i,j+1} - u_{i,j-1}}{2h} - \alpha_p \frac{u_{i-1,j+1} - u_{i-1,j-1}}{2h}\right) t_p^y + \mathrm{O}(h^2) \\ &:= \frac{1}{h} T_x u + h\left(\frac{1}{2} + \alpha_p\right) t_p^x u_{xx} + \mathrm{O}(h^2).\end{aligned}$$

By plugging $(u_t^-)_p$ into the formula for u_{xx}, we arrive an equation for u_{xx}:

$$\left(1 + \left(\frac{1}{2} + \alpha_p\right) b_p t_p^x t_p^x\right) u_{xx} = \frac{1}{h^2}(L_x^{(1)} u + b_p t_p^x T_x u + J_p),$$

where

$$\begin{aligned} b_p &= -(\beta_p + \beta_p^2)(\rho_p^+ - \rho_p^-), \\ J_p &= -\left((1+2\beta_p)\rho_p^+ [u]_p + (\beta_p + \beta_p^2) h \left(\frac{[\varepsilon u_n]}{\hat{\varepsilon}_p} n_p^x + \rho_p^+ [u_t] t_p^x\right)\right). \end{aligned}$$

Case 2. *The interface intersects grid segments in both x and y-directions*. See Fig. 7, Case (2). The discretization formulae for u_{xx} and u_{yy} at (i,j) are derived by the following three steps:

- Dimension-by-dimension discretization: We apply the one-dimensional formula (16) in both x and y directions to get

$$u_{xx} = \frac{1}{h^2}\left(L_x^{(1)}u - (1+2\beta_p)[u]_p - (\beta_p+\beta_p^2)h\frac{[\varepsilon u_x]_p}{\hat{\varepsilon}_p}\right) + \mathrm{O}(h),$$
$$u_{yy} = \frac{1}{h^2}\left(L_y^{(1)}u - (1+2\beta_q)[u]_q - (\beta_q+\beta_q^2)h\frac{[\varepsilon u_y]_q}{\hat{\varepsilon}_q}\right) + \mathrm{O}(h).$$

Here, the coefficients are defined in (12) with respect to the intersections p and q.

- Decomposition of the jump data:

$$[\varepsilon u_x]_p = [\varepsilon u_n]_p n_p^x + (\varepsilon_p^+[u_t]_p + (\varepsilon_p^+ - \varepsilon_p^-)(u_t^-)_p)(t_p^x),$$
$$[\varepsilon u_y]_q = [\varepsilon u_n]_q n_q^y + (\varepsilon_q^+[u_t]_q + (\varepsilon_q^+ - \varepsilon_q^-)(u_t^-)_q)(t_q^y).$$

- Coupling and one-side interpolation:

$$\begin{aligned}(u_t^-)_p &= \left(\frac{u_{i,j}-u_{i-1,j}}{h} + \left(\frac{1}{2}+\alpha_p\right)hu_{xx}\right)t_p^x \\ &\quad + \left((1+\alpha_p)\frac{u_{i,j}-u_{i,j-1}}{h} - \alpha_p\frac{u_{i-1,j}-u_{i-1,j-1}}{h} + \frac{1}{2}hu_{yy}\right)t_p^y + \mathrm{O}(h^2) \\ &:= \frac{1}{h}T_x u + h\left(\frac{1}{2}+\alpha_p\right)t_x^p u_{xx} + h\frac{1}{2}t_y^p u_{yy}, \\ (u_t^-)_q &= \left(\frac{u_{i,j}-u_{i,j-1}}{h} + \left(\frac{1}{2}+\alpha_q\right)hu_{yy}\right)t_q^y \\ &\quad + \left((1+\alpha_q)\frac{u_{i,j}-u_{i-1,j}}{h} - \alpha_q\frac{u_{i,j-1}-u_{i-1,j-1}}{h} + \frac{1}{2}hu_{xx}\right)t_q^x + \mathrm{O}(h^2) \\ &:= \frac{1}{h}T_y u + h\left(\frac{1}{2}+\alpha_q\right)t_q^y u_{yy} + h\frac{1}{2}t_q^x u_{xx}.\end{aligned}$$

By plugging $(u_t^-)_p$, $(u_t^-)_q$ back to the formulae u_{xx} and u_{yy}, we arrive the following coupling equation:

$$\mathbf{M}\begin{bmatrix}u_{xx}\\ u_{yy}\end{bmatrix} = \begin{bmatrix}L_x u + b_p t_p^x T_x u + J_p \\ L_y u + b_q t_q^y T_y u + J_q\end{bmatrix}, \tag{21}$$

where

$$\mathbf{M} = \begin{bmatrix}1-\left(\frac{1}{2}+\alpha_p\right)b_p t_p^x t_p^x & -\frac{1}{2}b_p t_p^x t_p^y \\ -\frac{1}{2}b_q t_q^x t_q^y & 1-\left(\frac{1}{2}+\alpha_q\right)b_q t_q^y t_q^y\end{bmatrix},$$
$$b_p = -(\beta_p+\beta_p^2)(\rho_p^+ - \rho_p^-),$$
$$b_q = -(\beta_q+\beta_q^2)(\rho_q^+ - \rho_q^-),$$
$$J_p = -\left((1+2\beta_p)[u]_p + (\beta_p+\beta_p^2)h\left(\frac{[\varepsilon u_n]_p}{\hat{\varepsilon}_p}n_x^p + \rho_p^+[u_t]_p t_x^p\right)\right),$$
$$J_q = -\left((1+2\beta_q)[u]_q + (\beta_q+\beta_q^2)h\left(\frac{[\varepsilon u_n]_q}{\hat{\varepsilon}_q}n_y^q + \rho_q^+[u_t]_q t_y^q\right)\right).$$

We shall see in Appendix A that $\mathbf{M}$ is invertible if $\kappa h < \text{Const}$. By inverting $\mathbf{M}$ in (21), we obtain finite difference approximation formula of u_{xx} and u_{yy} at (i,j).

Remark. Notice that the definitions of T_x in case 1 and 2 are slightly different. A unified definition in d dimensions will be given by (39).

4. CIM for complex interface problems

In dealing with complex interface problems in d dimensions, we classify the underlying Cartesian grid points into interiors, normal on-fronts and exceptionals. A standard central finite difference method is applied to interior grid points; the CIM2 is adopted at normal on-front points; and the CIM1 is used at those exceptional grid points. We call this method the hybrid coupling interface method, or simply the coupling interface method. In most applications, the number of grid points is $O(h^{-d})$ for interiors, $O(h^{1-d})$ for normal on-fronts, and O(1) for exceptionals. This leads to a second-order accuracy for the hybrid coupling interface method.

4.1. Classification of grid points

In d dimensions, we consider our domain to be $[0,1]^d$. We partition $[0,1]$ into N subintervals evenly. Let $h = 1/N$. We use the multi-index $\mathbf{i}$ to stand for $(i_1,\ldots,i_d)$. Let $\mathbf{e}_k, k = 1,\ldots,d$ be the Euclidean unit vectors. We define $\gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_k}$ to be the number of the intersections of the segment $[\mathbf{x}_\mathbf{i}, \mathbf{x}_{\mathbf{i}+\mathbf{e}_k})$ and the interface Γ, and we assume $\gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_k} \leqslant 1$ for all $\mathbf{i}$ and k. This assumption is quite mild for most applications. A grid point $\mathbf{x}_\mathbf{i}$ is called an *interior* grid point if for all $1 \leqslant k \leqslant d, \gamma_{\mathbf{i}-\frac{1}{2}\mathbf{e}_k} = \gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_k} = 0$, otherwise we call it on-front. We further classify those on-front grid points into *normal* and *exceptional*. To give a precise definition, we first define the interface orientation indicator $s_{\mathbf{i},k}$ at $\mathbf{x}_\mathbf{i}$ in the direction $\mathbf{e}_k$ to be $s_{\mathbf{i},k} := \gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_k} - \gamma_{\mathbf{i}-\frac{1}{2}\mathbf{e}_k}$. A normal on-front grid point $\mathbf{x}_\mathbf{i}$ is defined to satisfy the following conditions:

- $\forall k, \gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_k} + \gamma_{\mathbf{i}-\frac{1}{2}\mathbf{e}_k} = 0$ or 1, and when $\gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_k} + \gamma_{\mathbf{i}-\frac{1}{2}\mathbf{e}_k} = 1$ and if $\gamma_{\mathbf{i}+\frac{1}{2}s_{\mathbf{i},k}\mathbf{e}_k} = 1$, then $\gamma_{\mathbf{i}+\frac{3}{2}s_{\mathbf{i},k}\mathbf{e}_k} = 0$;
- when $\gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_k} + \gamma_{\mathbf{i}-\frac{1}{2}\mathbf{e}_k} = 1$, then for all $j \neq k$
  - if $\gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_j} = \gamma_{\mathbf{i}-\frac{1}{2}\mathbf{e}_j} = 0$ then $\gamma_{\mathbf{i}-s_{\mathbf{i},k}\mathbf{e}_k+\frac{1}{2}\mathbf{e}_j} = \gamma_{\mathbf{i}-s_{\mathbf{i},k}\mathbf{e}_k-\frac{1}{2}\mathbf{e}_j} = 0$,
  - if $\gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_j} = 1$ then $\gamma_{\mathbf{i}-s_{\mathbf{i},k}\mathbf{e}_k-\frac{1}{2}\mathbf{e}_j} = 0$,
  - if $\gamma_{\mathbf{i}-\frac{1}{2}\mathbf{e}_j} = 1$ then $\gamma_{\mathbf{i}-s_{\mathbf{i},k}\mathbf{e}_k+\frac{1}{2}\mathbf{e}_j} = 0$.

The first condition means that we can apply the CIM2 to every coordinate direction. The second condition allows us to perform one-side interpolation for the cross derivatives $u_{x_kx_j}$ at $\mathbf{x}_\mathbf{i}$. Fig. 8 contains all possible cases of a normal on-front grid points in two and three dimensions, up to rotation and reflection. In Fig. 8, the centered grid point is a normal on-front grid point $\mathbf{x}_\mathbf{i}$; the bullet points are the grid points appeared in the finite difference stencil about $\mathbf{x}_\mathbf{i}$. There are 8 such points in two dimensions and 12–14 points in three dimensions. A pseudocode for grid point classification will be put in Appendix A.

4.2. CIM1 in d dimensions

Let $\mathbf{x}_\mathbf{i}$ be an on-front grid point at which we plan to discretize the PDE (1). For simplicity of notation, let us drop the index $\mathbf{i}$. The interface Γ partitions the domain into two regions $\Omega^\pm$. As before, we call the component where $\mathbf{x}$ is situated Ω^- and the other Ω^+. At $\mathbf{x}$, we first perform the following approximation:

$$-\nabla\cdot(\varepsilon(\mathbf{x})\nabla u(\mathbf{x})) = -\frac{\varepsilon(\mathbf{x})}{h}\sum_{k=1}^{d}\left(\frac{\partial}{\partial x_k}u^-\left(x+\frac{1}{2}h\mathbf{e}_k\right) - \frac{\partial}{\partial x_k}u^-\left(x-\frac{1}{2}h\mathbf{e}_k\right)\right) + O(1).$$

Our goal is provide a first-order finite difference approximation to these quantities. We shall only discuss the case for $\partial u^-/\partial x_k(\mathbf{x}+h\mathbf{e}_k/2), k = 1,\ldots,d$. The discussion for the case of $\partial u^-/\partial x_k(\mathbf{x}-h\mathbf{e}_k/2), k = 1,\ldots,d$ is identical. The discretization procedure consists of the following steps.

4.2.1. Dimension-by-dimension discretization

At $\mathbf{x}$, we adopt the one-dimensional formula (5) in each segment $(\mathbf{x}, \mathbf{x}+h\mathbf{e}_k)$. Namely, up to an error $O(h)$,

$$\frac{\partial}{\partial x_k}u^-\left(\mathbf{x}+\frac{1}{2}h\mathbf{e}_k\right) \approx \frac{1}{h}\left(\bar{\rho}_k^+(u(\mathbf{x}+h\mathbf{e}_k)-u(\mathbf{x})) - \bar{\rho}_k^+[u]_{\widehat{\mathbf{x}_k}} - \beta_k h\frac{[\varepsilon\nabla u\cdot\mathbf{e}_k]_{\widehat{\mathbf{x}_k}}}{\bar{\varepsilon}_k}\right), \tag{22}$$

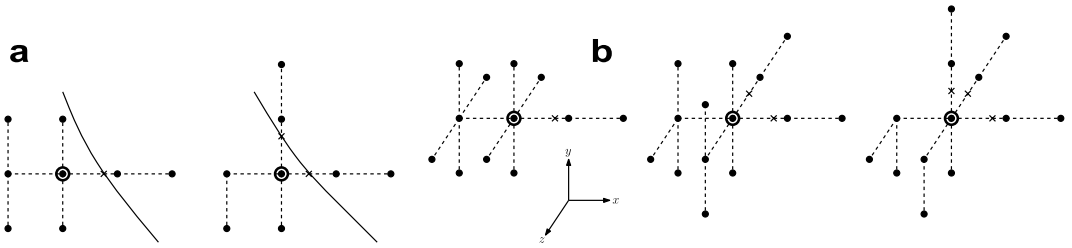


Fig. 8. Normal on-front points. bullet: grid points used in the stencil. cross: interface points. bullet with circle: $\mathbf{x}_{\mathbf{i}}$. (a) Two dimension: 2 cases and (b) three dimension: 3 cases.

where $\widehat{\mathbf{x}}_k$ is the intersection of the interface Γ and the grid segment $[\mathbf{x}, \mathbf{x}+h\mathbf{e}_k)$. The coefficients β_k, etc. are defined as those in the one-dimensional formula (4).

4.2.2. Decomposition of jump data in each dimension

To compute $[\varepsilon\nabla u\cdot\mathbf{e}_k]_{\widehat{\mathbf{x}}_k}$, we decompose $\mathbf{e}_k$ into a linear combination of $\mathbf{n}_k$ and $\mathbf{t}_k$ at $\widehat{\mathbf{x}}_k$, where $\mathbf{n}_k$ is the unit normal vector of the interface at $\widehat{\mathbf{x}}_k$ from Ω^- to Ω^+ and $\mathbf{t}_k$ is the unit vector in the projection of $\mathbf{e}_k$ onto the tangent plane of Γ at $\widehat{\mathbf{x}}_k$. The jump $[\varepsilon\nabla u\cdot\mathbf{e}_k]_{\widehat{\mathbf{x}}_k}$ is decomposed into the follows:

$$\begin{aligned}[\varepsilon\nabla u\cdot\mathbf{e}_k]_{\widehat{\mathbf{x}}_k} &= [\varepsilon\nabla u\cdot\mathbf{n}_k]_{\widehat{\mathbf{x}}_k}(\mathbf{n}_k\cdot\mathbf{e}_k) + [\varepsilon\nabla u\cdot\mathbf{t}_k]_{\widehat{\mathbf{x}}_k}(\mathbf{t}_k\cdot\mathbf{e}_k)\\ &= [\varepsilon\nabla u\cdot\mathbf{n}_k]_{\widehat{\mathbf{x}}_k}(\mathbf{n}_k\cdot\mathbf{e}_k) + (\varepsilon_k^+[\nabla u\cdot\mathbf{t}_k]_{\widehat{\mathbf{x}}_k} + (\varepsilon_k^+-\varepsilon_k^-)\nabla u^-(\widehat{\mathbf{x}}_k)\cdot\mathbf{t}_k)(\mathbf{t}_k\cdot\mathbf{e}_k).\end{aligned} \tag{23}$$

The quantities $[\varepsilon\nabla u\cdot\mathbf{n}_k]_{\widehat{\mathbf{x}}_k} = \sigma(\widehat{\mathbf{x}}_k)$ and $[\nabla u\cdot\mathbf{t}_k]_{\widehat{\mathbf{x}}_k} = \partial\tau/\partial\mathbf{t}_k(\widehat{\mathbf{x}}_k)$ are obtained from the prescribed jump conditions. After substituting (23) into (22), we get

$$\frac{\partial}{\partial x_k}u^-\left(\mathbf{x}+\frac{1}{2}h\mathbf{e}_k\right) = \frac{1}{h}(\bar{D}_k u(\mathbf{x}) + \bar{b}_k h(\nabla u^-(\widehat{\mathbf{x}}_k)\cdot\mathbf{t}_k)(\mathbf{t}_k\cdot\mathbf{e}_k) + \bar{J}_k) + \mathrm{O}(h). \tag{24}$$

where

$$\bar{D}_k u(\mathbf{x}) = \bar{\rho}_k^+(u(\mathbf{x}+h\mathbf{e}_k)-u(\mathbf{x})), \tag{25}$$

$$\bar{b}_k = -\beta_k(\bar{\rho}_k^+ - \bar{\rho}_k^-), \tag{26}$$

$$\bar{J}_k = -\left(\bar{\rho}_k^+[u]_{\widehat{\mathbf{x}}_k} + \beta_k h\left(\frac{[\varepsilon\nabla u\cdot\mathbf{n}_k]}{\bar{\varepsilon}_k}(\mathbf{n}_k\cdot\mathbf{e}_k) + \bar{\rho}_k^+[\nabla u\cdot\mathbf{t}_k]_{\widehat{\mathbf{x}}_k}(\mathbf{t}_k\cdot\mathbf{e}_k)\right)\right). \tag{27}$$

4.2.3. Coupling and one-side interpolation

We approximate $\partial u^-/\partial x_j(\widehat{\mathbf{x}}_k)$ in terms of $\partial u^-/\partial x_j(\mathbf{x}+h\mathbf{e}_j/2)$ (with $\mathrm{O}(h)$ error) by Taylor expansion as the follows.

- $j=k:\frac{\partial}{\partial x_k}u^-(\hat{\mathbf{x}}_k)\approx\frac{\partial}{\partial x_k}u(\mathbf{x}+\frac{1}{2}h\mathbf{e}_k)$,
- $j\neq k:\frac{\partial}{\partial x_j}u^-(\hat{\mathbf{x}}_k)\approx\begin{cases}\frac{u(\mathbf{x}+h\mathbf{e}_j)-u(\mathbf{x})}{h} & \text{for } \gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_j}=0,\\ \frac{\partial}{\partial x_j}u^-(\mathbf{x}+\frac{1}{2}h\mathbf{e}_j) & \text{for } \gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_j}\neq 0.\end{cases}$

By plugging this formula back to (24), we arrive a $d\times d$ systems of linear equations:

$$\bar{\mathbf{M}}\left[\left(\frac{\partial}{\partial x_k}u\left(\mathbf{x}+\frac{1}{2}h\mathbf{e}_k\right)\right)_{k=1}^{d}\right] = \frac{1}{h}\left[(\bar{D}_k u(\mathbf{x}) + \bar{b}_k\bar{T}_k u(\mathbf{x})(\mathbf{t}_k\cdot\mathbf{e}_k) + \bar{J}_k)_{k=1}^{d}\right],$$

where

$$\bar{T}_k u(\mathbf{x}) = \sum_{j=1, j\neq k}^{d} (u(\mathbf{x}+h\mathbf{e}_j) - u(\mathbf{x}))(\mathbf{t}_k \cdot \mathbf{e}_j)\left(1 - \gamma_{i+\frac{1}{2}\mathbf{e}_j}\right).$$

The entries of $\bar{\mathbf{M}}$ are:

$$(\bar{\mathbf{M}})_{k,j} = \begin{cases} 1 - \gamma_k \bar{b}_k (\mathbf{t}_k \cdot \mathbf{e}_k)^2 & \text{if } k = j, \\ -\gamma_j \gamma_k \bar{b}_k (\mathbf{t}_k \cdot \mathbf{e}_j)(\mathbf{t}_k \cdot \mathbf{e}_k) & \text{if } k \neq j. \end{cases}$$

Notice that γ_k here is an abbreviation of $\gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_k}$.

4.3. CIM2 in d dimensions

The CIM2 discretization procedure is divided into the following three steps.

4.3.1. Dimension-by-dimension discretization

We first approximate $-\frac{\partial}{\partial x_k}(\varepsilon(\mathbf{x})\frac{\partial u}{\partial x_k})$ by

$$\begin{aligned} -\frac{\partial}{\partial x_k}\left(\varepsilon(\mathbf{x})\frac{\partial u}{\partial x_k}\right) &= -\left(\frac{\partial}{\partial x_k}\varepsilon(\mathbf{x})\right)\left(\frac{\partial}{\partial x_k}u(\mathbf{x})\right) - \varepsilon(\mathbf{x})\frac{\partial^2}{\partial x_k^2}u(\mathbf{x}) \\ &= -\frac{1}{h^2}\left((D_k^{(s_k)}\varepsilon(\mathbf{x}))(D_k^{(s_k)}u(\mathbf{x})) + \varepsilon(\mathbf{x})h^2\frac{\partial^2}{\partial x_k^2}u(\mathbf{x})\right) + \mathrm{O}(h), \end{aligned} \tag{28}$$

where $s_k := \gamma_{\mathbf{i}+\frac{1}{2}\mathbf{e}_k} - \gamma_{\mathbf{i}-\frac{1}{2}\mathbf{e}_k}$ and

$$D_k^{(s_k)}u(\mathbf{x}) := \frac{1}{2}((1-s_k)u(\mathbf{x}+h\mathbf{e}_k) + 2s_k u(\mathbf{x}) - (1+s_k)u(\mathbf{x}-h\mathbf{e}_k)) \tag{29}$$

$$= \begin{cases} \frac{1}{2}(u(\mathbf{x}+h\mathbf{e}_k) - u(\mathbf{x}-h\mathbf{e}_k)) & \text{if } s_k = 0, \\ u(\mathbf{x}) - u(\mathbf{x}-h\mathbf{e}_k) & \text{if } s_k = 1, \\ u(\mathbf{x}+h\mathbf{e}_k) - u(\mathbf{x}) & \text{if } s_k = -1. \end{cases} \tag{30}$$

Next, we derive a finite difference approximation to $\partial^2 u/\partial x_k^2$. We adopt the one-dimensional formula (16):

$$\frac{\partial^2}{\partial x_k^2}u(\mathbf{x}) = \frac{1}{h^2}\left(L_k^{(s_k)}u(\mathbf{x}) - (1+2\beta_k)[u]_{\widehat{\mathbf{x}}_k} - s_k(\beta_k+\beta_k^2)h\frac{[\varepsilon\nabla u\cdot\mathbf{e}_k]_{\widehat{\mathbf{x}}_k}}{\hat{\varepsilon}_k}\right) + \mathrm{O}(h), \tag{31}$$

where $L_k^{(s_k)}u(\mathbf{x})$ is defined as

$$L_k^{(s_k)}u(\mathbf{x}) = \begin{cases} a_{k,-s_k}u(\mathbf{x}-s_k h\mathbf{e}_k) + a_{k,0}u(\mathbf{x}) + a_{k,s_k}u(\mathbf{x}+s_k h\mathbf{e}_k) + a_{k,2s_k}u(\mathbf{x}+2s_k h\mathbf{e}_k) & \text{if } s_k = \pm 1, \\ u(\mathbf{x}-h\mathbf{e}_k) - 2u(\mathbf{x}) + u(\mathbf{x}+h\mathbf{e}_k) & \text{if } s_k = 0. \end{cases} \tag{32}$$

4.3.2. Decomposition of jump data in each dimension

The jump $[\varepsilon\nabla u\cdot\mathbf{e}_k]_{\widehat{\mathbf{x}}_k}$ is decomposed into the follows:

$$\begin{aligned} [\varepsilon\nabla u\cdot\mathbf{e}_k]_{\widehat{\mathbf{x}}_k} &= [\varepsilon\nabla u\cdot\mathbf{n}_k]_{\widehat{\mathbf{x}}_k}(\mathbf{n}_k\cdot\mathbf{e}_k) + [\varepsilon\nabla u\cdot\mathbf{t}_k]_{\widehat{\mathbf{x}}_k}(\mathbf{t}_k\cdot\mathbf{e}_k) \\ &= [\varepsilon\nabla u\cdot\mathbf{n}_k]_{\widehat{\mathbf{x}}_k}(\mathbf{n}_k\cdot\mathbf{e}_k) + \left(\hat{\varepsilon}_k^+[\nabla u\cdot\mathbf{t}_k]_{\widehat{\mathbf{x}}_k} + (\hat{\varepsilon}_k^+ - \hat{\varepsilon}_k^-)(\nabla u^-(\widehat{\mathbf{x}}_k)\cdot\mathbf{t}_k)\right)(\mathbf{t}_k\cdot\mathbf{e}_k). \end{aligned} \tag{33}$$

The quantities $[u]_{\widehat{\mathbf{x}}_k}$, $[\varepsilon\nabla u\cdot\mathbf{n}_k]_{\widehat{\mathbf{x}}_k}$ and $[\nabla u\cdot\mathbf{t}_k]_{\widehat{\mathbf{x}}_k}$ can be obtained from the prescribed jump conditions. Substituting (33) into (31), we obtain

$$\frac{\partial^2}{\partial x_k^2}u(\mathbf{x}) = \frac{1}{h^2}\left(L_k^{(s_k)}u(\mathbf{x}) + s_k b_k(\nabla u^-(\widehat{\mathbf{x}}_k)\cdot\mathbf{t}_k)(\mathbf{t}_k\cdot\mathbf{e}_k) + J_k\right) + \mathrm{O}(h), \tag{34}$$

where

$$b_k = -(\beta_k + \beta_k^2)(\rho_k^+ - \rho_k^-), \tag{35}$$

$$J_k = -\left((1+2\beta_k)[u]_{\widehat{\mathbf{x}}_k} + s_k(\beta_k+\beta_k^2)h\left(\frac{[\varepsilon \nabla u\cdot \mathbf{n}_k]_{\widehat{\mathbf{x}}_k}}{\hat{\varepsilon}_k}(\mathbf{n}_k\cdot\mathbf{e}_k) + \rho_k^+ [\nabla u\cdot \mathbf{t}_k]_{\widehat{\mathbf{x}}_k}\cdot(\mathbf{t}_k\cdot\mathbf{e}_k)\right)\right). \tag{36}$$

4.3.3. Coupling and one-side interpolation

We use the following formula to give a second-order approximation to $\partial u^-/\partial x_j(\widehat{\mathbf{x}}_k)$:

$$\frac{\partial u^-}{\partial x_j}(\widehat{\mathbf{x}}_k) = \begin{cases} \frac{1}{h}D_k^{(s_k)}u(\mathbf{x}) + \left(\frac{1}{2}+\alpha_k\right)s_k h\frac{\partial^2}{\partial x_k^2}u(\mathbf{x}) + \mathrm{O}(h^2) & \text{for } j=k, \\ \frac{1}{h}D_j^{(s_j)}((1+\alpha_k)u(\mathbf{x}) - \alpha_k u(\mathbf{x}-s_k h\mathbf{e}_k)) + s_j\frac{h}{2}\frac{\partial^2}{\partial x_j^2}u(\mathbf{x}) + \mathrm{O}(h^2) & \text{otherwise.} \end{cases} \tag{37}$$

This formula is derived from Taylor expansion. Basically, the cross derivatives are approximated by one-side interpolation, whereas the principal second-order derivatives $u_{x_k x_k}$ are remained. The purpose to do this is to reduce the total number of grid points used for one-side interpolations.

From (37), we obtain

$$\nabla u^-(\widehat{\mathbf{x}}_k)\cdot\mathbf{t}_k = \frac{1}{h}T_k u(\mathbf{x}) + h\left(s_k\left(\frac{1}{2}+\alpha_k\right)(\mathbf{t}_k\cdot\mathbf{e}_k)\frac{\partial^2}{\partial x_k^2}u(\mathbf{x}) + \frac{1}{2}\sum_{j=1,j\neq k}^{d} s_j(\mathbf{t}_k\cdot\mathbf{e}_j)\frac{\partial^2}{\partial x_j^2}u(\mathbf{x})\right) + \mathrm{O}(h^2), \tag{38}$$

where $T_k u(\mathbf{x})$ is defined as

$$T_k u(\mathbf{x}) = (\mathbf{t}_k\cdot\mathbf{e}_k)D_k^{(s_k)}u(\mathbf{x}) + \sum_{j=1,j\neq k}^{d}(\mathbf{t}_k\cdot\mathbf{e}_j)D_j^{(s_j)}((1+\alpha_k)u(\mathbf{x}) - \alpha_k u(\mathbf{x}-s_k h\mathbf{e}_k)). \tag{39}$$

Combining (34), (36) and (38), we obtain a $d\times d$ system of linear equations for $(\partial^2 u/\partial x_k^2(\mathbf{x}))_{k=1}^d$:

$$\mathbf{M}\left(\frac{\partial^2}{\partial x_k^2}u(\mathbf{x})\right)_{k=1}^{d} = \frac{1}{h^2}(Lu(\mathbf{x}) + Tu(\mathbf{x}) + J),$$

where $\mathbf{M}_{d\times d} = (m_{k,j})_{d\times d}$ is defined by

$$m_{k,j} = \begin{cases} 1 - |s_k|(\frac{1}{2}+\alpha_k)b_k(\mathbf{t}_k\cdot\mathbf{e}_k)^2 & j=k, \\ -\frac{1}{2}s_j s_k b_k(\mathbf{t}_k\cdot\mathbf{e}_j)(\mathbf{t}_k\cdot\mathbf{e}_j) & j\neq k, \end{cases} \tag{40}$$

$$L = (L_1,\cdots,L_d)^{\mathrm{T}},$$

$$T = (s_1 b_1 T_1,\cdots,s_d b_d T_d)^{\mathrm{T}},$$

$$J = (J_1,\cdots,J_d)^{\mathrm{T}}.$$

We shall show in Appendix A that under the condition $\kappa h < \mathrm{Const}$, the matrix $\mathbf{M}$ is invertible. Here κ is the total curvature. Hence we get

$$h^2\left(\frac{\partial^2}{\partial x_k^2}u(\mathbf{x})\right)_{k=1}^{d} = \mathbf{M}^{-1}((Lu(\mathbf{x}) + Tu(\mathbf{x}) + J)). \tag{41}$$

Substituting this formula into (28) and defining a vector $\tilde{\boldsymbol{\varepsilon}}$ by

$$\tilde{\boldsymbol{\varepsilon}} = \varepsilon(\mathbf{x})[\,1 \quad 1 \quad \cdots \quad 1\,]\mathbf{M}^{-1} = [\,\tilde{\boldsymbol{\varepsilon}}_1 \quad \tilde{\boldsymbol{\varepsilon}}_2 \quad \cdots \quad \tilde{\boldsymbol{\varepsilon}}_d\,],$$

we get

$$-\nabla\cdot(\varepsilon(\mathbf{x})\nabla u(\mathbf{x})) = -\frac{1}{h^2}\sum_{k=1}^{d}((D_k^{(s_k)}\varepsilon(\mathbf{x}))(D_k^{(s_k)}u(\mathbf{x})) + \tilde{\boldsymbol{\varepsilon}}_k(L_k u(\mathbf{x}) + s_k a_{t,k}T_k u(\mathbf{x}) + J_k)) + \mathrm{O}(h). \tag{42}$$

5. Numerical experiments

In this section, we perform the following numerical investigation:

1. convergence tests for CIM1 in three dimensions,
2. convergence tests and a comparison study for CIM2 in two and three dimensions,
3. convergence tests and a comparison study for the (hybrid) CIM for complex interface problems in two and three dimensions.

5.1. Numerical implementation

The numerical implementation of CIM is easy. It is only few hundred lines of C code. The logic of the classification of grid points is also simple, there are only two cases in two dimensions and three cases in three dimensions. Algebraic multigrid (AMG) is adopted for solving the resulting linear systems. Gauss–Seidel method is used as a smoother. For readers' convenience, a pseudocode is provided in Appendix A.

5.2. Convergence tests of the CIM1

Example 1. We propose six interfaces in three dimensions to test the convergence of the CIM1. They are plotted in Fig. 9.

- Eight balls: $\phi(x,y,z) = \min_{0\leqslant k\leqslant 7}\{\sqrt{(x-x_k)^2+(y-y_k)^2+(z-z_k)^2}\} - 0.3$, where $(x_k,y_j,z_k) = ((-1)^{\lfloor k/4\rfloor}\times 0.5, (-1)^{\lfloor k/2\rfloor}\times 0.5, (-1)^k\times 0.5)$, $0\leqslant k\leqslant 7$.
- Ellipsoid: $\phi(x,y,z) = 2x^2+3y^2+6z^2-1.3^2$.
- Peanut: $\phi(r,\theta,\phi) = r - 0.5 - 0.2\sin(2\theta)\sin(\phi)$.
- Donut: $\phi(x,y,z) = (\sqrt{x^2+y^2}-0.6)^2+z^2-0.09$.
- Banana[1]: $\phi(x,y,z) = (7x+6)^4 + 2401y^4 + 3601.5z^4 + 98(7x+6)^2y^2 + 98(7x+6)^2z^2 + 4802y^2z^2 - 94(7x+6)^2 + 3822y^2 - 4606z^2 + 1521$.
- Popcorn: $\phi(x,y,z) = \sqrt{x^2+y^2+z^2} - r_0 - \sum_{k=0}^{11} A e^{-((x-x_k)^2+(y-y_k)^2+(z-z_k)^2)/\sigma^2}$, where

$$(x_k,y_k,z_k) = \frac{r_0}{\sqrt{5}}\left(2\cos\left(\frac{2k\pi}{5}\right), 2\sin\left(\frac{2k\pi}{5}\right), 1\right), \quad 0\leqslant k\leqslant 4,$$

$$(x_k,y_k,z_k) = \frac{r_0}{\sqrt{5}}\left(2\cos\left(\frac{(2(k-5)-1)\pi}{5}\right), 2\sin\left(\frac{(2(k-5)-1)\pi}{5}\right), -1\right), \quad 5\leqslant k\leqslant 9,$$

$$(x_{10},y_{10},z_{10}) = (0,0,r_0),$$

$$(x_{11},y_{11},z_{11}) = (0,0,-r_0).$$

The parameters are: $r_0 = 0.6$, $A = 2$, $\sigma = 0.2$.

We choose the following coefficients and test function:

$$\varepsilon(\mathbf{x}) = \begin{cases} \varepsilon^-, & \mathbf{x}\in\Omega^-, \\ \varepsilon^+, & \mathbf{x}\in\Omega^+, \end{cases}$$

with $\varepsilon^- = 2$, $\varepsilon^+ = 80$, and

[1] This example is taken from http://www.mai.liu.se/ejoh/math.html.

$$= \begin{cases} x^3 + xy^2 + y^3 + \\ xy + x^4 + y^4 + xz \end{cases} \in \Omega^-,$$

Th s designed so that the c jump conditions and the source f a the governing Eq. (1). 1/h) versus the absolute error $\|$ umber N varies from 20 to re fit of the absolute errors s 1 is between 1.1 and ce-dimensional complex interfac

and comparison study of CIM2

ion, we perform the numerical investiga , (ii) a comparison with some other s

d orientation l of 1

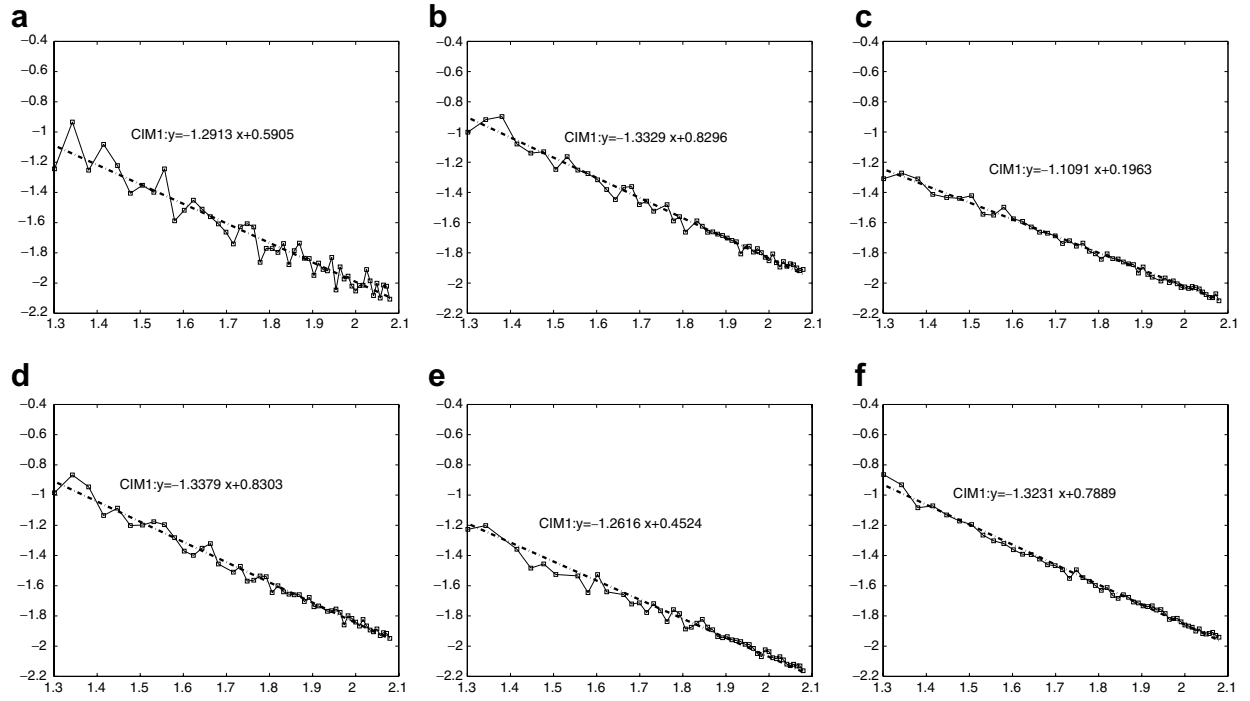


Fig. 10. Convergence tests for the CIM1. The x-axis: $\log_{10}(N)$, the y-axis: $\log_{10}(\|u-u_e\|_\infty)$, where N is the number of grid point in each dimension ranging from 20 to 120. The least square fit shows that the order of accuracy of the CIM1 is about 1.1–1.3: (a) 8 balls, (b) ellipsoid, (c) peanut, (d) donut, (e) banana, (f) popcorn.

$$\varepsilon(x,y)=\begin{cases}\varepsilon^-, & \phi(x,y)<0,\\ \varepsilon^+, & \phi(x,y)\geqslant 0,\end{cases}$$

with $\varepsilon^-=2$ and $\varepsilon^+=80$. The exact solution u is chosen to be symmetric about 0:

$$u(x,y)=\begin{cases}r^2 & \text{for } \phi(x,y)<0,\\ \frac{1}{\varepsilon_2}(\frac{1}{2}r^4+r^2)+c & \text{for } \phi(x,y)\geqslant 0,\end{cases}$$

where $r=\sqrt{x^2+y^2}$, and $c=1-\frac{9}{8\varepsilon_2}$. To investigate the grid orientation effect, we rotate the interface with θ varying from 0° to 90° with increment 2°. Fig. 11 is the maximum errors versus θ with mesh size $h=0.02$ (or $N=100$). We observe that the maximum errors vary only slightly, within tolerance of the second-order errors. *No grid orientation effect is found.*

5.4. Convergence and comparison study for the CIM2

For comparison study we shall compare with EJIIM, MIIM, DIIM, JCCS and MIB. The compared cases and methods are listed in Table 1.

In the test problems below, the computational domain is $\Omega=[-1,1]^d$, and the interface is represented by $\phi(\mathbf{x})=0$. The region Ω is partitioned into $\Omega^-:=\{\mathbf{x}\in\Omega|\phi(\mathbf{x})<0\}$ and $\Omega^+:=\{\mathbf{x}\in\Omega|\phi(\mathbf{x})>0\}$. The exact solution u_e is prescribed. The jump conditions, boundary conditions and the source terms are then computed analytically from Eq. (1). In these comparison studies, the compared item is the maximum norm of the absolute error u, i.e. $\|u-u_e\|_\infty$. In addition, we also list the total CPU time and the maximum norm of the errors of ∇u along the interface. We use (37) and (34) to compute the gradients on the interface.

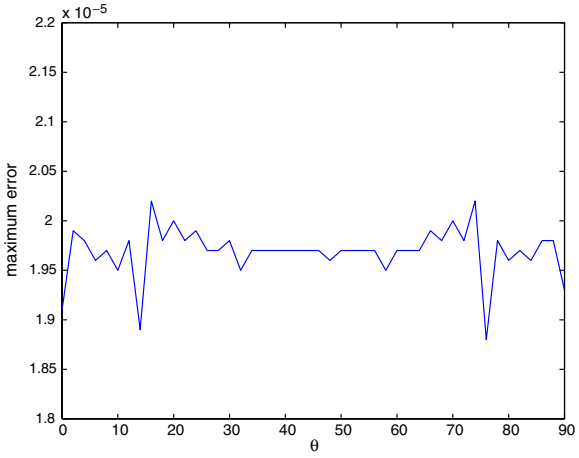


Fig. 11. Study of grid orientation effect for the CIM2. As we vary the grid orientation θ from 0° to 90°, we find the errors change within the tolerance error for the case of $N = 100$. No grid orientation effect is found for the CIM2.

Table 1
Table of compared methods and benchmark test cases

	Method Year	EJIIM [49] 2000	MIIM [30,8] 2001, 2003	DIIM [5] 2004	JCCS [48] 2004	MIB [53] 2006
2D	Example 3	√	√	√		
	Example 4				√	√
3D	Example 5		√			

Examples 3 and 4 are two benchmark problems in two dimensions. Example 5 is a comparison test of interface problems in three dimensions. The corresponding figures in each examples are the absolute errors computed by the CIM2. The corresponding tables are the comparison results. The compared item is the absolute error $\|u - u_e\|_{\infty,\Gamma}$. In addition, we also list the CPU time and the gradient error.

We list the following observed results from these comparison tables:

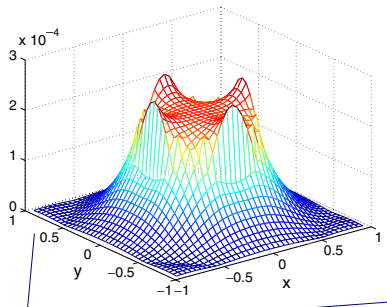
1. CIM2 is second-order accurate for both u and its gradients.
2. CIM2 produces less absolute error as compared with other methods, despite it uses few interpolation points.
3. CIM2 is less sensitive to the contrast of ε.

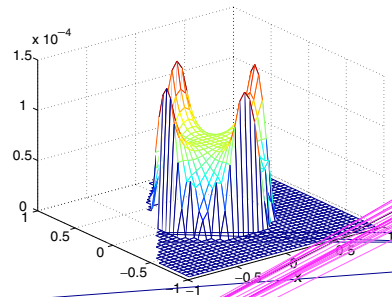
In addition, we show the gradients (i.e. $\|\nabla u - \nabla u_e\|_{\infty,\Gamma}$) are also second-order accurate. In the column of the CPU time spent by CIM2, we observe that they grow slightly more than linear as a function of number of unknowns. This is because the reduction rate of AMG increases from 0.1 to 0.7 as we refine the meshes.

Example 3. This benchmark test is quoted from [26]:

$$\phi(x,y) = r - 0.5,$$
$$\varepsilon(x,y) = \begin{cases} 1 + r^2, & (x,y) \in \Omega^-, \\ b, & (x,y) \in \Omega^+, \end{cases}$$
$$u_e(x,y) = \begin{cases} r^2, & (x,y) \in \Omega^-, \\ (r^4/2 + r^2 + 0.1\log(2r))/b - (0.5^4/2 + 0.5^2)/b + 0.5^2, & (x,y) \in \Omega^+, \end{cases}$$
$$f(x,y) = -8r^2 - 4,$$

where $r = \sqrt{x^2 + y^2}$ and b is a parameter. We show the results for $b = 10, 1000$ and 0.001 (Fig. 12, Tables 2–4).





Example 4. This example is quoted from [48,53]:

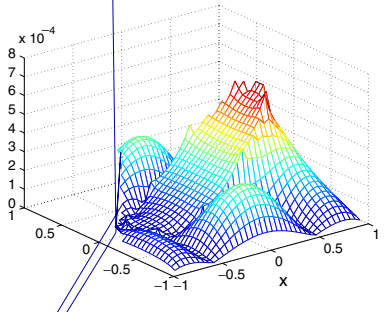
$$\phi(x,y) = \left(\frac{x^2}{18/27}\right)^2 + \left(\frac{y}{10/27}\right)^2 - 1,$$

$$\varepsilon(x,y) = \begin{cases} \varepsilon^-, & (x,y) \in \Omega^-, \\ \varepsilon^+, & (x,y) \in \Omega^+, \end{cases}$$

$$u_e(x,y) = \begin{cases} e^x \cos y, & (x,y) \in \Omega^-, \\ 5e^{-x^2-y^2/2}, & (x,y) \in \Omega^+, \end{cases}$$

$$f(\mathbf{x}) = \begin{cases} 0, & (x,y) \in \Omega^-, \\ -5e^{-x^2-y^2/2}(-3+4x^2+y^2), & (x,y) \in \Omega^+. \end{cases}$$

We show the cases for $\varepsilon^+ = 1, \varepsilon^- = 10, 1000$ as those in [48,53] (Fig. 13, Tables 5 and 6).



Example 5. This three-dimensional example is taken from [8]:

$$\phi(x,y,z) = r - 0.5,$$

$$\begin{cases} 1 + r^2, & (x,y,z) \in \Omega^-, \\ b, & (x,y,z) \in \Omega^+, \end{cases}$$

$$\begin{cases} r^2, & (x,y,z) \in \Omega^-, \\ (r^4/2 + r^2)/b - (0.5^4/2 + 0.5^2)/b + 0.5^2, & (x,y,z) \in \Omega^+, \end{cases}$$

$$-(10r^2 + 6),$$

$$\sqrt{y^2 + z^2}$$ (Fig. 14, Tables 7–9).

and comparison for the hybrid CIM

ion, we shall test hybrid CIM for complex interface problems in two and three dimensions. n convergence test for the those three-dimensional complex interface problems tested by

Table 6
Example 4: $\varepsilon^- = 1000$, $\varepsilon^+ = 1$

N	CIM2			MIB	JCCS
	CPU	$\|\nabla u - \nabla u_e\|_{\infty,\Gamma}$	$\|u - u_e\|_\infty$	$\|u - u_e\|_\infty$	$\|u - u_e\|_\infty$
20	0.01	1.551×10^{0}	3.539×10^{-1}	9.130×10^{-2}	2.803×10^{0}
40	0.08	4.682×10^{-1}	1.100×10^{-1}	2.764×10^{-2}	7.543×10^{-1}
80	0.41	8.966×10^{-2}	2.028×10^{-2}	7.524×10^{-3}	1.940×10^{-1}
160	2.26	2.799×10^{-2}	6.462×10^{-3}	2.169×10^{-3}	4.906×10^{-2}
320	7.29	6.343×10^{-3}	1.437×10^{-3}	4.841×10^{-4}	1.232×10^{-2}

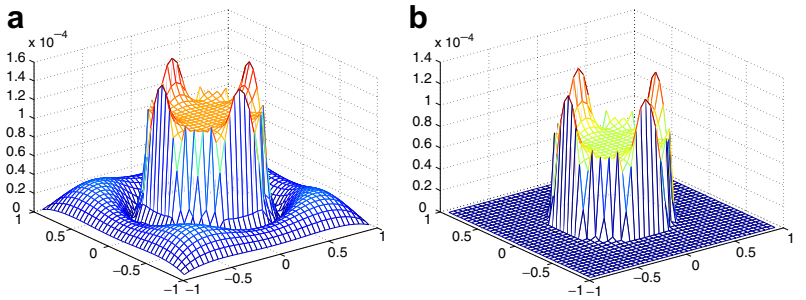


Fig. 14. Example 5. A slice of $|u - u_e|$ at $z = 0$ at mesh $40 \times 40 \times 40$. (a) $|u - u_e|$: $b = 10$ and (b) $|u - u_e|$: $b = 1000$.

Table 7
Example 5: $b = 1$

N	CIM2				MIIM, 27 points	
	CPU	$\|\nabla u - \nabla u_e\|_{\infty,\Gamma}$	$\|u_a - u_e\|_\infty / \|u_e\|_\infty$	Order	$\|u_a - u_e\|_\infty / \|u_e\|_\infty$	Order
26	1.52	1.005×10^{-2}	1.822×10^{-4}		1.247×10^{-3}	
52	20.5	3.685×10^{-3}	4.153×10^{-5}	2.133	3.979×10^{-3}	1.648
104	212	9.729×10^{-4}	9.529×10^{-6}	2.124	9.592×10^{-4}	2.052
208	2355	2.540×10^{-4}	2.230×10^{-6}	2.095	–	–

Table 8
Example 5: $b = 10$

N	CIM2				MIIM, 27 points	
	CPU	$\|\nabla u - \nabla u_e\|_{\infty,\Gamma}$	$\|u_a - u_e\|_\infty / \|u_e\|_\infty$	Order	$\|u_a - u_e\|_\infty / \|u_e\|_\infty$	Order
26	1.45	7.174×10^{-3}	4.332×10^{-4}		1.525×10^{-3}	
52	19.14	2.693×10^{-3}	9.240×10^{-5}	2.229	5.240×10^{-4}	1.541
104	161	7.401×10^{-4}	1.636×10^{-5}	2.498	1.010×10^{-4}	2.375
208	1867	1.979×10^{-4}	3.330×10^{-6}	2.297	–	–

Table 9
Example 5: $b = 1000$

N	CIM2				MIIM, 27 points	
	CPU	$\|\nabla u - \nabla u_e\|_{\infty,\Gamma}$	$\|u_a - u_e\|_\infty / \|u_e\|_\infty$	Order	$\|u_a - u_e\|_\infty / \|u_e\|_\infty$	Order
26	1.48	6.825×10^{-3}	9.133×10^{-4}		3.845×10^{-3}	
52	24.54	2.594×10^{-3}	2.466×10^{-4}	1.889	1.111×10^{-3}	1.649
104	209	7.183×10^{-4}	3.447×10^{-5}	2.839	1.605×10^{-4}	2.791
208	3299	1.925×10^{-4}	4.727×10^{-6}	2.866	–	–

CIM1 in the beginning of this section. We observe an improvement by using hybrid CIM over the CIM1. Next, we shall perform a comparison study with the FIIM by a benchmark complex interface problem in two dimensions. We shall observe that hybrid CIM produces less error.

5.5.1. Number of exceptional grid points

In order to maintain second-order accuracy of the hybrid CIM, the number of exceptional points should stay O(1) as we increase the total number of grid points. Fig. 15 contains the numbers of exceptional points versus N (the number of grid points in each dimension) with N ranging from 20 to 500 printed at every $\Delta N = 2$. We observe that the numbers of exceptional points are less than 200 in most cases as we vary the number of grid points from 20^3 to 500^3. For the case of 8 balls, this number of exceptional points can go up to 1200 out of the total number of grid point 420^3. However, the occurrence of the exceptional points is rare for this case.

5.5.2. Convergence tests for the hybrid CIM

Fig. 16 is the convergence plots for these complex interface problems. The parameter $\varepsilon^- = 2$, $\varepsilon^+ = 80$. We vary N from 20 to 120. The log–log plots (with base 10) of N versus $\|u - u_e\|_\infty$ are provided. We use least square fit to find the orders of convergence. We observe that the convergence is about second order. The oscillation behavior reflects the variation of numbers of exceptional points as we vary N. All but the eight-ball case have small variation. This is consistent to the study of number of exceptional points is O(1) in Fig. 15. For the eight-ball simulation, it has no exceptional points for most N. We see a nice second-order least square fit for these test cases. There are few cases the errors increase. This is due to too many balls inside and not enough resolution. Even in this case, the hybrid CIM does improve CIM1.

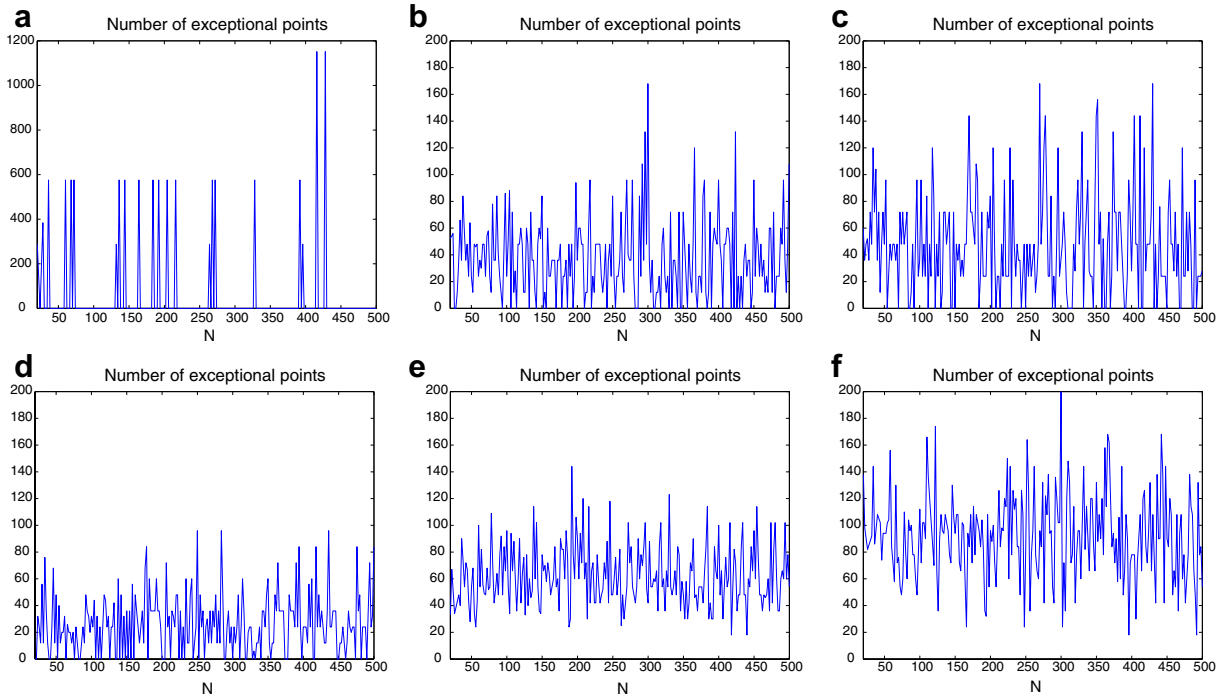


Fig. 15. Number of exceptional points for three-dimensional complex interface problems. The number of grid points ranging from 20^3 to 500^3. The total number of exceptional points is less than 200 for most cases: (a) 8 balls, (b) ellipsoid, (c) donuts, (d) peanut, (e) banana and (f) popcorn.

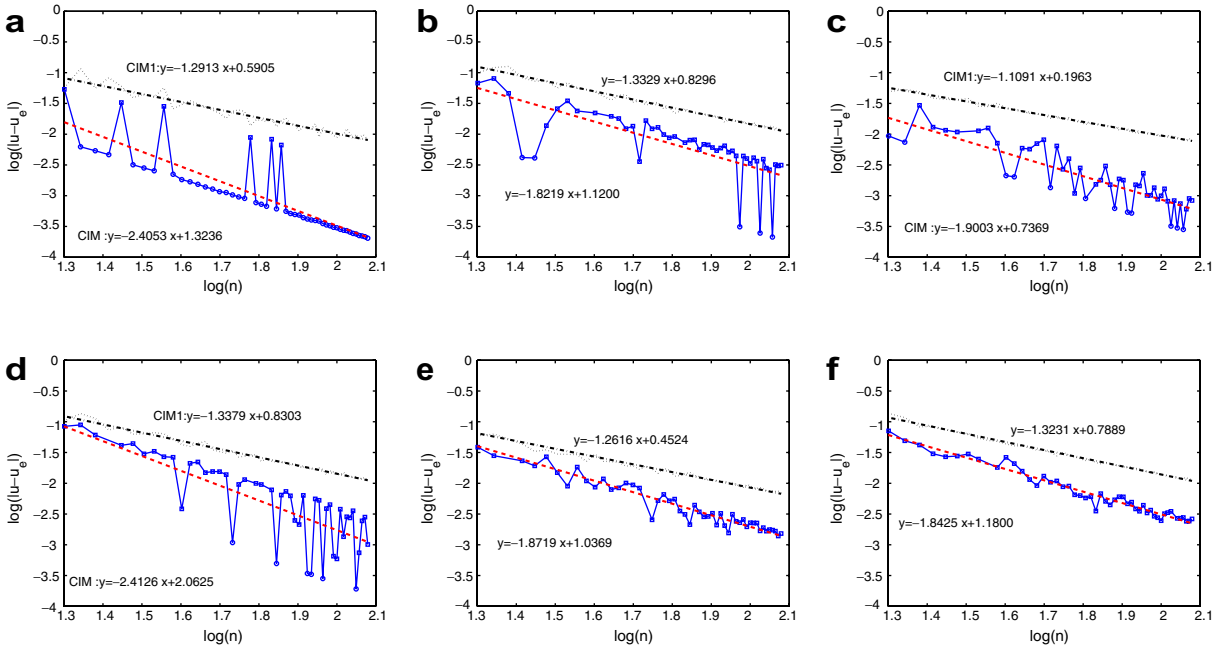


Fig. 16. This is the log–log plot of N versus the absolute errors. The top straight line is the least square fit of the error from CIM1. The bottom straight line is the least square fit of the error from the hybrid CIM. The x-axis is $\log_{10}(N)$ with N ranging from 20 to 120; the y-axis is $\log_{10}(|u-u_e|)$ with range $[-4,-1]$. The hybrid CIM is about second order: (a) 8 balls, (b) ellipsoid, (c) donuts, (d) peanut, (e) banana and (f) popcorn.

5.5.3. A comparison study with FIIM

In the paper of FIIM [28], a complex flower-like interface was proposed as a prototype to study unstable interface problems. The interface is defined by

$$\phi(r,\theta) = r - r_0 - 0.2\sin(\omega\theta),$$

where $r = \sqrt{(x-x_c)^2+(y-y_c)^2}$, $\theta = \arctan\left((y-y_c)/(x-x_c)\right)$, and $x_c = y_c = 0.2/\sqrt{20}$. We consider two cases: case (a), $r_0 = 0.5$, $\omega = 5$, case (b), $r_0 = 0.4$, $\omega = 12$, see Fig. 17. The coefficients is given by

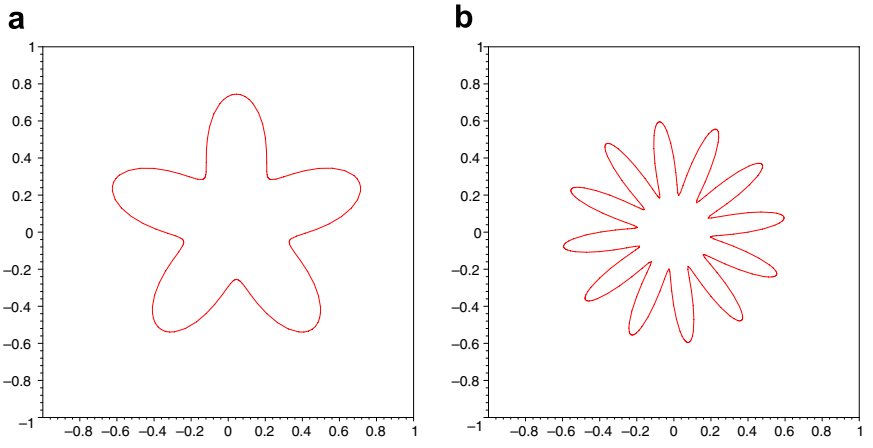


Fig. 17. The flower interfaces in two dimensions (a) $r_0 = 0.5$, $\omega = 5$ and (b) $r_0 = 0.4$, $\omega = 12$.

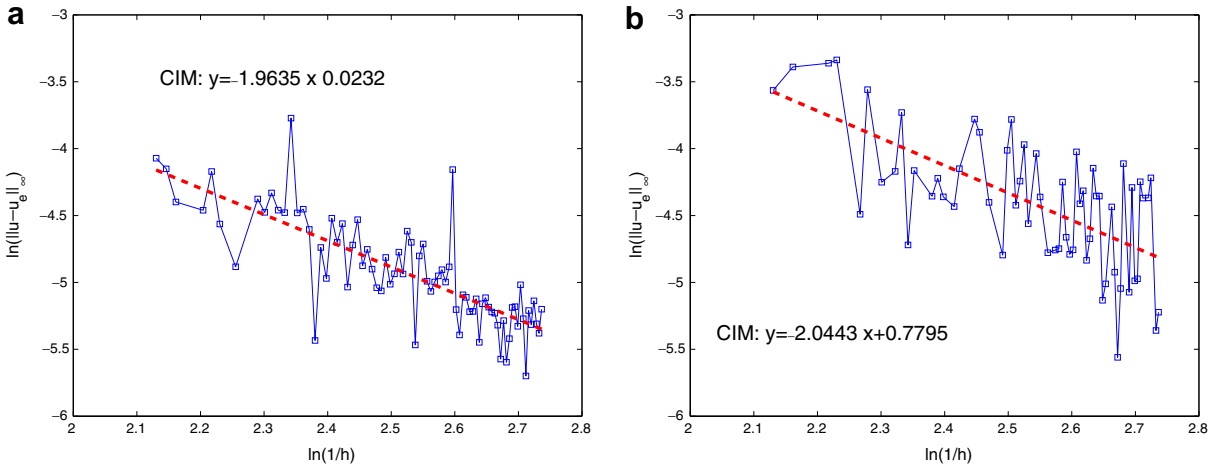


Fig. 18. Convergence test for hybrid CIM. The x-axis is $\log(1/h)$ with range $[2, 2.8]$; the y-axis is $\log(\|u - u_e\|_\infty)$ with range $[-6, -3]$: (a) $r_0 = 0.5$, $\omega = 5$, $\varepsilon^- = 1000$, $\varepsilon^+ = 1$ and (b) $r_0 = 0.4$, $\omega = 12$, $\varepsilon^- = 100$, $\varepsilon^+ = 1$.

$$\varepsilon(x,y) = \begin{cases} \varepsilon^-, & (x,y) \in \Omega^-, \\ \varepsilon^+, & (x,y) \in \Omega^+, \end{cases}$$

where $\varepsilon^- = 1000, \varepsilon^+ = 1$ for case (a), $\varepsilon^- = 100, \varepsilon^+ = 1$ for case (b). The test function u is given by

$$u(x,y) = \begin{cases} (x^2 + y^2), & (x,y) \in \Omega^-, \\ (x^2 + y^2)^2 - 0.1 \log\left(2\sqrt{x^2 + y^2}\right), & (x,y) \in \Omega^+. \end{cases}$$

The source term and the jump conditions are derived from Eq. (1).

Fig. 18 is the convergence plot of the hybrid CIM for N ranging from 270 to 1090 with increment $\Delta N = 10$. From the least square fit of the absolute errors, we see the hybrid CIM is nearly second order for these complicated interfaces. For case (a), we see there are two runs having larger errors due to the pollution from the exceptional points. Nevertheless, the tendency of these two errors is still nearly second order.

As we compare the absolute errors of hybrid CIM and FIIM. For case (a), the best result quoted from Fig. 5b of FIIM for $N \in (270,1090)$ is about $e^{-8.5} \sim 2 \times 10^{-4}$ at $N \sim 960$, whereas the absolute error produced by CIM at the same resolution is 2.533×10^{-6}. Moreover, the hybrid CIM produces error below 2×10^{-4} for all $N \geqslant 270$. For case (b), the best result quoted from Fig. 6 of FIIM for $N \in (270,1090)$ is about $e^{-8.5} \sim 2 \times 10^{-4}$ at resolutions $N \sim 590$ and $N \sim 1080$, whereas the absolute errors produced by CIM are 3.9650×10^{-5} and 4.369×10^{-6} at $N = 590$ and $N = 1080$, respectively. Moreover, the error produced by the hybrid CIM is less than 2×10^{-4} for all $N \geqslant 650$.

6. Conclusion

In this paper, we deal with elliptic complex interface problems in any dimensions. We propose a first-order, a second-order and a hybrid coupling interface method (CIM1, CIM2 and CIM, respectively) under Cartesian grid for solving such elliptic complex interface problems. The ingredients of CIM (first-order and second-order) consist of (1) dimension-by-dimension discretization, (2) decomposition of one-dimensional jump data, and (3) coupling and one-side interpolation. Solvability of the coupling equation is shown. The key is that such a coupling reduces the number of interpolation points, and results in a compact finite difference scheme. Thus, it is more flexible to handle complex interface problems. In the hybrid CIM, we classify the underlying Cartesian grids into interiors, normals and exceptionals, where a standard central finite difference method, the CIM2 and the CIM1 are applied respectively. Due to the facts that the number of exceptional grids is O(1) in most applications, the CIM can maintain second-order accuracy globally.

The resulting linear systems is solved by an AMG iterator with reduction rates 0.1–0.7 in each V-cycle. The computational cost is slightly above linear. Numerical results show that the CIM can handle complex interface problems in any dimensions.

Appendix A

A.1. Cancellation in the ghost fluid method

In the ghost fluid method (GFM) [36], a term $[\varepsilon u_t]/h$ is neglected. We expect that it should produce an $O(1/h)$ truncation error and result in non-convergence. However, there is a convergence proof for GFM by Liu and Sideris [37]. Below, we construct an example to show the $O(1/h)$ truncation does appear. However, a cancellation from opposite signs of the truncation errors causes convergence.

The domain is chosen to be $[-1,1]\times[0,2]$. The interface Γ is a straight line given by

$$\phi(x,y) = x\cos\theta + y\sin\theta = 0.$$

The source function is chosen to be $f\equiv 0$. The coefficient $\varepsilon(\cdot)$ is

$$\varepsilon(x,y) = \begin{cases} \varepsilon^-, & (x,y)\in\Omega^-, \\ \varepsilon^+, & (x,y)\in\Omega^+, \end{cases}$$

with $\varepsilon^- = 2$ and $\varepsilon^+ = 80$. We shall construct an exact solution u such that it is linear on both sides of the interface with $[u] = [\varepsilon u_n] = 0$, but $[\varepsilon u_t] = \varepsilon^+ - \varepsilon^- \neq 0$. Such a u is given by

$$u(x,y) = \begin{cases} \xi/\varepsilon^- + \eta & (x,y)\in\Omega^-, \\ \xi/\varepsilon^+ + \eta & (x,y)\in\Omega^+, \end{cases}$$

where $\xi = x\cos\theta + y\sin\theta$, $\eta = -x\sin\theta + y\cos\theta$.

We choose small $\theta = 0.01745$ so that the interface Γ does not intersect any vertical grid line before $N\approx 60$. In this case, the truncation error (the residue) is $O(1/h)$ and the true error looks like $O(\log h)$. The method seems not converge. However, if we further refine the mesh. The interface will eventually intersect vertical grid lines. At this moment, both the truncation error and the true error shoot up, but an truncation error with opposite sign is produced at the intersection of the interface with the vertical grid line. This causes an cancellation and this absolute error decreases as mesh is further refined, despite the truncation error is still $O(1/h)$. If we further refine the mesh, more vertical intersections occur, at which truncation errors with opposite sign are produced. The cancellation can be seen from the decreasing of the sum of the truncation errors. Thus, this numerical investigation suggests that the convergence is due to a cancellation (Figs. 19 and 20).

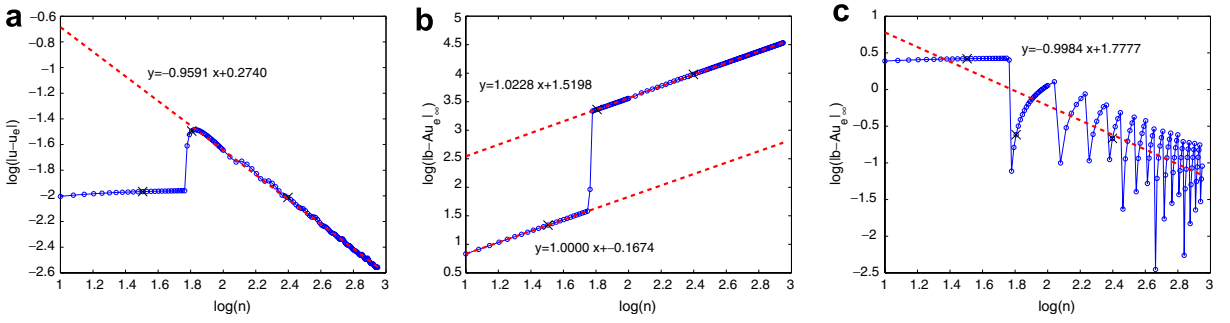


Fig. 19. The log–log plots of various errors versus N. The true error $\|u-u_e\|_\infty = O(h)$, the maximum norm of the residue (truncation error) grows like $O(1/h)$, whereas the sum of the residue decreases like $O(h)$: (a) the true error $\|u-u_e\|_\infty$, (b) maximum norm of the residule and (c) sum of the residule.

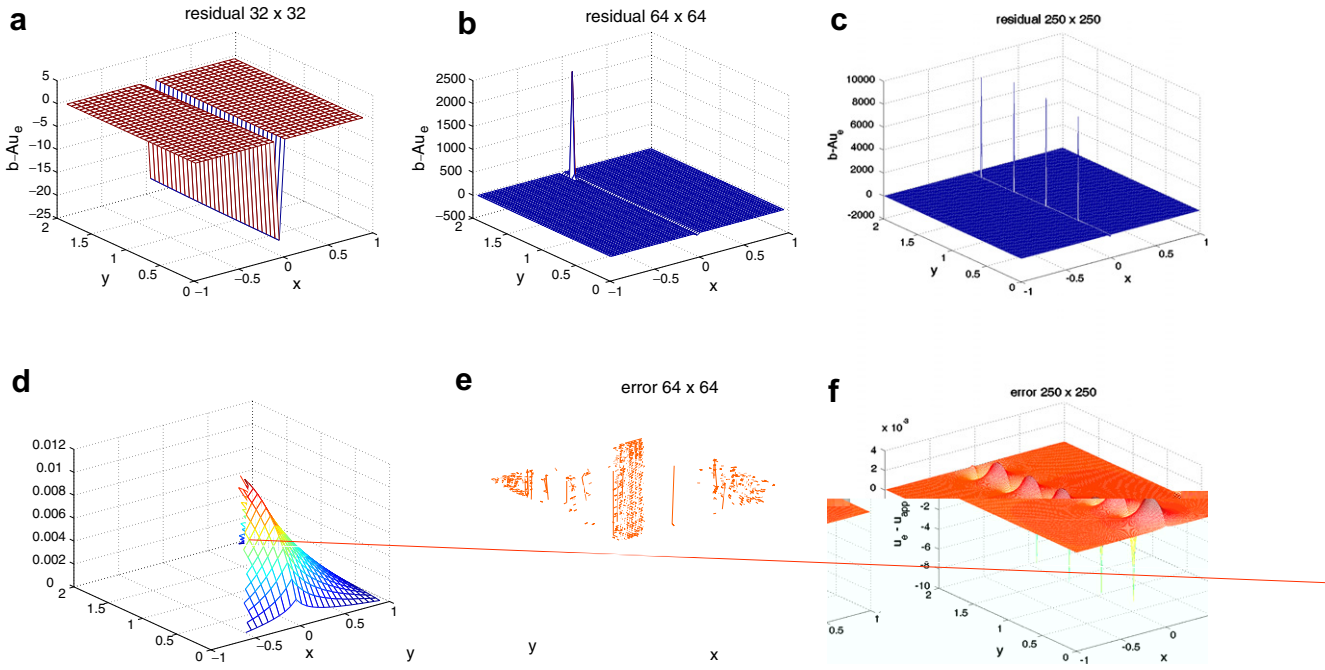


Fig. 20. The residuals (top) and the corresponding true errors (bottom) before and after the occurrence of the intersection of the interface with the vertical grid lines. (a,d): no intersection, (b,e): one intersection, (c,f): four intersections. (a) Residue at mesh 32×32, (b) residue at mesh 64×64, (c) residue at mesh 250×250, (d) true error at mesh 32×32, (e) true error at mesh 64×64 and (f) true error at mesh 250×250.

A.2. Non-singularity of the coupling matrix $\boldsymbol{M}$

In this subsection, we shall show the coupling matrix $\mathbf{M}$ for the CIM2 defined by (40) is invertible, provided $\kappa h < \text{Const}$. Here, κ is the total curvature. The proof for the invertibility of the coupling matrix $\bar{M}$ for the CIM1 is the same. We neglect it. Below, the parameters are defined by (12).

Proposition 1. *The diagonal entries* $m_{k,k}$ *of the matrix* $\mathbf{M}$ *is bounded by* 1 *and* $\frac{\varepsilon_k^+}{\varepsilon_k^-}$.

Proof. The diagonal entries of $\mathbf{M}$ are

$$\begin{aligned} m_{k,k} &= 1 + \left(\frac{1}{2} + \alpha_k\right)(\beta_k + \beta_k^2)(\rho_k^+ - \rho_k^-)(\mathbf{t}_k \cdot \mathbf{e}_k)^2 \\ &:= 1 + (r_k - 1)(\mathbf{t}_k \cdot \mathbf{e}_k)^2. \end{aligned}$$

By the Lemma below, we get r_k is bounded by 1 and $\frac{\varepsilon_k^+}{\varepsilon_k^-}$. Using $|\mathbf{t}_k \cdot \mathbf{e}_k| \leqslant 1$, we get $m_{k,k}$ is also bounded by 1 and $\frac{\varepsilon_k^+}{\varepsilon_k^-}$. □

Lemma 1. *Following the definition of the parameters in Section 4, let r be defined by*

$$r = 1 + \left(\frac{1}{2} + \alpha\right)(\beta + \beta^2)(\rho^+ - \rho^-).$$

Then r is a monotonic function in α *for* $0 \leqslant \alpha \leqslant 1$, *and r is bounded by* $\varepsilon^+/\varepsilon^-$ *and* 1.

Proof. For simplification of notation below, let us abbreviate $(\frac{1}{2} + \alpha)(\beta + \beta^2)$ and $(\frac{1}{2} + \beta)(\alpha + \alpha^2)$ by A and B, respectively. We have

$$r = 1 + \left(\frac{1}{2} + \alpha\right)(\beta + \beta^2)(\rho^+ - \rho^-) = \frac{(A+B)\varepsilon^+}{A\varepsilon^- + B\varepsilon^+} = \frac{(1+2\alpha\beta)\varepsilon^+}{A\varepsilon^- + B\varepsilon^+}.$$

Recall $\beta = 1 - \alpha$. Substituting this into the above formula, then taking the derivative with respect to α, we get

$$\frac{\mathrm{d}r}{\mathrm{d}\alpha} = \frac{2\varepsilon^+(2\alpha + 2\alpha^2 - 8\alpha^3 + 4\alpha^4 + 3)(\varepsilon^- - \varepsilon^+)}{(A\varepsilon^- + B\varepsilon^+)^2}.$$

The denominator is positive, while the numerator $(2\alpha + 2\alpha^2 - 8\alpha^3 + 4\alpha^4 + 3) = 2\alpha(1-\alpha) + 4\alpha^2(1-\alpha)^2 + 3 \geqslant 3$ for $0 \leqslant \alpha \leqslant 1$. Hence the extremal values of r occur when $\alpha = 0$ and 1, they are $\varepsilon^+/\varepsilon^-$ and 1, respectively. □

Let us denote $\det \mathbf{M}$ by Δ. It is a function of $\mathbf{n}_1, \ldots, \mathbf{n}_d, \varepsilon_1^\pm, \ldots, \varepsilon_d^\pm, \alpha_1, \ldots, \alpha_d$ where $\mathbf{n}_k$ is the normal of the interface Γ at the intersection point $\widehat{\mathbf{x}}_k$. We shall use perturbation method to find a lower bound of Δ. First, we study the flat case:

$$\mathbf{n}_1 = \cdots = \mathbf{n}_d = \mathbf{n},\ \varepsilon_1^- = \cdots = \varepsilon_d^- = \varepsilon^-,\ \varepsilon_1^+ = \cdots = \varepsilon_d^+ = \varepsilon^+. \tag{43}$$

Under this flat assumption, the matrix $\mathbf{M}$ in three dimension can be expressed as

$$\mathbf{M} = \begin{bmatrix} 1 + (r_1 - 1)(n_2^2 + n_3^2) & -(r_1 - 1 - \mu_1)n_1 n_2 & -(r_1 - 1 - \mu_1)n_1 n_3 \\ -(r_2 - 1 - \mu_2)n_2 n_1 & 1 + (r_2 - 1)(n_1^2 + n_3^2) & -(r_2 - 1 - \mu_2)n_2 n_3 \\ -(r_3 - 1 - \mu_3)n_3 n_1 & -(r_3 - 1 - \mu_3)n_3 n_2 & 1 + (r_3 - 1)(n_2^2 + n_3^2) \end{bmatrix},$$

where $\mathbf{n} = (n_1, n_2, n_3)$, $\mu_j = \alpha_j(\beta_j + \beta_j^2)(\rho_j^+ - \rho_j^-), j = 1, 2, 3$.

Theorem 1. *Under the flat assumption (43),*

$$\Delta \geqslant \min\left\{1, \left(\frac{\varepsilon^+}{\varepsilon^-}\right)^{d-1}\right\}.$$

Proof. There are two cases:

Case 1: $\varepsilon^- \leqslant \varepsilon^+$. In this case, from the definitions of μ_j, r_j and (12), it is easy to see that $r_j \geqslant 1$ and $0 \leqslant \mu_j \leqslant r_j - 1, j = 1, 2, 3$. We differentiate Δ in μ_1 to get

$$\frac{\partial \Delta}{\partial \mu_1} = (1 + (1 - n_3^2)\mu_3)(r_2 - 1 - \mu_2)n_2^2 + (1 + (1 - n_2^2)\mu_2)(r_3 - 1 - \mu_3)n_3^2 + (n_2^2 + n_3^2)(r_2 - 1 - \mu_2) \times (r_3 - 1 - \mu_3), \geqslant 0$$

for μ_1 satisfying $0 \leqslant \mu_1 \leqslant r_1 - 1$. Similarly, we can get $\partial\Delta/\partial\mu_j \geqslant 0$ for $0 \leqslant \mu_j \leqslant r_j - 1$.
Case 2: $\varepsilon^- > \varepsilon^+$. In this case, we have $0 < r_j < 1$ and $r_j - 1 < \mu_j < 0$. And we can get

$$\frac{\partial \Delta}{\partial \mu_1} = n_2^2 r_3(r_2 - 1 - \mu_2) + n_3^2 r_2(r_3 - 1 - \mu_3) - n_2^2 n_3^2 \mu_2(r_3 - 1 - \mu_3) - n_2^2 n_3^2 \mu_3(r_2 - 1 - \mu_2) < 0.$$

Similarly, we also get $\partial\Delta/\partial\mu_j < 0$ for $r_j - 1 < \mu_j < 0$.
With this monotonicity property of Δ, we obtain

$$\Delta(\mu_1, \mu_2, \mu_3) \geqslant \Delta(0, \mu_2, \mu_3) \geqslant \Delta(0, 0, \mu_3) \geqslant \Delta(0, 0, 0) = r_2 r_3 n_1^2 + r_1 r_3 n_2^2 + r_1 r_2 n_3^2.$$

The last equality is obtained by directly calculation. By Proposition 1, for each $k, r_k \geqslant \min\{1, \varepsilon^+/\varepsilon^-\}$. Under the constraint $\|\mathbf{n}\|^2 = 1$, we obtain

$$r_2r_3n_1^2 + r_1r_3n_2^2 + r_1r_2n_3^2 \geqslant \min\left\{1, \left(\frac{\varepsilon^+}{\varepsilon^-}\right)^2\right\}. \qquad \square$$

Theorem 2. *At a normal on-front grid point* $\mathbf{x}$, *the determinant* Δ *of the matrix* $\mathbf{M}$ *satisfies*

$$\Delta \geqslant \min\left\{1, \left(\frac{\varepsilon^+}{\varepsilon^-}\right)^{d-1}\right\} - C(\|\nabla\mathbf{n}\|_{\infty,\mathbf{x}} + \|\nabla\varepsilon^{\pm}\|_{\infty,\mathbf{x}})h,$$

where $\mathbf{n}$ *is the normal of the interface* Γ, *and* C *is a positive constant. The notation* $\|\cdot\|_{\infty,\mathbf{x}}$ *denotes for taking maximal values on the interface* Γ *near the on-front grid point* $\mathbf{x}$.

Proof. The function Δ is a smooth function of $\mathbf{n}_1, \ldots, \mathbf{n}_d, \varepsilon_1^{\pm}, \ldots, \varepsilon_d^{\pm}$. Using first variation of Δ in a neighborhood of $\mathbf{n}_1 = \cdots = \mathbf{n}_d = \mathbf{n}$, $\varepsilon_1^- = \cdots = \varepsilon_d^- = \varepsilon^-$, $\varepsilon_1^+ = \cdots = \varepsilon_d^+ = \varepsilon^+$, we obtain our theorem. $\square$

A.3. Pseudocode

Algorithm 5. First-order coupling interface method (CIM1) in d dimensions

procedure CIM1 ($\mathbf{x}_i$,h)
  for $s = -1, 1$
    for $k = 1 : d$
      $(\bar{D}_k u_i, \hat{\mathbf{x}}_k, \bar{\rho}_k^{\pm}, \bar{\varepsilon}_k, \beta_k, \gamma_k) \leftarrow$ 1stDerivative$(\mathbf{x}_i, h\mathbf{e}_k, s)$
      if $\gamma_k = 1$ **then**
        $\mathbf{n}_k \leftarrow$ the unit normal vector to interface at $\hat{\mathbf{x}}_k$
        $\mathbf{t}_k \leftarrow$ the unit tangential vector(the projection of the Euclidean unit vector onto $\mathbf{n}_k$).
        $([u]_{\hat{\mathbf{x}}_k}, [\varepsilon\nabla u \cdot \mathbf{n}_k]_{\hat{\mathbf{x}}_k}, [\nabla u \cdot \mathbf{t}_k]_{\hat{\mathbf{x}}_k}) \leftarrow$ the jump data locate at $\hat{\mathbf{x}}_k$ from $\mathbf{x}_i$ side to the other side
        $\bar{J}_k \leftarrow -\left(s\rho_k^+[u]_{\hat{\mathbf{x}}_k} + \beta_k h\left(\frac{[\varepsilon\nabla u\cdot\mathbf{n}_k]_{\hat{\mathbf{x}}_k}}{\bar{\varepsilon}_k}(\mathbf{n}_k\cdot\mathbf{e}_k) + \rho_k^+[\nabla u\cdot\mathbf{t}_k]_{\hat{\mathbf{x}}_k}(\mathbf{t}_k\cdot\mathbf{e}_k)\right)\right)$
        $\bar{T}_k \leftarrow$
      else
        $\mathbf{n}_k \leftarrow \mathbf{e}_k, \mathbf{t}_k \leftarrow \mathbf{0}, \bar{J}_k \leftarrow 0, \bar{T}_k \leftarrow 0$
      end if
      $b_k = -\beta_k(\rho_k^+ - \rho_k^-)$
    end for
    for $k = 1 : d$ **do**
      for $\ell = 1 : d$ **do**
        if $k = \ell$ **then**
          $(\bar{M})_{k,\ell} \leftarrow 1 - \gamma_k b_k(\mathbf{t}_k\cdot\mathbf{e}_k)^2$ ▷$\bar{M}$ is a $d\times d$ matrix.
        else
          $(\bar{M})_{k,\ell} \leftarrow -\gamma_k\gamma_j b_k(\mathbf{t}_k\cdot\mathbf{e}_k)(\mathbf{t}_k\cdot\mathbf{e}_j)$
        end if
      end for
      $\bar{v}_k \leftarrow \frac{1}{h}(\bar{D}_k u_i + b_k\bar{T}_k u_i(\mathbf{t}_k\cdot\mathbf{e}_k) + \bar{J}_k)$ ▷$\bar{v}$ is a $d\times 1$ vector.
    end for
    $[\frac{\partial}{\partial x_1}u(\mathbf{x}+\frac{1}{2}sh\mathbf{e}_1), \frac{\partial}{\partial x_2}u(\mathbf{x}+\frac{1}{2}sh\mathbf{e}_2), \ldots, \frac{\partial}{\partial x_d}u(\mathbf{x}+\frac{1}{2}sh\mathbf{e}_d)]^{\mathrm{T}} = \bar{M}^{-1}\bar{v}$
  end for
  $-\nabla\cdot(\varepsilon(\mathbf{x})\nabla u(\mathbf{x})) = -\sum_{k=1}^{d}\varepsilon(\mathbf{x})(\frac{\partial}{\partial x_k}u(\mathbf{x}+\frac{1}{2}h\mathbf{e}_k) - \frac{\partial}{\partial x_k}u(\mathbf{x}-\frac{1}{2}h\mathbf{e}_k))$
end procedure

Algorithm 6. Second-order coupling interface method (CIM2) in d dimensions

procedure CIM2 $(\mathbf{x}_i, h)$
  for $k = 1 : d$ **do**
    $(L_k^{(s)} u_i, \hat{\mathbf{x}}_k, \rho_k^{\pm}, \hat{\varepsilon}_k, \beta_k, s_k) \leftarrow$ 2ndDerivative$(\mathbf{x}_i, h\mathbf{e}_k)$
    if $s_k \neq 0$ **then**
      $\mathbf{n}_k \leftarrow$ the unit normal vector to interface at $\hat{\mathbf{x}}_k$
      $\mathbf{t}_k \leftarrow$ the unit tangential vector (the projection of the Euclidean unit vector onto $\mathbf{n}_k$).
      $([u]_{\hat{\mathbf{x}}_k}, [\varepsilon \nabla u \cdot \mathbf{n}_k]_{\hat{\mathbf{x}}_k}, [\nabla u \cdot \mathbf{t}_k]_{\hat{\mathbf{x}}_k}) \leftarrow$ the jump data locate at $\hat{\mathbf{x}}_k$ from $\mathbf{x}_i$ side to the other side
      $J_k \leftarrow -((1+2\beta_k)\rho_k^+ [u]_{\hat{\mathbf{x}}_k} + s_k(\beta_k + \beta_k^2) h(\frac{[\varepsilon \nabla u \cdot \mathbf{n}_k]_{\hat{\mathbf{x}}_k}}{\hat{\varepsilon}_k}(\mathbf{n}_k \cdot \mathbf{e}_k) + \rho_k^+ [\nabla u \cdot \mathbf{t}_k]_{\hat{\mathbf{x}}_k}(\mathbf{t}_k \cdot \mathbf{e}_k)))$
    else
      $\mathbf{n}_k \leftarrow \mathbf{e}_k, \mathbf{t}_k \leftarrow \mathbf{0}$, $J_k \leftarrow 0$
    end if
    $b_k = -(\beta_k + \beta_k^2)(\rho_k^+ - \rho_k^-)$
  end for
  for $k = 1 : d$ **do**
    for $\ell = 1 : d$ **do**
      if $k = \ell$ **then**
        $(M)_{k,\ell} \leftarrow 1 - |s_k|(\frac{3}{2} - \beta_k) b_k (\mathbf{t}_k \cdot \mathbf{e}_k)^2$ ▷M is a $d \times d$ matrix.
      else
        $(M)_{k,\ell} \leftarrow -\frac{1}{2} s_k s_j b_k (\mathbf{t}_k \cdot \mathbf{e}_k)(\mathbf{t}_k \cdot \mathbf{e}_j)$
      end if
    end for
    $v_k \leftarrow \frac{1}{h^2}(L_k u(\mathbf{x}) + s_k b_k T_k u_i (\mathbf{t}_k \cdot \mathbf{e}_k) + J_k)$ ▷v is a $d \times 1$ vector.
  end for
  $\left[\frac{\partial^2}{\partial x_1^2} u(\mathbf{x}), \frac{\partial^2}{\partial x_2^2} u(\mathbf{x}), \ldots, \frac{\partial^2}{\partial x_d^2} u(\mathbf{x})\right]^{\mathrm{T}} = M^{-1} v$
  $-\nabla \cdot (\varepsilon(\mathbf{x}) \nabla u(\mathbf{x})) = -\sum_{k=1}^{d} \frac{1}{h^2}(D^{(s_k)} \varepsilon(\mathbf{x}) D^{(s_k)} u(\mathbf{x})) + \varepsilon(\mathbf{x}) \frac{\partial^2}{\partial x_k^2} u(\mathbf{x})$
end procedure

Algorithm 7. Grid labeling

Require: Computational Domain D, levelset function ϕ, the dielectric function ε^+ and ε^-, $\mathbf{x} \in \mathbb{R}^p$
function $G =$ GridLabeling $(\phi, \varepsilon^+, \varepsilon^-)$
  for all $\mathbf{x} \in D$ **do**
    if $\phi(\mathbf{x}) > \epsilon$ **then**
      $G(\mathbf{x}) \leftarrow 1$
    else if $\phi(\mathbf{x}) < -\epsilon$ **then**
      $G(\mathbf{x}) \leftarrow 0$
    else
      $t \leftarrow 0$
      for $j = 1 : p$ **do**
        if $\phi(\mathbf{x} \pm h\mathbf{e}_j) > \epsilon$ **then**
          $G(\mathbf{x}) \leftarrow 1$, $t \leftarrow 1$, exit for
        else if $\phi(\mathbf{x} \pm h\mathbf{e}_j) < -\epsilon$ **then**
          $G(\mathbf{x}) \leftarrow 0$, $t \leftarrow 1$, exit for
        end if
      end for
      if $t = 0$ **then**

```
            if ε⁺(x) > ε⁻(x) then
                G(x) ← 1
            else
                G(x) ← 0
            end if
        end if
    end if
  end for
end function
```

Algorithm 8. Classification of all points

```
Require: Computational Domain D, Grid labeling G, x ∈ ℝ^p
function CL = CLASSFICATION(G)
  for all x ∈ D do
    n ← 0
    for j = 1 : p, s = −1, 1 do
      if G(x) ≠ G(x + she_j) then
        n ← 1, exit for
      end if
    end for
    if n = 0 then                                          ▷ Interior points
      CL(x) ← 0, exit for
    end if
    CL(x) ← 2
    for j = 1 : p do                                       ▷ For first kind of exceptional points
      if G(x) ≠ G(x + he_j) and G(x) ≠ G(x − he_j) then
        CL(x) ← 1, exit for
      end if
      if G(x) ≠ G(x + he_j) and G(x) = G(x + 2he_j) then
        CL(x) ← 1, exit for
      end if
      if G(x) ≠ G(x − he_j) and G(x) = G(x − 2he_j) then
        CL(x) ← 1, exit for
      end if
    end for
    for j = 1 : p, s = −1, 1 do                            ▷ For second kind of exceptional points
      if G(x) ≠ G(x + she_j) then
        for k = 1 : p, k ≠ j, t = −1, 1 do
          if G(x + t he_k) ≠ G(x) and G(x − t he_k − she_j) ≠ G(x) then
            CL(x) ← 1
          end if
        end for
      end if
    end for
  end for
end function
```

References

[1] L. Adams, T.P. Chartier, New geometric immersed interface multigrid solvers, SIAM Journal on Scientific Computing 25 (2004) 1516–1533.
[2] L. Adams, T.P. Chartier, A comparison of algebraic multigrid and geometric immersed interface multigrid methods for interface problems, SIAM Journal on Scientific Computing 26 (2005) 762–784.
[3] L. Adams, Z.L. Li, The immersed interface/multigrid methods for interface problems, SIAM Journal on Scientific Computing 24 (2002) 463–479.
[4] J.B. Bell, C.N. Dawson, G.R. Shubin, An unsplit, higher-order Godunov method for scalar conservation-laws in multiple dimensions, Journal of Computational Physics 74 (1988) 1–24.
[5] P.A. Berthelsen, A decomposed immersed interface method for variable coefficient elliptic equations with non-smooth and discontinuous solutions, Journal of Computational Physics 197 (2004) 364–386.
[6] Z.M. Chen, J. Zou, Finite element methods and their convergence for elliptic and parabolic interface problems, Numerische Mathematik 79 (1998) 175–202.
[7] C.S. Peskin, The immersed boundary method, Acta Numerica (2002) 1–39.
[8] S.Z. Deng, K. Ito, Z.L. Li, Three-dimensional elliptic solvers for interface problems and applications, Journal of Computational Physics 184 (2003) 215–243.
[9] M.A. Dumett, J.P. Keener, An immersed interface method for anisotropic elliptic problems on irregular domains in 2d, Numerical Methods for Partial Differential Equations 21 (2005) 397–420.
[10] R.P. Fedkiw, T. Aslam, B. Merriman, S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method), Journal of Computational Physics 152 (1999) 457–492.
[11] A.L. Fogelson, J.P. Keener, Immersed interface methods for Neumann and related problems in two and three dimensions, SIAM Journal on Scientific Computing 22 (2001) 1630–1654.
[12] F. Gibou, R. Fedkiw, A fourth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problem, Journal of Computational Physics 202 (2005) 577–601.
[13] F. Gibou, R.P. Fedkiw, L.T. Cheng, M.J. Kang, A second-order-accurate symmetric discretization of the Poisson equation on irregular domains, Journal of Computational Physics 176 (2002) 205–227.
[14] J. Glimm, O.A. Mcbryan, A computational model for interfaces, Advances in Applied Mathematics 6 (1985) 422–435.
[15] R. Glowinski, T.W. Pan, J. Periaux, A fictitious domain method for Dirichlet problem and applications, Computer Methods in Applied Mechanics and Engineering 111 (1994) 283–303.
[16] M. Holst, R.E. Kozack, F. Saied, S. Subramaniam, Multigrid-based Newton iterative method for solving the full nonlinear Poisson–Boltzmann equation, Biophysical Journal 66 (1994) A130.
[17] M. Holst, F. Saied, Multigrid solution of the Poisson–Boltzmann equation, Journal of Computational Chemistry 14 (1993) 105–113.
[18] J.G. Huang, J. Zou, A mortar element method for elliptic problems with discontinuous coefficients, IMA Journal of Numerical Analysis 22 (2002) 549–576.
[19] K. Ito, Z.L. Li, Solving a nonlinear problem in magneto-rheological fluids using the immersed interface method, Journal of Scientific Computing 19 (2003) 253–266.
[20] K. Ito, Z.L. Li, Y. Kyei, Higher-order, cartesian grid based finite difference schemes for elliptic equations on irregular domains, SIAM Journal on Scientific Computing 27 (2005) 346–367.
[21] H. Johansen, P. Colella, A cartesian grid embedded boundary method for Poisson equations on irregular domains, Journal of Computational Physics 147 (1998) 60–85.
[22] J.D. Kandilarov, A Rothe-immersed interface method for a class of parabolic interface problems, Numerical Analysis and its Applications 3401 (2005) 328–336.
[23] J.D. Kandilarov, L.G. Vulkov, The immersed interface method for a nonlinear chemical diffusion equation with local sites of reactions, Numerical Algorithms 36 (2004) 285–307.
[24] M.C. Lai, Z.L. Li, X.B. Lin, Fast solvers for 3d Poisson equations involving interfaces in a finite or the infinite domain, Journal of Computational and Applied Mathematics 191 (2006) 106–125.
[25] L. Lee, R.J. Leveque, An immersed interface method for incompressible Navier–Stokes equations, SIAM Journal on Scientific Computing 25 (2003) 832–856.
[26] R.J. Leveque, Z.L. LI, The immersed interface method for elliptic-equations with discontinuous coefficients and singular sources, SIAM Journal on Numerical Analysis 31 (1994) 1019–1044.
[27] Z.L. Li, Immersed interface methods for moving interface problems, Numerical Algorithms 14 (1997) 269–293.
[28] Z.L. Li, A fast iterative algorithm for elliptic interface problems, SIAM Journal on Numerical Analysis 35 (1998) 230–254.
[29] Z.L. Li, An overview of the immersed interface method and its applications, Taiwanese Journal of Mathematics 7 (2003) 1–49.
[30] Z.L. Li, K. Ito, Maximum principle preserving schemes for interface problems with discontinuous coefficients, SIAM Journal on Scientific Computing 23 (2001) 339–361.
[31] Z.L. Li, M.C. Lai, The immersed interface method for the Navier–Stokes equations with singular forces, Journal of Computational Physics 171 (2001) 822–842.
[32] Z.L. Li, T. Lin, X.H. Wu, New cartesian grid methods for interface problems using the finite element formulation, Numerische Mathematik 96 (2003) 61–98.
[33] Z.L. Li, D.S. Wang, J. Zou, Theoretical and numerical analysis on a thermo-elastic system with discontinuities, Journal of Computational and Applied Mathematics 92 (1998) 37–58.

[34] Z.L. Li, W.C. Wang, I.L. Chern, M.C. Lai, New formulations for interface problems in polar coordinates, SIAM Journal on Scientific Computing 25 (2003) 224–245.
[35] M.N. Linnick, H.F. Fasel, A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains, Journal of Computational Physics 204 (2005) 157–192.
[36] X.D. Liu, R.P. Fedkiw, M.J. Kang, A boundary condition capturing method for Poisson's equation on irregular domains, Journal of Computational Physics 160 (2000) 151–178.
[37] Xu-Dong Liu, Thomas C. Sideris, Convergence of the ghost fluid method for elliptic equations with interfaces, Mathematics of Computation 72 (244) (2003) 1731–1746.
[38] A. Mayo, The fast solution of Poissons and the biharmonic-equations on irregular regions, SIAM Journal on Numerical Analysis 21 (1984) 285–299.
[39] A. Mayo, Fast high-order accurate solution of Laplace equation on irregular regions, SIAM Journal on Scientific and Statistical Computing 6 (1985) 144–157.
[40] A. Mckenney, L. Greengard, A. Mayo, A fast Poisson solver for complex geometries, Journal of Computational Physics 118 (1995) 348–355.
[41] C.S. Peskin, Numerical-analysis of blood-flow in heart, Journal of Computational Physics 25 (1977) 220–252.
[42] J. Ruge, K. Stuben, Algebraic multigrid, in: S.F. Mccormick (Ed.), Multigrid Methods, vol. 4, SIAM, Philadelphia, 1987, pp. 73–130.
[43] G.R. Shubin, J.B. Bell, An analysis of the grid orientation effect in numerical-simulation of miscible displacement, Computer Methodsi In Applied Mechanics and Engineering 47 (1984) 47–71.
[44] A.N. Tikhonov, A.A. Samarskii, Homogeneous difference schemes, USSR Computational Mathematics and Mathematical Physics 1 (1962) 5–67.
[45] A.K. Tornberg, B. Engquist, Regularization techniques for numerical approximation of PDEs with singularities, Journal of Scientific Computing 19 (2003) 527–552.
[46] A.K. Tornberg, B. Engquist, Numerical approximations of singular source terms in differential equations, Journal of Computational Physics 200 (2004) 462–488.
[47] J.H. Walther, G. Morgenthal, An immersed interface method for the vortex-in-cell algorithm, Journal of Turbulence 3 (2002) 1–9.
[48] W.C. Wang, A jump condition capturing finite difference scheme for elliptic interface problems, SIAM Journal on Scientific Computing 25 (2004) 1479–1496.
[49] A. Wiegmann, K.P. Bube, The explicit-jump immersed interface method: Finite difference methods for PDEs with piecewise smooth solutions, SIAM Journal on Numerical Analysis 37 (2000) 827–862.
[50] J.J. Xu, Z.L. Li, J. Lowengrub, H.K. Zhao, A level-set method for interfacial flows with surfactant, Journal of Computational Physics 212 (2005) 590–616.
[51] S. Xu, Z.J. Wang, Systematic derivation of jump conditions for the immersed interface method in three-dimensional flow simulation, SIAM Journal on Scientific Computing 27 (2006) 1948–1980.
[52] X.Z. Yang, B. Li, Z.L. Li, The immersed interface method for elasticity problems with interfaces, Dynamics of Continuous Discrete and Impulsive Systems-Series A-Mathematical Analysis 10 (2003) 783–808.
[53] 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, Journal of Computational Physics 213 (2006) 1–30.