Journal of Computational Physics 229 (2010) 6405–6426

Contents lists available at ScienceDirect

Journal of Computational Physics

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

# A second order virtual node method for elliptic problems with interfaces and irregular domains

Jacob Bedrossian *, James H. von Brecht, Siwei Zhu, Eftychios Sifakis, Joseph M. Teran

*University of California-Los Angeles, Department of Mathematics, 520 Portola Plaza, Math Sciences Building 6363, Los Angeles, CA 90095, United States*

ARTICLE INFO

*Article history:*
Received 26 August 2009
Received in revised form 3 April 2010
Accepted 4 May 2010
Available online 24 May 2010

*Keywords:*
Elliptic interface problems
Embedded interface methods
Virtual node methods
Variational methods

ABSTRACT

We present a second order accurate, geometrically flexible and easy to implement method for solving the variable coefficient Poisson equation with interfacial discontinuities or on irregular domains, handling both cases with the same approach. We discretize the equations using an embedded approach on a uniform Cartesian grid employing virtual nodes at interfaces and boundaries. A variational method is used to define numerical stencils near these special virtual nodes and a Lagrange multiplier approach is used to enforce jump conditions and Dirichlet boundary conditions. Our combination of these two aspects yields a symmetric positive definite discretization. In the general case, we obtain the standard 5-point stencil away from the interface. For the specific case of interface problems with continuous coefficients, we present a discontinuity removal technique that admits use of the standard 5-point finite difference stencil everywhere in the domain. Numerical experiments indicate second order accuracy in $L^\infty$.



## 1. Introduction

Elliptic interface problems such as

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

$$[u] = a(\mathbf{x}) \quad \mathbf{x}\in\Gamma, \tag{2}$$

$$[\beta(\mathbf{x})\nabla u\cdot n] = b(\mathbf{x}) \quad \mathbf{x}\in\Gamma, \tag{3}$$

$$u = p(\mathbf{x}) \quad \mathbf{x}\in\partial\Omega_d, \tag{4}$$

$$\beta(\mathbf{x})\nabla u\cdot n = q(\mathbf{x}) \quad \mathbf{x}\in\partial\Omega_n \tag{5}$$

have a wide variety of applications in physics and engineering, and naturally arise when two dissimilar materials interact across a thin interface, either as quasistatic problems or in the discretization of time-dependent problems. Common examples include immiscible, incompressible fluids in contact and phase change problems. The interface $\Gamma$ is generally a co-dimension one closed curve that divides the domain into an interior $\Omega^-$ and an exterior region $\Omega^+$ such that $\Omega = \Omega^+\cup\Omega^-\cup\Gamma\subset\mathbb{R}^2$ (see Fig. 1). The scalar coefficient $\beta$ and the source term $f$ can exhibit discontinuities across $\Gamma$, but have smooth restrictions $\beta^+, f^+$ to $\Omega^+$ and $\beta^-, f^-$ to $\Omega^-$. We let $n(\mathbf{x})$ denote the outward unit normal to $\Omega^-$ at a point $\mathbf{x}\in\Gamma$, and define $[v](\mathbf{x}) := v^+(\mathbf{x}) - v^-(\mathbf{x}) := \lim_{\epsilon\to 0^+} v(\mathbf{x}+\epsilon n(\mathbf{x})) - \lim_{\epsilon\to 0^+} v(\mathbf{x}-\epsilon n(\mathbf{x}))$ as the "jump" of the quantity $v$ across the interface $\Gamma$. The relevant physics generally determine the jumps in the solution (2) and in the flux (3), as well as the boundary conditions on $\partial\Omega$. Unless stated otherwise, we assume the curves $\Gamma$, $\partial\Omega$ are smooth.

* Corresponding author. Tel.: +1 440 554 6984.
*E-mail addresses:* jacob.bedrossian@ucla.math.ucla.edu (J. Bedrossian), jub@math.ucla.edu (J.H. von Brecht), siwei@math.ucla.edu (S. Zhu), sifakis@math.ucla.edu (E. Sifakis), jteran@math.ucla.edu (J.M. Teran).

0021-9991/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2010.05.002

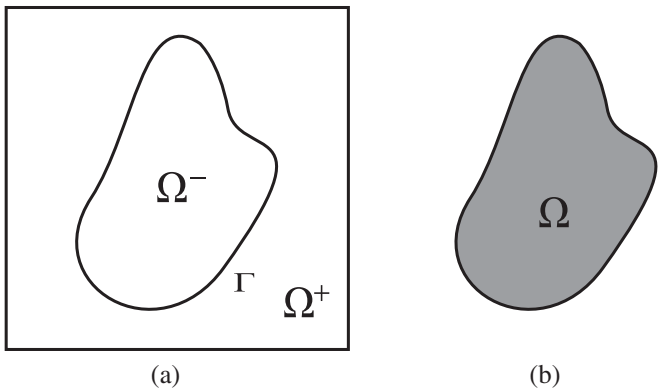



**Fig. 1.** Graphical depiction of the problems (1–5). The image on the left depicts the relevant domains for interface problems, and the image on the right depicts the domain for embedded boundary problems.

Due to irregular geometry of the interface in many physical phenomena, a natural approach to the numerical approximation is the finite element method (FEM) with unstructured meshes that conform to the geometry of $\Gamma$ [1–6]. However, meshing complex interface geometries can prove difficult and time-consuming when the interface frequently changes shape (especially in 3D). Also, many numerical methods, such as finite differences, do not naturally apply to unstructured meshes. These concerns motivated the development of "embedded" (or, "immersed") methods that approximate solutions to (1)–(3) on Cartesian grids or structured meshes that do not conform to the interface. Despite advances in this direction, embedded methods that retain higher order accuracy in $L^\infty$ typically introduce relatively difficult linear algebra problems and complex implementations that sometimes require significant effort to adapt to general applications.

With these concerns in mind, we introduce a second order virtual node method for approximating the elliptic interface problem (1)–(3) with irregular embedded Neumann and Dirichlet boundaries on a uniform Cartesian grid. We use a regular Cartesian grid because it simplifies the implementation, permits straightforward Lagrange multiplier spaces and achieves higher order accuracy in $L^\infty$. Our approach uses duplicated Cartesian bilinear elements along the interface to introduce additional "virtual" nodes that accurately account for the lack of regularity. The method is variational (to define the stencil on the interface cells in a symmetric way) and uses Lagrange multipliers to enforce embedded Dirichlet conditions and the jump conditions (2) weakly. In the general case, our choice of Lagrange multiplier space admits a symmetric positive definite discretization. For the special case of smooth $\beta$, we present a novel discontinuity removal technique to allow the use of the standard 5-point difference stencil everywhere in the domain. This is unlike the numerous FEM approaches that use similar virtual node representations on unstructured meshes [7–14], as although some finite difference methods possess the notable advantage of discontinuity removal [15], to our knowledge a technique that retains the original system matrix is largely unexplored in the FEM frameworks. In all cases, our method yields the standard 5-point difference stencil away from the boundaries and interfaces. Numerical experiments indicate second order accuracy in $L^\infty$.

The remainder of the paper proceeds as follows: we review existing embedded methods in Section 2; in Section 3 we detail the proposed numerical method by first introducing its application to embedded Neumann problems in Section 3.1, and then to embedded Dirichlet and embedded interface problems in Sections 3.2 and 3.3, respectively; Section 3.3.1 details the special case of continuous $\beta$; Section 4 describes a simple implementation and lastly, we demonstrate the accuracy of the method with numerical experiments in Section 5 and conclude with a short summary and discussion.

## 2. Existing methods

The Immersed Interfaced Method (IIM) is perhaps the most popular finite difference method for approximating (1)–(3) to second order accuracy. LeVeque and Li first proposed the IIM for approximating elliptic interface problems in [16] and the term now applies to a widely researched and extensively applied class of finite difference methods [17–23]. See [15] and the references therein for a complete exposition of the method and its numerous applications, and [24] for justification of the general IIM approach. Using generalized Taylor expansions, the original IIM adaptively modifies the stencil to obtain $\mathcal{O}(h)$ truncation error along the interface. For smooth $\beta$, this reduces to the standard 5-point finite difference stencil, but otherwise results in a non-symmetric discretization that follows from locally solving constrained optimization problems that enforce a discrete maximum principle [25]. The IIM also generally requires the evaluation of higher-order jump conditions and surface derivatives along the interface. This can lead to difficulty in implementation, especially in 3D [26,20,15,21]. The piecewise-polynomial interface method of [27] is a notable new approach to the IIM that does not require the derivation of additional jump conditions and accurately treats complex interfaces. The works of [28–31,15] describe other various attempts to improve the efficiency and reduce the complexity of the IIM.

Extrapolation based finite difference schemes such as [32–37] introduce fictitious points along coordinate axes and use the known jump conditions to determine their values. The Ghost Fluid Method (GFM) of [32] exemplifies such methods. For

two and three dimensional problems, the GFM neglects the tangential flux terms $[\beta \nabla u \cdot \tau]$ in determining the fictitious values, resulting in a symmetric positive definite but first order [38] method. In fact, the GFM approach of Liu et al. [32] motivated our approach. However, given the similarity of our final approach to another of Fedkiw's methods, the virtual node algorithm [11–13], we describe our method accordingly. Our variational approach at duplicated interface cells prevents the loss of accuracy inherent in ignoring the tangential jumps. Various other approaches attain higher order accuracy by accounting for the tangential flux in other ways, often sacrificing simplicity and symmetry of discretization in the process. For instance, the Coupling Interface Method (CIM) proposed in [33] extends the GFM to higher dimensions by using a second order extension at most grid points, but reverting to a first order method at grid points where the second order extension cannot apply. The method couples jump conditions in different directions to express the tangential derivatives, and the use of one-sided differences results in a non-symmetric discretization. Similarly, the Matched Interface and Boundary (MIB) method [34] uses higher order extrapolations of the solution matched with higher order one-sided discretizations of the jump conditions to determine the values at fictitious points. The MIB method accounts for non-zero $[\beta \nabla u \cdot \tau]$ by differentiating the given jump conditions using one-sided interpolations. This widens the stencil in several directions that depend on the local geometry, and results in a non-symmetric discretization. The work of [39] extended the MIB to handle high curvature geometry, and [40] provides a 3D version. In [41] Hou and Liu also use techniques seemingly inspired by the analysis of the original GFM approach done in [38]. They develop a second order variational GFM by altering finite element interpolating functions to capture the jump conditions in the solution. Their approach is remarkably robust to non-smooth interface geometry, but results in a non-symmetric discretization in the general case. The recent works of [42,43] treated the cases of Robin and Neumann boundary conditions by altering the 5-point stencil along the boundary using a finite volume like approach. This results in an symmetric positive definite discretization.

Ideas similar to the extrapolation based finite difference schemes have also seen extensive use in FEM, for instance in the fictitious domain methods for embedded boundary problems [44–46,10,47–50] or the 'extended finite element methods' (XFEM) [51–58].[1] Fictitious domain methods handle embedded boundaries by including every element that intersects the interface into the discretization. This naturally introduces "virtual nodes" (or "ghost nodes") into the resulting discretization. The XFEM "enriches" the standard finite element basis with additional discontinuous basis functions, thereby introducing new degrees of freedom. These basis functions exist only at the nodes of elements that intersect the interface, and usually are the standard basis elements multiplied by a generalized Heaviside function. The methods of [7–10,13,14] introduce a related virtual node concept to provide the additional degrees of freedom required to represent the discontinuities. The most straightforward implementation of this virtual node concept [8–10] yields a representation equivalent to the standard Heaviside enrichment of the XFEM. However, this approach generalizes to the slightly richer representations of [11,12,14] that attain more geometric detail, particularly when dealing with coarse grids and non-smooth interfaces. Moreover, virtual node representations are considered more geometrically intuitive and easier to incorporate into existing FEM code [10,9,14] than traditional Heaviside enrichment.

The solution spaces of these FEM approaches generally do not satisfy the embedded boundary or interface conditions. Thus, these methods impose linear constraints with either penalty methods or Lagrange multipliers to enforce the conditions in some weak sense. For example, see [44–46,10] and the references therein. When using Lagrange multipliers, the Ladyzhenskaya–Babuška–Brezzi inf–sup conditions place stringent limitations on the types of constraints that will retain optimal convergence rates of the approximation spaces [60,61,49,62,57,46]. Such inf–sup restrictions generally limit the strength of the Lagrange multiplier space relative to the solution approximation space. For certain elements, designing the proper approximation spaces is a non-trivial task [57,55]. Moreover, the use of Lagrange multipliers requires the solution of an indefinite saddle-point system that can potentially introduce significant cost. Applying stabilization through a consistent penalty method, such as Nitsche's method, presents an alternative approach [10,50,46,8]. However, these can have adverse effects on conditioning and require the determination of the stabilization parameters. Instead of using Lagrange multipliers or stabilization, the methods of [63–66,41] alter the basis functions to either satisfy the constraints directly, or simplify the process of doing so. In this regard, such methods represent the finite element analogues of the IIM, especially [41,65,66].

The method of [67] offers a finite volume approach to embedded boundary problems. Like some fictitious domain methods, XFEM and our virtual node method, this method uses partially empty cells along the boundary. However, the one-sided quadratic interpolations used to compute the fluxes along the boundary yield a non-symmetric system. See [68] for a more recent 3D version applied to Poisson's equation and the heat equation. In [69], Oevermann and Klein proposed a second order finite volume method for interface problems, and simplified and extended their method to 3D in [70]. In an approach similar to ours, any Cartesian cell that intersects the interface yields a distinct bilinear (or trilinear) representation of the solution. The jump conditions are then built into the difference stencil by locally solving constrained overdetermined systems. An asymptotic technique resolves the problem of vanishing cell volumes, though it requires specific treatment for each possible cell geometry. The resulting system is non-symmetric for the general case of $[\beta] \neq 0$.

When $[\beta] \neq 0$ the majority of these second order methods do not retain a symmetric positive definite stencil. While the FEM approaches that use stabilization do retain a symmetric positive definite discretization [10], generally the FEM that use Lagrange multipliers, such as [53], result in a symmetric indefinite discretization. Although we use Lagrange multipliers, we

[1] See [59] for corrections to IIM convergence estimates.

present a simple method of reducing the indefinite system to a symmetric positive definite system using a null-space method. On the other hand, when the coefficient $\beta$ is smooth across the interface, methods such as the original IIM achieve second order accuracy by only altering the right hand side of the system. For this case, we present a method that uses the virtual node framework that also retains the original left hand side.

## 3. Description of numerical method

Our method naturally handles both interfacial discontinuities and irregular domains embedded in a Cartesian grid. In fact, a slight modification of our approach to embedded boundary conditions yields our method for interfacial discontinuities. Furthermore, our treatment of embedded Dirichlet boundary conditions is just a slight modification of our treatment of embedded Neumann boundary conditions. Therefore, we first present our method for embedded Neumann boundary conditions followed by our method for Dirichlet boundary conditions and then finally present our approach to interfacial discontinuities.

### 3.1. Embedded Neumann

Our approach to solving embedded Neumann problems is very similar to that proposed by Almgren et al. in [47], as well as some XFEM approaches, e.g. [53]. The recent methods proposed in [42,43] are comparable in accuracy to our method and are straightforward to implement.

Similar to [47], we discretize the embedded Neumann problem,

$$-\nabla\cdot(\beta(\mathbf{x})\nabla u(\mathbf{x})) = f(\mathbf{x}), \quad \mathbf{x}\in\Omega, \tag{6}$$

$$\beta(\mathbf{x})\nabla u\cdot n = q(\mathbf{x}) \quad \mathbf{x}\in\partial\Omega, \tag{7}$$

over a regular Cartesian grid (one that does not have to conform to $\partial\Omega$) using the energy minimization form of (6) and (7):

$$\text{over all } u\in\mathbf{H}^1(\Omega), \quad \text{minimize}$$

$$E(u) = e(u) - (f,u)_\Omega - (q,u)_{\partial\Omega} = \int_\Omega \frac{1}{2}\nabla u\cdot\beta\nabla u\,dx - \int_\Omega fu\,dx - \int_{\partial\Omega} qu\,dS. \tag{8}$$

We begin by embedding the domain $\Omega$ in a regular Cartesian grid $\mathcal{G}^h$ with grid-spacing $\Delta y = \Delta x = h$. We include all Cartesian cells (or elements) $c_k$ that intersect $\Omega$ in the discretization, and refer to this set $\mathcal{C}^h = \{c_k \cap \Omega \neq \emptyset\} \subset \mathcal{G}^h$ as the "computational domain" (see Fig. 2). Also, we define the set of all cells that intersect the boundary as $\mathcal{C}^h_{\partial\Omega} = \{c_k \cap \partial\Omega \neq \emptyset\} \subset \mathcal{C}^h$. We define the solution space $\mathbf{V}^h \subset \mathbf{H}^1(\Omega)$ as the space of continuous functions that are bilinear over each cell $c_k \in \mathcal{C}^h$. This approximation includes some partially empty cells that intersect the boundary and introduces "virtual" grid nodes (and virtual degrees of freedom) that lie outside of the domain. See Fig. 3 for a diagram labeling the degrees of freedom along a typical boundary. We refer to the portion of the cell that lies in the domain $\Omega$ as the "material" region, and use the term "material" nodes to describe grid nodes lying inside $\Omega$. For $u^h \in \mathbf{V}^h$, we write $u^h(\mathbf{x}) = \sum_{i=1}^n u_i N_i(\mathbf{x})$ for $\vec{u} = (u_1,\ldots,u_n) \in \mathbb{R}^n$ where $N_i(\mathbf{x})$ are the standard piecewise bilinear interpolation basis functions associated with the grid nodes. Here, $n$ denotes the number of degrees of freedom in the discretization and corresponds to the number of grid nodes that compose the cells of $\mathcal{C}^h$.

Using the virtual node representation, we define a discrete energy $E^h(u^h)$ over $u^h \in \mathbf{V}^h$. Although we could discretize the energy directly with the piecewise bilinear representation, this would result in a second order 9-point stencil away from the interface (as in [47]). To retain the standard 5-point difference stencil away from the boundary we use different definitions of the energy over $\mathcal{C}^h \setminus \mathcal{C}^h_{\partial\Omega}$ and $\mathcal{C}^h_{\partial\Omega}$,

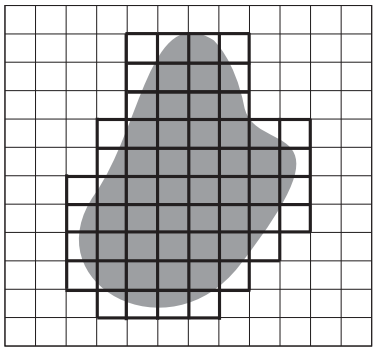


**Fig. 2.** Embedding $\Omega$ in a Cartesian grid. The computational domain consists of all cells $c_k \in \mathcal{G}^h$ that intersect $\Omega$. Such cells are outlined in bold. This procedure introduces virtual degrees of freedom into the discretization, namely those nodes in the bold grid that do not lie in the shaded domain $\Omega$ itself.

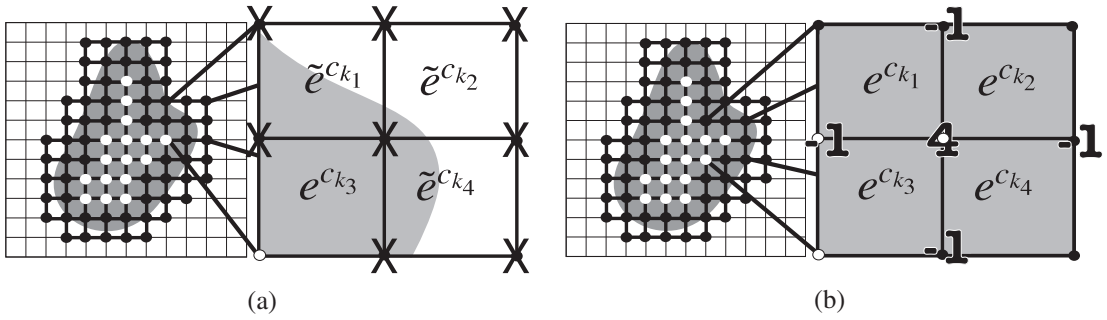



**Fig. 3.** Illustration of the interior and boundary stencils. The black degrees of freedom have a modified stencil; the stencil is unaltered at the white degrees of freedom. On the left, the nodes marked with an **X** contribute a non-zero entry to the stencil for the center node via the cell-wise energies. The right figure depicts the 5-point stencil for the center node that results from the definition of $e^{c_{k_i}}$.

$$E^h(u^h) = \sum_{c_k \in \mathcal{C}^h \setminus \mathcal{C}^h_{\partial\Omega}} e^{c_k}(u^h) - (f, u^h)^{c_k}_{\Omega} + \sum_{c_k \in \mathcal{C}^h_{\partial\Omega}} \tilde{e}^{c_k}(u^h) - (f, u^h)^{c_k}_{\Omega} - (q, u^h)^{c_k}_{\partial\Omega}, \tag{9}$$

where the superscripts denote restriction to cell $c_k$. Over cells $c_k \in \mathcal{C}^h \setminus \mathcal{C}^h_{\partial\Omega}$ that do not intersect the boundary, we define $e^{c_k}(u^h)$ as

$$e^{c_k}(u^h) = \frac{\overline{\beta} h^2}{4} \left\{ \left( \frac{u_{i+1,j} - u_{ij}}{h} \right)^2 + \left( \frac{u_{ij+1} - u_{ij}}{h} \right)^2 + \left( \frac{u_{i+1,j+1} - u_{i+1,j}}{h} \right)^2 + \left( \frac{u_{i+1,j+1} - u_{ij+1}}{h} \right)^2 \right\}. \tag{10}$$

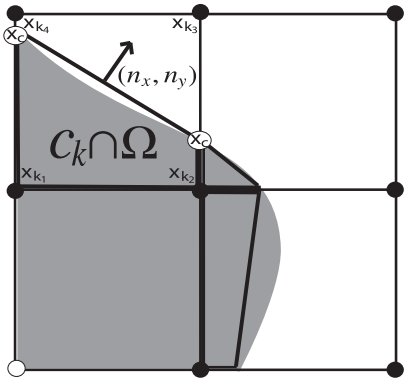
Here $\bar{\beta}$ denotes the cell average, and $\{u_{p,q}\}$ denote the degrees of freedom at the four corners of the cell. If a cell $c_k \in \mathcal{C}^h_{\partial\Omega}$, i.e. the cell intersects the boundary, then we use the Cartesian bilinear representation to define $\tilde{e}^{c_k}(u^h)$. If we let $\{N_{p,q}\}$ denote the bilinear basis functions associated with the four corners of the cell, this yields the discretization

$$\tilde{e}^{c_k}(u^h) = \frac{1}{2} \sum_{r,s,r',s' \in \{0,1\}} u_{i+r,j+s} u_{i+r',j+s'} \int_{c_k \cap \Omega} \bar{\beta} \nabla N_{i+r,j+s} \cdot \nabla N_{i+r',j+s'} \, dx. \tag{11}$$

We evaluate the integrals analytically using the divergence theorem based on a polygonal representation of $\partial\Omega$ as in Almgren et al. [47]. See Fig. 4 and Section 4 where the procedure is described in more detail. The tilde denotes the different discretizations of the energy over cells that intersect the boundary. Notice we evaluate each integral only over the portion of the cell that lies within the domain. Similarly, as in [69,70] we define the cell average $\overline{\beta}$ as the average only over $c_k \cap \Omega$. We discretize the other forms cell-wise as

$$(f, u^h)^{c_k}_{\Omega} = \sum_{r,s \in \{0,1\}} u_{i+r,j+s} \int_{c_k \cap \Omega} \bar{f} N_{i+r,j+s} \, dx, \tag{12}$$

$$(q, u^h)^{c_k}_{\partial\Omega} = \sum_{r,s \in \{0,1\}} u_{i+r,j+s} \int_{c_k \cap \partial\Omega} \bar{q} N_{i+r,j+s} \, dS. \tag{13}$$




$$\int_{c_k \cap \Omega} \overline{\beta} \nabla N_i \cdot \nabla N_j \, d\mathbf{x} =$$

$$\overline{\beta} \int_{c_k \cap \Omega} (\partial_y N_i)(\partial_y N_j) + (\partial_x N_i)(\partial_x N_j) \, d\mathbf{x} =$$

$$\overline{\beta} \int_{c_k \cap \Omega} p_2(x) + q_2(y) \, dxdy =$$

$$\overline{\beta} \int_{c_k \cap \Omega} \nabla \cdot (p_3(x), q_3(y)) \, dxdy =$$

$$\overline{\beta} \int_{\partial(c_k \cap \Omega)} (p_3(x), q_3(y)) \cdot (n_x, n_y) \, dS$$

**Fig. 4.** Polygonal representation of $\partial\Omega$. We compute the modified stencil analytically by using the divergence theorem on the material region $\Omega \cap c_k$ in each cell. Here, $p_n(x)$ and $q_n(y)$ denote appropriate polynomials of order $n$ in a single variable. Notice the relatively small area of the material region in the top, right cell. As this area approaches zero, the virtual node at the top, right of this cell introduces ill-conditioning into the stiffness matrix.

Here $\bar{f}$ is the average source over $c_k \cap \Omega$ and $\bar{q}$ is the average normal flux over $c_k \cap \partial\,\Omega$. Again, we evaluate the integrals analytically, applying the divergence theorem where necessary. We minimize the discrete energy (9) by solving the linear system

$$A\vec{u} = \vec{f}, \tag{14}$$

$$A_{ij} = \frac{\partial^2}{\partial u_i \partial u_j} E^h(u^h), \tag{15}$$

$$f_i = \frac{\partial}{\partial u_i}\left((f, u^h)_\Omega + (q, u^h)_{\partial\Omega}\right) \tag{16}$$

for the vector $\vec{u}$. We use the standard FEM term "stiffness matrix" to refer to the matrix $A$, and it is clear from the derivation that $A$ is symmetric and positive semi-definite. With this approach, our definition of the energy (11) results in a slightly denser stencil near the boundary, as all four degrees of freedom in a cell couple together if $\partial\Omega$ passes through that cell. See Fig. 3 for a graphical depiction of the stencil definitions and the sparsity pattern of the stiffness matrix. In Section 4, the practical construction of $A$ and $\vec{f}$ is given in more detail.

We should note that the conditioning of the stiffness matrix may deteriorate when cells have very small material regions. This arises from the increasing irrelevance of virtual degrees of freedom (see the upper right node in Fig. 4). The respective row and column in $A$ and the corresponding entry in $\vec{f}$ all approach zero simultaneously, however simple Jacobi preconditioning eliminated serious conditioning issues in our numerical experiments.

### 3.2. Embedded Dirichlet

In this section, we detail how a slight modification of our embedded Neumann approach allows us to solve embedded Dirichlet problems

$$-\nabla\cdot(\beta(\mathbf{x})\nabla u(\mathbf{x})) = f(\mathbf{x}), \quad \mathbf{x}\in\Omega, \tag{17}$$

$$u = p(\mathbf{x}) \quad \mathbf{x}\in\partial\Omega, \tag{18}$$

within our virtual node framework. Although alternatives that are easier to implement exist for this particular problem, for instance [35], a straightforward combination of our embedded Neumann and embedded Dirichlet approaches yields our method for embedded interface problems. This results in a method that encapsulates all types of boundary conditions in a unified framework.

For the embedded Dirichlet case, we use the constrained minimization problem:

$$\begin{aligned} &\text{over all } u\in\mathbf{H}^1(\Omega), \quad \text{minimize} \\ &E(u) = e(u) - (f,u)_\Omega \quad \text{such that} \end{aligned} \tag{19}$$

$$(u,\mu)_{\partial\Omega} = (p,\mu)_{\partial\Omega} \quad \forall\mu\in\mathbf{H}^{-1/2}(\partial\Omega). \tag{20}$$

We discretize the energy (19) exactly as in the Neumann case, so the only difference comes in discretizing the constraints (20). We proceed by selecting a finite dimensional subspace $\mathbf{\Lambda}^h \subset \mathbf{H}^{-1/2}(\partial\Omega)$, and enforce (20) for all $\mu^h \in \mathbf{\Lambda}^h$. Not all plausible choices will yield an acceptably accurate approximation, as in general $(\mathbf{\Lambda}_h, \mathbf{V}_h)$ must satisfy an inf–sup stability criterion to retain the optimal convergence rates of the approximation spaces [61]. One suitable choice for $\mathbf{\Lambda}^h$, used for instance by the XFEM [58], defines $\mu^h$ as piecewise constant over the intersection of $\partial\Omega$ with each Cartesian cell (see Fig. 5). In other words, we define $\mu^h \in \mathbf{\Lambda}^h$ as

$$\mu^h(\mathbf{x}) = \sum_{c_i\in\mathcal{C}^h_{\partial\Omega}} \mu_i \chi_{c_i\cap\partial\Omega}(\mathbf{x}),$$

where the sum ranges over all Cartesian cells $c_i$ that intersect the boundary $\left(c_i \in \mathcal{C}^h_{\partial\Omega}\right)$ and the characteristic functions $\chi_{c_i\cap\partial\Omega}$ are given by

$$\chi_{c_i\cap\partial\Omega}(\mathbf{x}) = \begin{cases} 1, & \mathbf{x}\in c_i\cap\partial\Omega, \\ 0, & \mathbf{x}\notin c_i\cap\partial\Omega. \end{cases}$$

With this choice of $\mathbf{\Lambda}^h$, satisfying (20) for all $\mu^h$ yields a set of sparse linear constraints $B\vec{u} = \vec{p}$ on the coefficient vector of the approximate solution $u^h$. Each row of the matrix $B$ corresponds to a Cartesian cell $c_i \in \mathcal{C}^h_{\partial\Omega}$ (see Fig. 5), and enforces the condition

$$\int_{c_i\cap\partial\Omega} u^h(\mathbf{x})dS = \int_{c_i\cap\partial\Omega} p(\mathbf{x})dS.$$

Therefore, if $\mathcal{C}^h_{\partial\Omega} = \{c_1,\ldots,c_m\}$ and $\vec{u}\in\mathbb{R}^n$, then $B\in\mathbb{R}^{m\times n}$ and

$$B_{ij} = \int_{c_i\cap\partial\Omega} N_j(\mathbf{x})dS$$

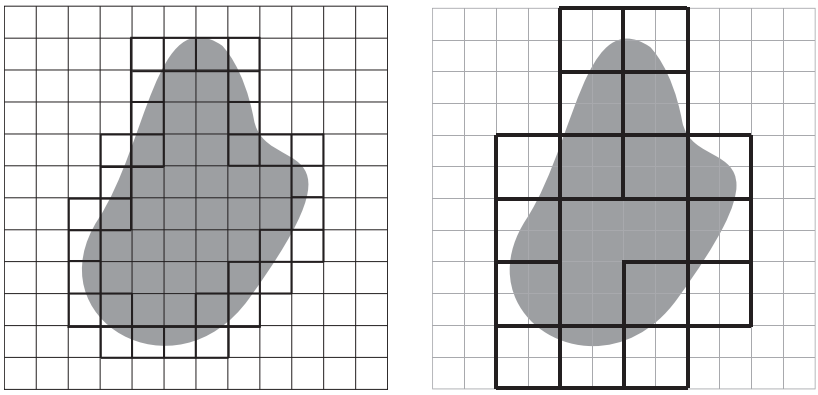



**Fig. 5.** Lagrange multiplier spaces. On the left: functions in $\mathbf{\Lambda}^h$ are piecewise constant over the intersection of the boldly outlined cells $c_i \in \mathcal{C}^h_{\partial\Omega}$ with the boundary $\partial\Omega$. On the right: functions in $\mathbf{\Lambda}^{2h}$ are piecewise constant over the intersection of the coarser bold cells $\hat{c}_i \in \mathcal{C}^{2h}_{\partial\Omega}$ with the boundary $\partial\Omega$. In the image on the right, the bold black lines mark the cells $\hat{c}_i \in \mathcal{G}^{2h}$.

for each Cartesian bilinear basis function $N_j(\mathbf{x})$. The corresponding entry in $\vec{p}$ is

$$p_i = \int_{c_i \cap \partial\Omega} p(\mathbf{x}) dS.$$

Again, we compute these integrals analytically. Discretizing (19) and (20) thus gives rise to the quadratic program:

$$\begin{aligned} &\text{minimize over } \vec{u} \in \mathbb{R}^n \\ &E^h(u^h) = e(u^h) - (f, u^h)_\Omega = \frac{1}{2}\vec{u}^t A\vec{u} - \vec{f}^t\vec{u} \\ &\text{subject to } B\vec{u} = \vec{p} \end{aligned} \tag{21}$$

The matrix $A$ and the vector $\vec{f}$ carry over exactly from the embedded Neumann case described in Section 3.1.

Unfortunately, solving this problem efficiently can require some care. While many approaches exist for solving minimization problems of the form (21) or the equivalent saddle-point system

$$\begin{pmatrix} A & B^t \\ B & 0 \end{pmatrix} \begin{pmatrix} \vec{u} \\ \vec{\lambda} \end{pmatrix} = \begin{pmatrix} \vec{f} \\ \vec{p} \end{pmatrix},$$

we use a null-space method to retain a symmetric positive definite discretization. See [71] for a survey of alternative approaches. For any matrix $Z$ whose columns span the null-space of $B$, and any vector $\vec{c}$ satisfying $B\vec{c} = \vec{p}$,

$$\vec{u} = \vec{c} + Z(Z^tAZ)^{-1}Z^t(\vec{f} - A\vec{c}) \tag{22}$$

uniquely solves (21). Therefore, given a null-basis $Z$ and a particular solution $\vec{c} \in \mathbb{R}^n$ satisfying $B\vec{c} = \vec{p}$, we solve the quadratic program (21) by solving the symmetric positive definite system $Z^tAZ\vec{v} = Z^t(\vec{f} - A\vec{c})$. The null-space of $A$ is spanned by the vector $(1, 1, \ldots, 1)^t \in \mathbb{R}^n$ and the entries of $B$ are all non-negative so $\ker(A) \cap \ker(B) = \{\vec{0}\}$. Therefore, $Z^tAZ > 0$ so we can use straightforward methods such as Conjugate Gradient to solve the symmetric positive definite linear algebra problem. However, obtaining $Z$ through computational methods such as QR factorization or the SVD can prove costly, and moreover produce dense representations of $Z$.

A *fundamental basis* presents an alternative to numerical factorization [71]. The matrix $B$ is full rank if and only if an ordering of the degrees of freedom exists so that $B = (B_m|B_{n-m})$ for some $m \times m$ non-singular matrix $B_m$. Any such ordering gives the corresponding fundamental basis

$$Z = \begin{pmatrix} -B_m^{-1}B_{n-m} \\ I_{n-m} \end{pmatrix}. \tag{23}$$

Clearly, $BZ = 0$ and $\vec{c} = \begin{pmatrix} B_m^{-1}\vec{p} \\ 0 \end{pmatrix}$ satisfies $B\vec{c} = \vec{p}$. Therefore, if we can solve systems of the form

$$B_m\vec{x} = \vec{d}, \tag{24}$$

efficiently, we can store the factors $B_m$, $B_{n-m}$, $A$ sparsely and compute the action of $Z^tAZ$ readily (e.g. for use in Conjugate Gradient). Note that, regardless of the choice of $B_m$, the symmetric positive definite stencil defined by $Z^tAZ$ coincides with the standard 5-point stencil for all degrees of freedom sufficiently far from the interface.

We now show that the rows and columns of the matrix $B$ can be re-ordered to produce a non-singular, upper triangular matrix $B_m$. Specifically, ordering the cut-cells $\{c_1, \ldots, c_m\} = \mathcal{C}^h_{\partial\Omega}$ lexicographically, and then selecting the lower-left node of the $i$th cut-cell as the $i$th degree of freedom (thus re-ordering the rows in $A$, $B$ and $\vec{u}$), gives $B = (B_m | B_{n-m})$ with $B_m$ upper triangular and non-singular (see Fig. 6). Unfortunately, despite the convenient triangular structure of $B_m$, prohibitively large numerical error persists when solving (24), even on relatively coarse grids. As the interface in a given cell recedes from the lower-left node of that cell (for instance, cells and nodes 1, 4, 5, 9, 11, 14, 17, 18, 19, 21, 22, 23, or 29 in Fig. 6), the corresponding row in $B_m$ has off-diagonal entries with substantially larger magnitude than the diagonal entry of that row. Generally, enough rows of this type exist so that $B_m$ behaves much like the matrix

$$C = \begin{pmatrix} 1 & 2 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 2 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & 1 & 2 & 0 \\ 0 & \cdots & \cdots & 0 & 1 & 2 \\ 0 & 0 & \cdots & \cdots & 0 & 1 \end{pmatrix}.$$

Considering the linear system $Cx = \vec{e_m}$ with $\vec{e_m} = (0, \ldots, 0, 1)^t$ illuminates the source of this error. Indeed, in this case $\|C^{-1}\vec{e_m}\|_\infty$ grows like $2^m$. The forward substitutions with $B^t_m$, needed for $Z^t$ multiplies, also exhibit this behavior. As $m$ increases under grid refinement, these behaviors quickly (in some cases, anything finer than a $64 \times 64$ grid) dominate machine precision. In practice, scalar multiples of $\vec{e_m}$ always appear in $B_{n-m}$, making such a $B_m$ practically unusable in a null-space method. Moreover, this problem persists in all similar constructions of $B_m$ (different orderings, node choices, etc.).

For this reason, we use an alternative approximation to $\mathbf{H}^{-1/2}(\partial\Omega)$ that produces a different set of linear constraints. Our choice permits an ordering of $B$ with a non-singular, diagonal sub-matrix $B_m$. If we enforce one constraint per cell as above, then in general there do not exist $m$ degrees of freedom that each only participate in one constraint, so that no ordering could produce a diagonal matrix $B_m$. Motivated by this observation, we approximate $\mathbf{H}^{-1/2}(\partial\Omega)$ using $\mathbf{\Lambda}^{2h}$, the space of Lagrange multipliers corresponding to the grid $\mathcal{G}^{2h}$. That is, for every $\mu^h \in \mathbf{\Lambda}^{2h}$,

$$\mu^h(\mathbf{x}) = \sum_{\hat{c}_k \in \mathcal{C}^{2h}_{\partial\Omega}} \mu_k \chi_{\hat{c}_k \cap \partial\Omega}(\mathbf{x}).$$

Thus, each row of $B$ now enforces the condition

$$\int_{\hat{c}_k \cap \partial\Omega} u^h(\mathbf{x}) dS = \int_{\hat{c}_k \cap \partial\Omega} p(\mathbf{x}) dS,$$

for each of the cells $\hat{c}_k \in \mathcal{C}^{2h}_{\partial\Omega} = \{\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_m\}$ (see Fig. 5) that intersect the boundary. In the figure, each of the cells in the grid $\mathcal{G}^{2h}$ is the union of four cells in the grid $\mathcal{G}^h$, so that at the center of each cell $\hat{c}_k \in \mathcal{G}^{2h}$ lies a degree of freedom $u_k$ whose associated nodal basis function $N_k$ vanishes outside the cell $\hat{c}_k$. Therefore for each cell $\hat{c}_k \in \mathcal{C}^{2h}_{\partial\Omega}$ we choose this central degree of freedom as the $k$th in our re-ordering. As such,

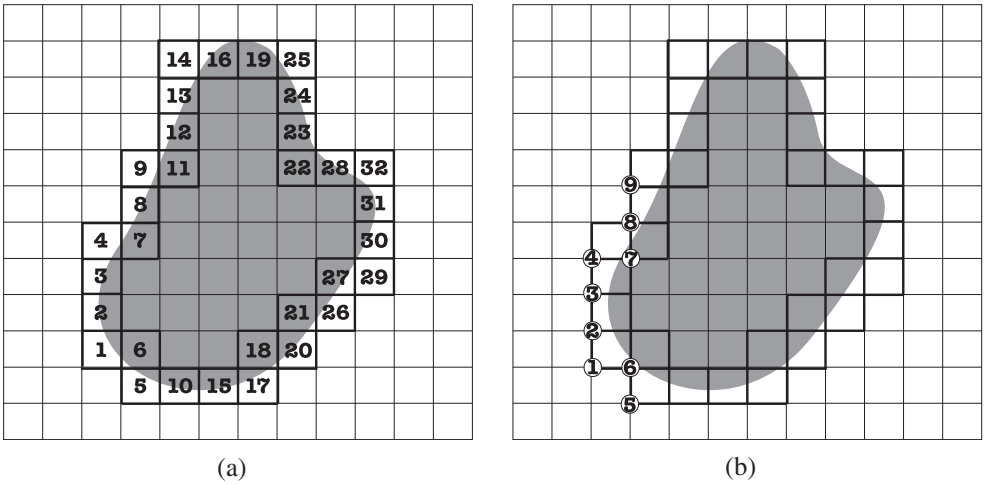



**Fig. 6.** Upper triangular ordering. The cell-centered numbers in figure (a) indicate the ordering of the cells $c_k \in \mathcal{C}^h_{\partial\Omega}$. The nodal numbers in figure (b) indicate the corresponding ordering for the first nine degrees of freedom.

$$\int_{\hat{c}_i \cap \partial\Omega} N_k \, dS = 0, \quad \forall i \neq k, \;\; 1 \leqslant i \leqslant m. \tag{25}$$

See Fig. 7 for a pictorial description of this ordering. Clearly, this gives $B = (B_m | B_{n-m})$ with $B_m$ diagonal and non-singular. We then use the corresponding fundamental basis (23) to trivially reduce the saddle-point problem to the symmetric positive definite system $Z^t A Z \vec{v} = Z^t(\vec{f} - A\vec{c})$ by applying the null-space method (22).

### 3.3. Embedded interface

To handle the full elliptic interface problem (1)–(3), we combine our embedded Neumann and embedded Dirichlet approaches in a straightforward way. We consider the equivalent minimization form of the problem (1)–(3):

$$\text{over all } u \in \mathbf{V} = \{u : u^{\pm} \in \mathbf{H}^1(\Omega^{\pm})\}, \quad \text{minimize}$$

$$E(u) = e(u) - (f,u)_{\Omega} - (b,\bar{u})_{\Gamma} = \int_{\Omega^+ \cup \Omega^-} \frac{1}{2} \nabla u \cdot \beta \nabla u \, dx - \int_{\Omega} f u \, dx - \int_{\Gamma} b \bar{u} \, dS \tag{26}$$

$$\text{such that } ([u],\mu)_{\Gamma} = (a,\mu)_{\Gamma} \quad \forall \mu \in \mathbf{H}^{-1/2}(\Gamma). \tag{27}$$

Here $\bar{u}(\mathbf{x})|_{\Gamma} = (u^+ + u^-)/2$. As before, we define discretizations of $\mathbf{V}$ and $\mathbf{H}^{-1/2}(\Gamma)$ and then solve the resulting discrete saddle-point problem. To define $\mathbf{V}^h \subset \mathbf{V}$, we separately discretize $\mathbf{H}^1(\Omega^+)$ and $\mathbf{H}^1(\Omega^-)$ using the same virtual node representation used to discretize the embedded Neumann problem. This will naturally introduce duplicate Cartesian cells that intersect the boundary, with independent copies associated with the interior and exterior discretizations (see Fig. 8). This discretization results in the block diagonal stiffness matrix for the interface problem,

$$A = \begin{pmatrix} A^+ & 0 \\ 0 & A^- \end{pmatrix},$$

where $A^+$ is the stiffness matrix associated with the embedded Neumann problem on $\Omega^+$ and $A^-$ is the stiffness matrix associated with the embedded Neumann problem on $\Omega^-$, as described in Section 3.1.

Similarly, along the interface we make the same choice of discrete Lagrange multiplier space as before, so that over every cell $\hat{c}_k \in \mathcal{C}_{\Gamma}^{2h}$,

$$\int_{\hat{c}_k \cap \Gamma} [u^h] dS = \int_{\hat{c}_k \cap \Gamma} a \, dS.$$

This results in the block interface constraint matrix $B = (B^+ | -B^-)$, where $B^{\pm}$ is respectively the constraint matrix associated with the embedded Dirichlet problem on the exterior or interior of the interface. In other words, $B_{ij} = \int_{\hat{c}_i \cap \Gamma} \text{sign}(j) N_j(\mathbf{x}) dS$, where $\text{sign}(j) = 1$ if degree of freedom $j$ is associated with $u^{+,h}$ and $\text{sign}(j) = -1$ if degree of freedom $j$ is associated with $u^{-,h}$. These discretization choices give the saddle-point problem

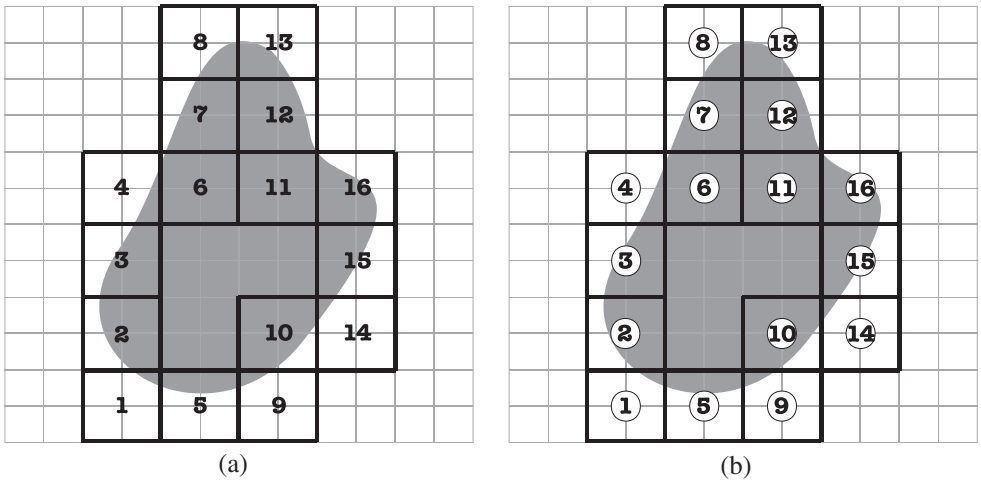



**Fig. 7.** Diagonal ordering scheme. We enforce one constraint per coarse cell, enumerated in the image on the left. In the image on the right, we index the degrees of freedom at the centers of the coarse cells by the corresponding constraint indices. This gives a constraint matrix $B$ with a diagonal sub-matrix $B_m$. Note that this gives a slightly denser $B_{n-m}$, since now as many as nine degrees of freedom may contribute to a given row for embedded Dirichlet problems.

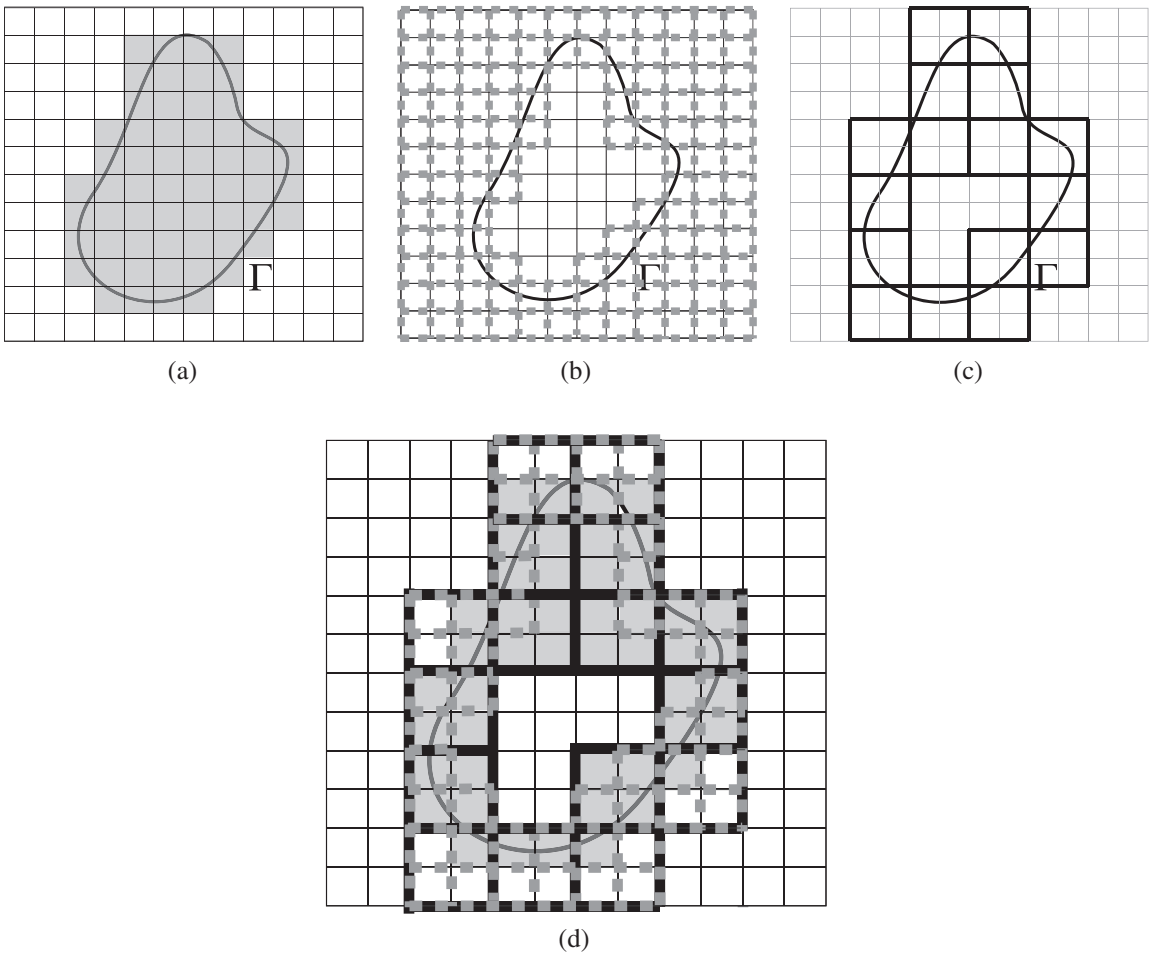



**Fig. 8.** In (a), the degrees of freedom lying on shaded cells define $u^{-,h}$, and in (b), the degrees of freedom lying on dashed cells define $u^{+,h}$. Applying our embedded Neumann approach on the shaded and dashed grids defines the matrices $A^-$ and $A^+$, respectively. Degrees of freedom associated with $u^{-,h}$ and $u^{+,h}$ are collocated along the interface. These representations couple together in the coarse cells outlined in figure (c). Figure (d) depicts the overlapping domains of definition of $u^{-,h}$ and $u^{+,h}$ in the coarse cells.

$$\begin{pmatrix} A^+ & 0 & B^{+t} \\ 0 & A^- & -B^{-t} \\ B^+ & -B^- & 0 \end{pmatrix} \begin{pmatrix} \vec{u^+} \\ \vec{u^-} \\ \vec{\lambda} \end{pmatrix} = \begin{pmatrix} \vec{f^+} \\ \vec{f^-} \\ \vec{a} \end{pmatrix}, \tag{28}$$

where $\vec{u}^+$ contains the degrees of freedom associated with the nodal values of the exterior discretization and $\vec{u}^-$ contains the degrees of freedom associated with the nodal values of the interior discretization. We once again solve the saddle-point system using the null-space method described above in Section 3.2 by defining an ordering $B = (B_m|B_{n-m})$ with $B_m$ diagonal. Given any ordering for the constraints, we choose the virtual degree of freedom at the center of the $i$th cell $\hat{c}_i \in \mathcal{C}_\Gamma^{2h}$ as the $i$th degree of freedom in our ordering. See Figs. 7 and 8 and Section 3.2 for more information. There are always at least two degrees of freedom associated with the center node. We choose the virtual degree of freedom as this results in a system $Z^tAZ$ with significantly better conditioning in practice.

*3.3.1. Virtual node discontinuity removal*

In general, our proposed method requires the solution of the symmetric positive definite system $Z^tAZ$. However, if the coefficient $\beta$ is smooth, the IIM and similar methods achieve uniform second order accuracy without altering the original 5-point difference stencil. In this section, we demonstrate how the virtual node framework similarly allows the use of the 5-point difference stencil for continuous coefficients. For simplicity of exposition, we assume $\beta(\mathbf{x}) \equiv 1$ for the rest of this section.

Suppose $c(\mathbf{x}) \in \mathbf{V}$ is constructed to satisfy the jump conditions (2) and (3) and $u(\mathbf{x})$ is the exact solution. Then as $[\beta] = 0$, the difference $w(\mathbf{x})$: $= u(\mathbf{x}) - c(\mathbf{x})$ satisfies $[\beta \nabla w \cdot n] = \beta[\nabla w \cdot n] = 0$ and $[w] = 0$. Since $w$ satisfies homogeneous jump conditions

$[\nabla w \cdot n] = 0$ and $[w] = 0$, we do not require virtual nodes to capture any discontinuities across $\Gamma$. In this manner, solving for $w$ presents an appealing alternative as the presence of virtual nodes no longer adversely affects the subsequent linear algebra problem. Therefore, when $[\beta] = 0$ we recover an approximation to (2) and (3) by separately discretizing $w$ and $c$, then setting $u = w + c$.

We discretize $w$ over the unduplicated grid $\mathcal{G}^h$ using $\mathbf{H}^1(\Omega)$ Cartesian piecewise bilinear elements. Consequently, if the grid $\mathcal{G}^h$ contains $r$ material degrees of freedom, then $\vec{w} \in \mathbb{R}^r$ contains the coefficients in terms of the bilinear basis. We discretize $u$ and $c$ using the full virtual node basis $\mathbf{V}^h$ as they possesses lower regularity across $\Gamma$. With these choices, we can represent the coefficient vector $\vec{u} \in \mathbb{R}^n$ $(n > r)$ of the approximate solution $u^h$ in the basis of $\mathbf{V}^h$ as $\vec{u} = \vec{c} + T\vec{w}$, where the matrix $T \in \mathbb{R}^{n\times r}$ maps from the bilinear basis to the virtual node basis. We determine this change of basis by a simple identification of virtual and material nodes, as a function $v^h \in \mathbf{V}^h$ satisfies homogeneous jump conditions if and only if the value of the function $v^h$ at a virtual node always equals its value at the associated material node. Therefore, $T$ maps the value at a given node in the original grid to every node, virtual or material, associated with the same location in the virtual node basis.

Although any ordering of degrees of freedom will suffice to construct $T$, for simplicity assume that

$$\vec{u} = (u_1, u_2, \ldots, u_{n_v}, u_{n_v+1}, u_{n_v+2}, \ldots, u_{2n_v}, u_{2n_v+1}, \ldots, u_n)^t.$$

Here, $\{u_k\}_{k=1}^{n_v}$ represent the $n_v := n - r$ total coefficients of the virtual degrees of freedom; $u_{n_v+k}$, $1 \leqslant k \leqslant n_v$, represents the coefficient of the real degree of freedom corresponding to the same physical node as $u_k$; the remaining $\{u_k\}_{k=2n_v+1}^{n}$ degrees of freedom do not lie on any cut-cells. Then

$$T = \begin{pmatrix} I_{n_v} & 0 \\ I_{n_v} & 0 \\ 0 & I_{n-2n_v} \end{pmatrix}. \tag{29}$$

More generally, each column of $T$ corresponds to a material node in the grid, and each row of $T$ corresponds to either a material node or a virtual node. Then the column of $T$ corresponding to a material node $\mathbf{x}_l$ simply has a one in the column corresponding to $\mathbf{x}_l$, a one in the column corresponding to any virtual node in the same physical location (i.e. coordinates) as $\mathbf{x}_l$, and zeros otherwise.

Determining $w^h$ now proceeds in a manner analogous to the null-space method used to solve (21): we wish to minimize the energy over all vectors of the form $\vec{u} = \vec{c} + T\vec{w}$. For the following discussion, suppose we define the discrete energy (26) using the Cartesian bilinear representation everywhere in the domain. Then substituting the expression for $\vec{u}$ into the energy (26) gives

$$E^h(\vec{u}) = \frac{1}{2}\vec{w}^t T^t A T \vec{w} - \vec{f}^t T\vec{w} + \vec{w}^t T^t A\vec{c} + \frac{1}{2}\vec{c}^t A\vec{c} - \vec{f}^t\vec{c}, \tag{30}$$

which defines an energy only over the original, material degrees of freedom $\vec{w} \in \mathbb{R}^r$. Differentiation with respect to $w_i$ then leads to the linear system

$$T^t A T\vec{w} = T^t(\vec{f} - A\vec{c}), \tag{31}$$

$$\vec{u} = \vec{c} + T\vec{w}. \tag{32}$$

Remarkably, the matrix $T^tAT$ is the straightforward discretization over the material degrees of freedom, i.e. a 9-point, second order approximation to the Laplacian. Moreover, as $\vec{w}$ corresponds to the material nodal values on a regular grid, we may operate on it instead with the standard 5-point difference stencil $\Delta^h$ and solve the system

$$\Delta^h\vec{w} = T^t(\vec{f} - A\vec{c}) \tag{33}$$

to provide an approximate solution at all of the relevant real degrees of freedom. This approach allows the application of efficient, black-box solvers for $\Delta^h$ and only requires constructing the right hand side of (33). Thus, the lack of regularity in the problem no longer adversely affects the linear algebra.

In principle, many different constructions could result in a satisfactory particular solution $c$. To minimize the computational effort, we construct a $c$ supported only along the interface. The time required to generate such a particular solution contributes negligibly to the overall computational cost. We assume that $\Omega^-$ does not intersect the computational boundary and construct a particular solution $c$ that vanishes on the exterior region. That is, $c|_{\Omega^+} = 0$ so that $c|_{\partial\Omega} = 0$, $[c] = -c^- = a$, $\beta[\nabla c \cdot n] = -\beta\nabla c^- \cdot n = b$. Therefore, we need only to define $c^-$ over those interior material and interior virtual nodes along the interface, and do so by straightforward extrapolation.

## 4. Implementation

In this section we detail a sample implementation of our method with the interface represented as a level set $\phi$, where $\Omega = \{\phi < 0\}$ for irregular domain problems and $\Omega^- = \{\phi < 0\}$ for embedded interface problems. We describe the implementation for the embedded Dirichlet case $\Omega = \{\phi < 0\}$, since the interface case is analogous.

First, we define the computational domain as those cells $c_k = \{\mathbf{x}_{k_1}, \mathbf{x}_{k_2}, \mathbf{x}_{k_3}, \mathbf{x}_{k_4}\}$ where $\phi(\mathbf{x}_{k_i}) < 0$ for at least one node $\mathbf{x}_{k_i}$. If $\phi(\mathbf{x}_{k_i}) < 0$ for all $1 \leqslant i \leqslant 4$ then $c_k$ lies in $\mathcal{C}^h \setminus \mathcal{C}^h_{\partial\Omega}$. Otherwise, the cell $c_k$ lies in $\mathcal{C}^h_{\partial\Omega}$. That is,

$$\mathcal{C}^h = \{c_k = \{\mathbf{x}_{k_1}, \mathbf{x}_{k_2}, \mathbf{x}_{k_3}, \mathbf{x}_{k_4}\} : \phi(\mathbf{x}_{k_i}) < 0 \text{ for at least one } i\}, \tag{34}$$

$$\mathcal{C}^h_{\partial\Omega} = \{c_k \in \mathcal{C}^h : \phi(\mathbf{x}_{k_i}) > 0 \text{ for at least one } i\}, \tag{35}$$

$$\mathcal{C}^h \setminus \mathcal{C}^h_{\partial\Omega} = \{c_k \in \mathcal{C}^h : \phi(\mathbf{x}_{k_i}) < 0, \forall i\}. \tag{36}$$

Next, we assemble the stiffness matrix $A$, the constraint matrix $B$, and the vectors $\vec{f}$ and $\vec{a}$ by looping over the cells $c_k \in \mathcal{C}^h$. The boundary contribution is described in Step 1 and the interior contribution is described in Step 2. Notice that if a node is not adjacent to any cell which is intersected by the boundary then the 5-point stencil is used.

**Step 1.** Adding boundary contribution to $A, \vec{f}$ and $B$

**for** $c_k \in \mathcal{C}^h_{\partial\Omega}$ **do**
  **for** $1 \leqslant i, j \leqslant 4$ **do**
    $A_{k_i k_j} += \bar{\beta} \int_{\Omega \cap c_k} \nabla N_{k_i} \cdot \nabla N_{k_j} \, d\mathbf{x}$ {see Fig. 4 for integration details}
  **end for**
  **for** $1 \leqslant i \leqslant 4$ **do**
    $f_{k_i} += \bar{f} \int_{\Omega \cap c_k} N_{k_i} \, d\mathbf{x}$ {see Fig. 4 for integration details}
    $l \leftarrow$ index of the coarse cell containing $c_k$ {see Fig. 7}
    $B_{l k_i} += \int_{c_k \cap \partial\Omega} N_{k_i} \, dS$
    $a_l += \bar{a} \int_{c_k \cap \partial\Omega} N_{k_i} \, dS$
  **end for**
**end for**

**Step 2.** Adding interior contribution to $A$ and $\vec{f}$

**for** $c_k \in \mathcal{C}^h \setminus \mathcal{C}^h_{\partial\Omega}$ **do**
  **for** $1 \leqslant i \leqslant 4$ **do**
    $A_{k_i k_i} += \bar{\beta}$
    $f_{k_i} += .25 h^2 \bar{f}$
    **for** $1 \leqslant j \leqslant 4$ **do**
      **if** $\mathbf{x}_{k_i} \neq \mathbf{x}_{k_j}$ and are edge connected **then**
        $A_{k_i k_j} -= .5\bar{\beta}$
      **end if**
    **end for**
  **end for**
**end for**

We compute the area integrals using a polygonal representation of $\Omega \cap c_k$ and the divergence theorem. The set of vertices of the polygon consists of all nodes with $\phi < 0$, as well as the two crossings $\mathbf{x}_c$ on the edges of the cell (see Fig. 4). Given a pair of nodes $\mathbf{x}_{k_i}$ and $\mathbf{x}_{k_j}$ with $\phi(\mathbf{x}_{k_i})\phi(\mathbf{x}_{k_j}) < 0$, we compute the edge crossing as

$$\theta = \frac{\phi(\mathbf{x}_{k_i})}{\phi(\mathbf{x}_{k_i}) - \phi(\mathbf{x}_{k_j})},$$

$$\mathbf{x}_c = \mathbf{x}_{k_j}\theta + \mathbf{x}_{k_i}(1 - \theta).$$

The divergence theorem converts the area integral of the second order polynomials $\nabla N_{k_i} \cdot \nabla N_{k_j}$ over the irregular polygon into a line integral of third order polynomials over the polygonal boundary (see Fig. 4). The integrals of the bilinear functions $N_{k_i}$ over $\partial\Omega \cap c_k$ are line integrals over the segment joining the two edge crossings. The simple low-order polynomials are integrated over each segment analytically.

The averages $\bar{f}$, $\bar{\beta}$ and $\bar{a}$ are also required. When $f$ is known at nodes, we use bilinear interpolation to represent it over a cell $c_k$, $f|_{c_k} = \sum_{i=1}^{4} f_{k_i} N_{k_i}(\mathbf{x})$. Then $\bar{f}$ may be computed by integrating this bilinear function over the material region and dividing by the area. These integrals are computed as an area integral using the divergence theorem as above. Thus the average $\bar{f}$ is,

$$\bar{f} = \frac{\sum_{i=1}^{4} f_{k_i} \int_{\Omega \cap c_k} N_{k_i}\, dx}{\text{Area}(\Omega \cap c_k)} = \frac{\sum_{i=1}^{4} f_{k_i} \int_{\Omega \cap c_k} N_{k_i}\, dx}{\sum_{i=1}^{4} \int_{\Omega \cap c_k} N_{k_i}\, dx}. \tag{37}$$

With $A$ and $B$ in hand, we re-order the degrees of freedom so that $B = (B_m | B_{n-m})$ with $B_m$ diagonal and non-singular. This amounts to finding the index $k_i$ of the degree of freedom at the center of the $l$th coarse cell, then permuting the degrees of freedom with indices $l$ and $k_i$ (see Fig. 7). Once we have re-ordered the degrees of freedom, the fundamental basis $Z$ and reduced constraints $\vec{c}$ can be easily computed (23). We then solve the system $Z^t A Z \vec{v} = Z^t(\vec{f} - A\vec{c})$ iteratively, performing multiplications with $Z, Z^t$ implicitly using the factors $B_m$ and $B_{n-m}$, and lastly recover the solution $\vec{u} = \vec{c} + Z\vec{v}$.

## 5. Numerical examples

This section presents a convergence test for each of the components of our method. We first demonstrate the expected second order accuracy for embedded Neumann and embedded Dirichlet problems in Sections 5.1 and 5.2, respectively, and for interface problems in Section 5.3. In Sections 5.3.2 and 5.3.3 we examine the performance of our method for the important special case when $\beta$ exhibits a large jump across the interface. Lastly, in Section 5.4 we demonstrate the effectiveness of this discontinuity removal technique on a $C^0$ Lipschitz segmented curve. The richer virtual node representation, as in Fig. 19, allows us to achieve second order results for a non-smooth interface while still retaining the standard 5-point finite difference stencil.

We ran all of the examples on a sequence of $N \times N$ grids, for $80 \leqslant N \leqslant 800$. Each grid ranges from $-1 \leqslant x \leqslant 1$, $-1 \leqslant y \leqslant 1$. The error plots depict $\log_{10}\|e\|_{L^\infty}$ versus $\log_{10} N$. The examples include both level set representations and Lagrangian representations of the interface. For interfaces that have more detail than the background grid can resolve, using a level set introduces non-negligible geometric regularization. See Section 6 for a discussion of the geometric precision of our method.

### 5.1. Embedded Neumann

We demonstrate the method applied to the embedded Neumann problem

$$\begin{aligned} &-\nabla \cdot \beta(\mathbf{x}) \nabla u = f, \quad \forall \mathbf{x} \in \Omega, \\ &\beta(\mathbf{x}) \nabla u \cdot n = q(\mathbf{x}), \quad \forall \mathbf{x} \in \partial\Omega_n. \end{aligned}$$

Here $\beta(\mathbf{x}) = 4 + x + y$. We chose the parameters $q$ and $f$ using the exact solution

$$u = (x^3 - y^3)\cos(x + y).$$

The embedded Neumann boundary, $\partial\Omega_n$, is given by the 5-pointed star with vertices

$$\begin{aligned} t_0 &= .1243, \\ r_i &= .35 + .3(i \bmod 2), \\ X_i &= r_i \cos\left(\frac{\pi i}{5} + t_0\right), \\ Y_i &= r_i \sin\left(\frac{\pi i}{5} + t_0\right), \end{aligned}$$

for $1 \leqslant i \leqslant 10$, represented as a Lagrangian curve. See Fig. 9 for the error plot. A least squares regression estimates the order of accuracy as 1.95.

### 5.2. Embedded Dirichlet

We demonstrate the method applied to the embedded Dirichlet problem

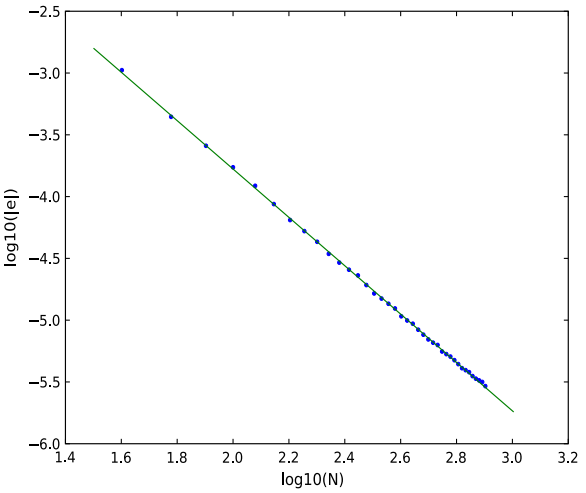
$$\begin{aligned} &\Delta u = 0, \quad \forall \mathbf{x} \in \Omega, \\ &u = p(\mathbf{x}), \quad \forall \mathbf{x} \in \partial\Omega_d = \partial\Omega. \end{aligned}$$

We chose the Dirichlet condition $p$ using the chosen exact solution

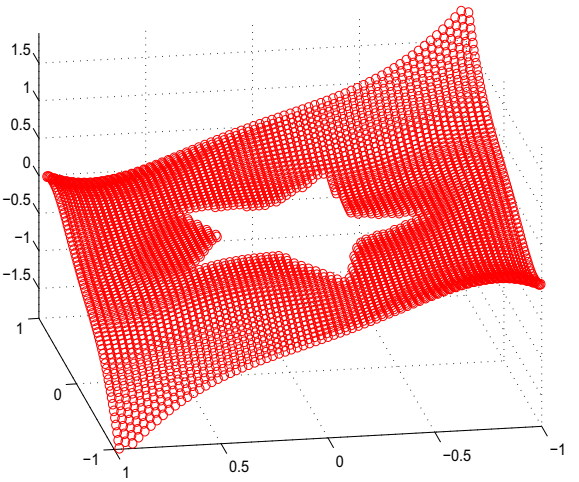
$$u = x^2 - y^2.$$

The embedded boundary $\partial\Omega$ is given by the curve

$$\begin{aligned} t_0 &= .00132, \\ r_0 &= .02\sqrt{5}, \\ r(t) &= .5 + .2\sin(5t), \\ X(\theta) &= r_0 + r(\theta + t_0)\cos(\theta + t_0), \\ Y(\theta) &= r_0 + r(\theta + t_0)\sin(\theta + t_0), \end{aligned}$$

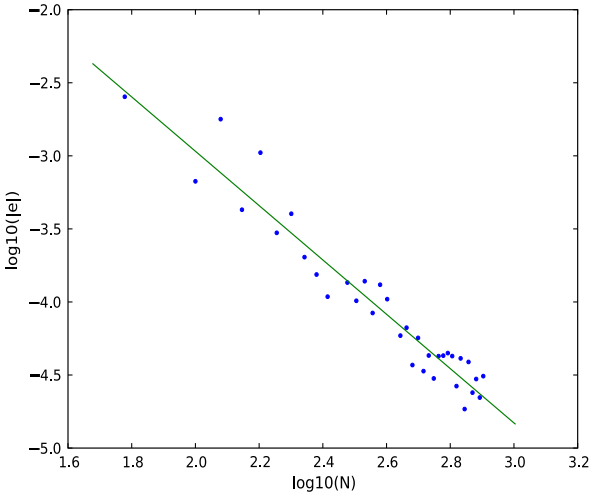



(a) Estimated order: 1.95

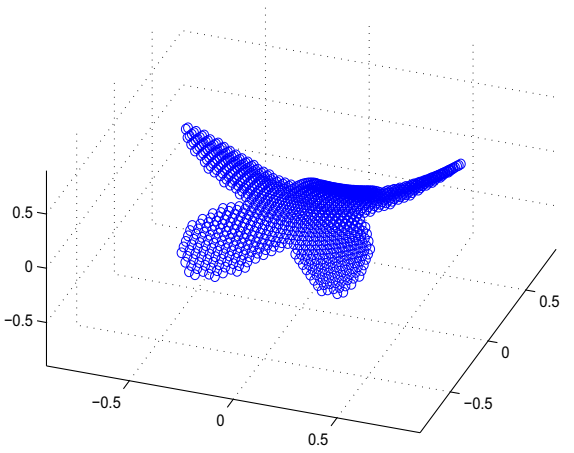



(b) Embedded Neumann problem on an exterior domain

**Fig. 9.** Numerical results for example Section 5.1.




(a) Estimated order: 1.86




(b) Embedded Dirichlet problem on an interior domain

**Fig. 10.** Numerical results for example Section 5.2.

represented as a Lagrangian curve. See Fig. 10 for the error plot. A least squares regression estimates the order of accuracy as 1.86.

### 5.3. Embedded interface

#### 5.3.1. Embedded interface example 1

We demonstrate the method applied to the embedded interface problem

$$
\begin{aligned}
&-\nabla\cdot(\beta(\mathbf{x})\nabla u) = f(\mathbf{x}), \quad \forall \mathbf{x}\in\Omega\setminus\Gamma,\\
&[u] = a(\mathbf{x}),\\
&[\beta(\mathbf{x})\nabla u\cdot n] = b(\mathbf{x}), \quad \forall \mathbf{x}\in\Gamma.
\end{aligned}
$$

Here $\beta(\mathbf{x}) = 4 + \sin(x+y)$ in the interior and $\beta(\mathbf{x}) = 2 + x^2 + y^2$ in the exterior. We chose the parameters $a$, $b$ and $f$ using the exact solution

$$u^- = \cos(y)\sin(x),$$
$$u^+ = 1 - x^2 - y^2.$$

The interface is given by the curve parametrized by

$$t_0 = .45234,$$
$$\theta(t) = t + \sin(4t),$$
$$r(t) = .60125 + .24012\cos(4t + \pi/2),$$
$$X(t) = r(t+t_0)\cos(\theta(t+t_0)),$$
$$Y(t) = r(t+t_0)\sin(\theta(t+t_0))$$

for $0 \leqslant t \leqslant 2\pi$ represented with a level set. See Fig. 11 for a plot of the error in the solution and in the gradient evaluated on the interface. A least squares regression estimates the order of accuracy of the solution as 1.92 and the order of accuracy of the gradient as.96. The gradient was evaluated point-wise at the mid-point $\mathbf{x}_M$ of the interface segment in each cell by differentiating the bilinear basis elements, that is, $\nabla u(\mathbf{x}_M) = \sum_{i=1}^{4} u_i \nabla N_i(\mathbf{x}_M)$.

*5.3.2. Embedded interface example 2*

In this example we examine the performance of the method in the case of when the coefficient $\beta$ has a large jump across the interface. The following example was taken from [32]. We solve the interface problem

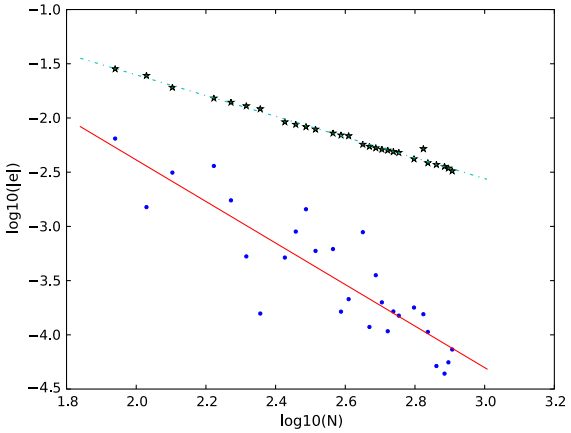
$$\nabla \cdot \beta(\mathbf{x})\nabla u = f, \quad \forall \mathbf{x} \in \Omega \setminus \Gamma,$$
$$[u] = a(\mathbf{x}),$$
$$[\nabla u \cdot n] = b(\mathbf{x}), \quad \forall \mathbf{x} \in \Gamma.$$

Here we take the coefficient to be piecewise constant, $\beta(\mathbf{x}) = \beta^+$ in the exterior and $\beta(\mathbf{x}) = \beta^-$ on the interior. We chose the parameters $a$, $b$ and $f$ using the exact solution

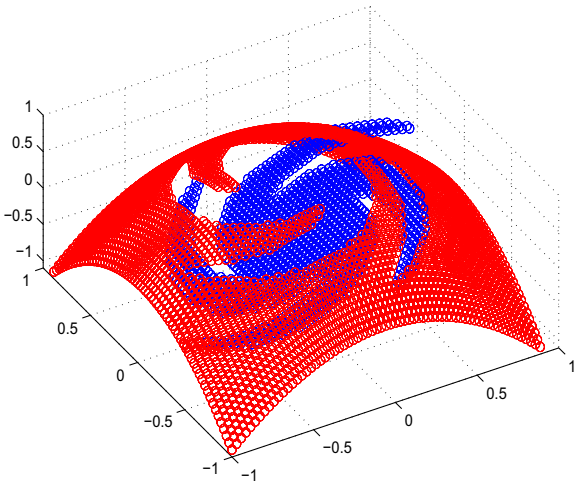
$$u^- = x^2 + y^2,$$
$$u^+ = .1(x^2+y^2)^2 - .01\ln(2\sqrt{x^2+y^2}).$$

The interface $\Gamma$ is given by the curve used in example Section 5.2,

$$t_0 = .00132,$$
$$r_0 = .02\sqrt{5},$$
$$r(t) = .5 + .2\sin(5t),$$
$$X(t) = r_0 + r(t+t_0)\cos(t+t_0),$$
$$Y(t) = r_0 + r(t+t_0)\sin(t+t_0),$$

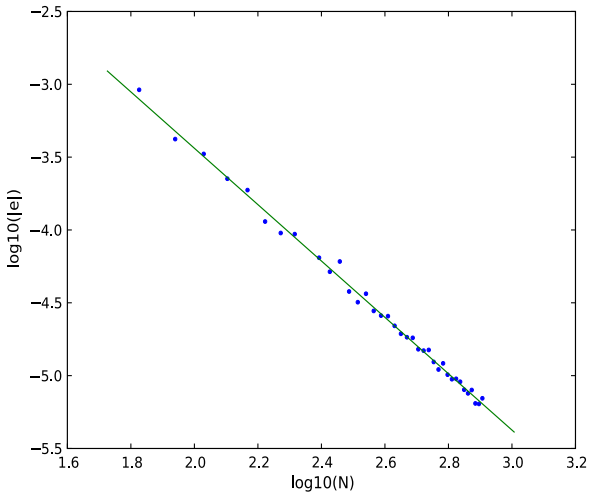



(a) Estimated order: 1.92 for solution (solid line) and .96 for gradient along the interface (dotted line)

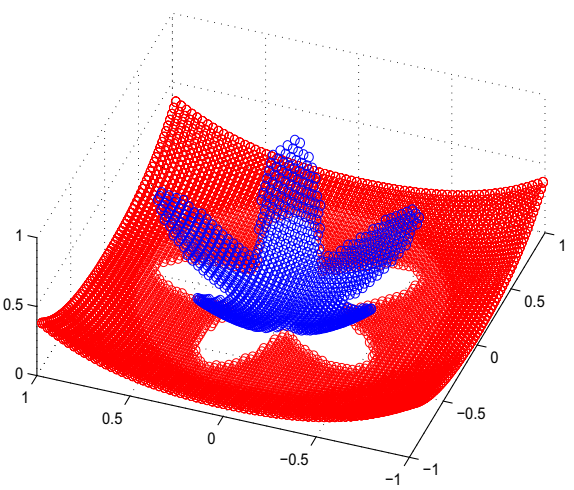



(b) Embedded interface problem with discontinuous coefficients

**Fig. 11.** Numerical results for example Section 5.3.

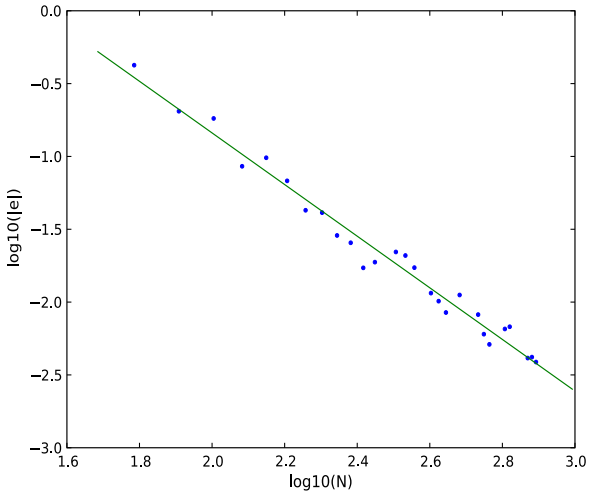



(a) Estimated order: 1.94, $\beta^- : \beta^+ = 1 : 10$

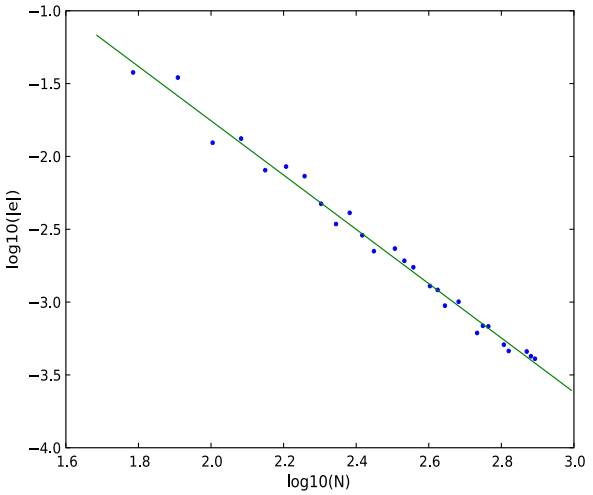


(b) Embedded interface problem with high contrast coefficients

**Fig. 12.** Numerical results for example Section 5.3.2.




(a) Estimated order: 1.86, $\beta^- : \beta^+ = 1 : 1000$




(b) Estimated order: 1.77, $\beta^- : \beta^+ = 1000 : 1$

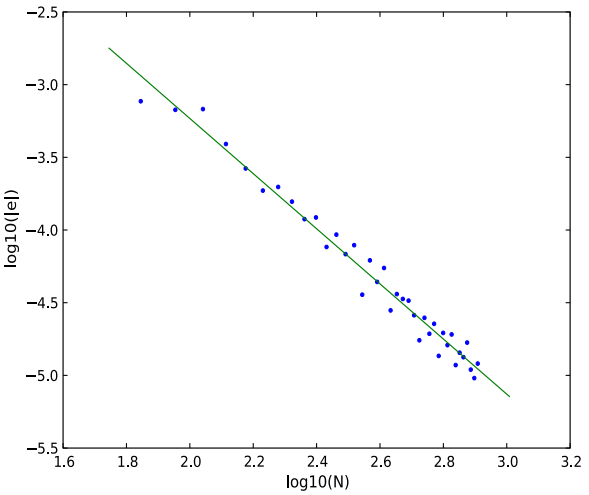
**Fig. 13.** Numerical results for example Section 5.3.2.

| Test case | Condition number (before IC) | Condition number (after IC) | PCG iterations | Time (sec) |
|---|---|---|---|---|
| 1:1 | $7.60x10^5$ | $3.42x10^4$ | 1106 | 95.19 |
| 1000:1 | $4.43x10^8$ | $2.13x10^7$ | 1803 | 157.45 |
| 1:1000 | $6.45x10^5$ | $4.59x10^4$ | 1751 | 150.27 |
| 5-point stencil | $2.58x10^5$ | $2.28x10^4$ | 723 | 60.23 |

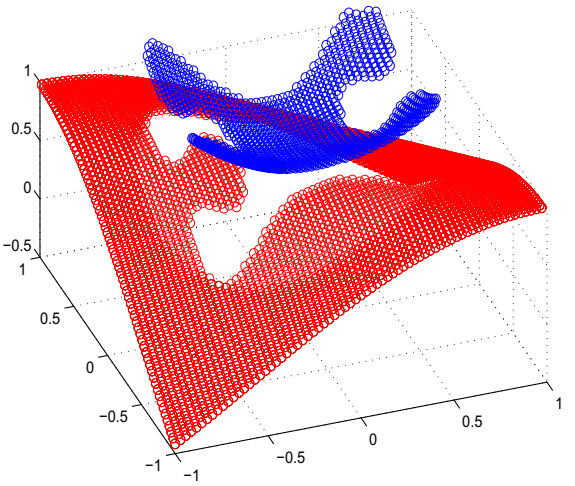
**Fig. 14.** Condition numbers of linear system and clock time of PCG at resolution $800 \times 800$ for example Section 5.3.2.

for $0 \leqslant t \leqslant 2\pi$. See Figs. 12 and 13 for a plot of the error for three values of the ratio $\beta^-:\beta +-$, 1:10, 1:1000 and 1000:1. A least squares regression estimated the order of accuracies as 1.94 for 1:10, 1.86 for 1:1000 and 1.77 for 1000:1.

See Fig. 14 for the number of Conjugate Gradient iterations, computer time in seconds and condition numbers of the linear systems before and after incomplete Cholesky preconditioning. Fig. 14 compares the performance to the standard 5-point Laplacian on a square with no interface as a reference. All tests were run with grid resolution $800 \times 800$ and to residual norm tolerance of $10^{-12}$. The linear system was solved using the PETSc Conjugate Gradient with the PETSc incomplete Cholesky




(a) Estimated order: 1.90



(b) Embedded interface problem with high contrast coefficients

**Fig. 15.** Numerical results for example Section 5.3.3.

preconditioner [72–74]. The code was run in serial on a 2.8 GHz laptop computer. All linear systems were normalized to have a constant diagonal before the preconditioner was applied. The high coefficient ratios incur a moderate cost but are still comparable to the standard 5-point discretization.

*5.3.3. Embedded interface example 3*

In this example we again examine the performance of the method in the case of when the coefficient $\beta$ has a large jump across the interface. We solve the interface problem

$$\nabla \cdot \beta(\mathbf{x})\nabla u = f, \quad \forall \mathbf{x} \in \Omega \setminus \Gamma,$$
$$[u] = a(\mathbf{x}),$$
$$[\nabla u \cdot n] = b(\mathbf{x}), \quad \forall \mathbf{x} \in \Gamma.$$

Here we take the coefficient to be piecewise constant, $\beta(\mathbf{x}) = \beta^+$ in the exterior and $\beta(\mathbf{x}) = \beta^-$ on the interior. We chose the parameters $a$, $b$ and $f$ using the exact solution

$$u^- = x^2 + y^2 + 1,$$
$$u^+ = \cos(x + y).$$

The interface $\Gamma$ is given by the curve,

$$\theta_0 = .00132,$$
$$X(\theta) = .6\cos(\theta + \theta_0) - .3\cos(\theta + \theta_0),$$
$$Y(\theta) = .47\sin(\theta + \theta_0) - .0047\sin(3\theta - 3\theta_0) + .13\sin(7\theta - 7\theta_0),$$

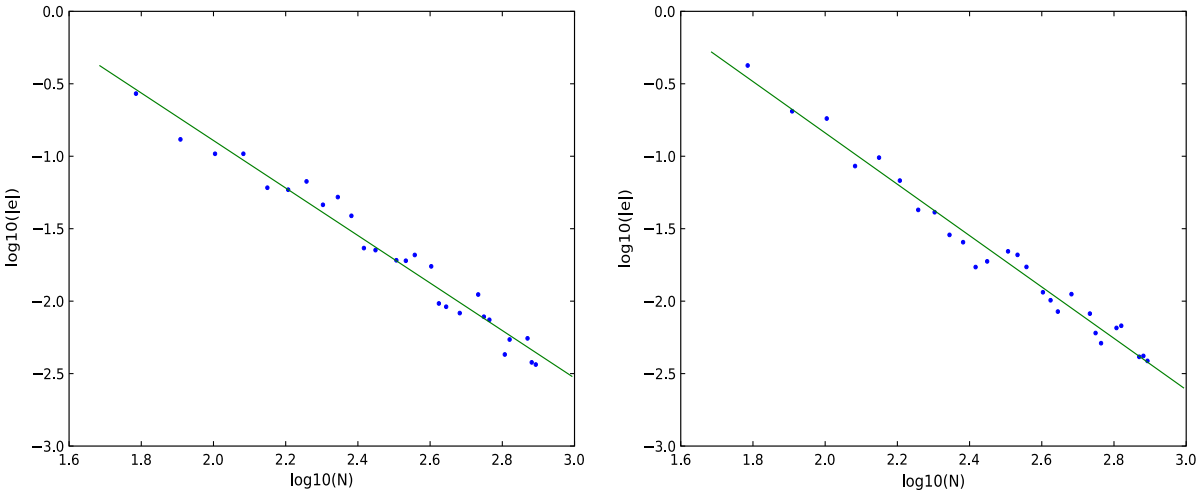
for $0 \leqslant \theta \leqslant 2\pi$. See Figs. 15 and 16 for a plot of the error for three values of the ratio $\beta^-$:$\beta^+$, 1:10, 1:1000 and 1000:1. A least squares regression estimated the order of accuracies as 1.90 for 1:10, 1.64 for 1:1000 and 1.77 for 1000:1.

See Fig. 17 for the number of Conjugate Gradient iterations, computer time in seconds and condition numbers of the linear systems before and after incomplete Cholesky preconditioning. Fig. 17 compares the performance to the standard 5-point Laplacian on a square with no interface as a reference. All tests were run with grid resolution $800 \times 800$ and to residual norm tolerance of $10^{-12}$. The linear system was solved using the PETSc Conjugate Gradient with the PETSc incomplete Cholesky preconditioner [72–74]. The code was run in serial on a 2.8 GHz laptop computer. All linear systems were normalized to have a constant diagonal before the preconditioner was applied. The high coefficient ratios incur a moderate cost but are still comparable to the standard 5-point discretization.

*5.4. Discontinuity removal*

We solve the interface problem

$$-\Delta u = f, \quad \forall \mathbf{x} \in \Omega \setminus \Gamma,$$
$$[u] = a(\mathbf{x}),$$
$$[\nabla u \cdot n] = b(\mathbf{x}), \forall \mathbf{x} \in \Gamma.$$

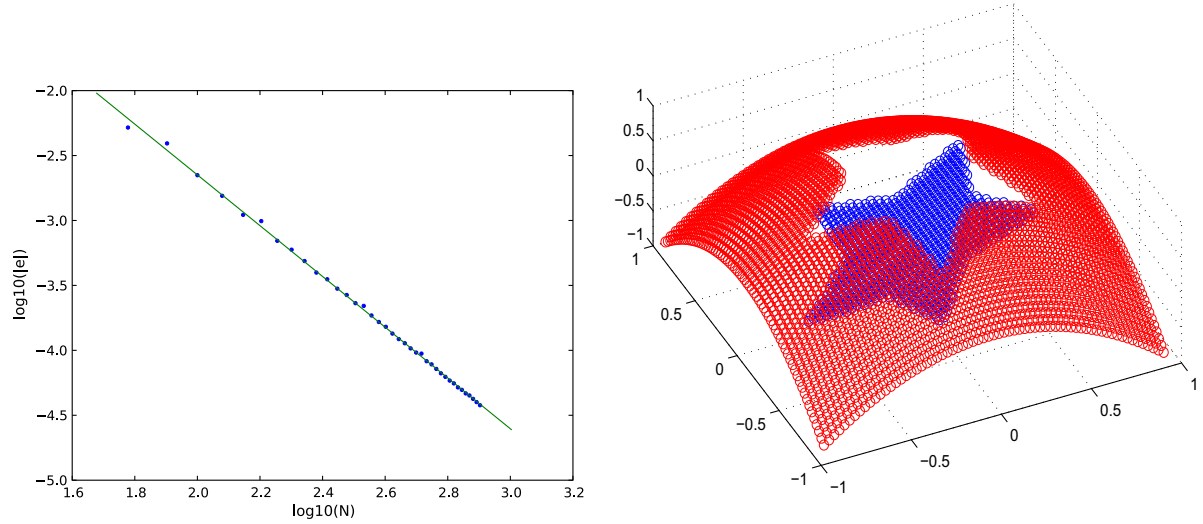



(a) Estimated order: 1.64, $\beta^- : \beta^+ = 1 : 1000$ (b) Estimated order: 1.77, $\beta^- : \beta^+ = 1000 : 1$

**Fig. 16.** Numerical results for example Section 5.3.2.

| Test case | Condition number (before IC) | Condition number (after IC) | PCG iterations | Time (sec) |
|---|---|---|---|---|
| 1:1 | $5.99x10^5$ | $3.57x10^4$ | 1082 | 82.97 |
| 1000:1 | $3.85x10^8$ | $2.13x10^7$ | 1802 | 138.04 |
| 1:1000 | $7.25x10^5$ | $7.25x10^4$ | 1915 | 145.78 |
| 5-point stencil | $2.58x10^5$ | $2.28x10^4$ | 723 | 60.23 |

**Fig. 17.** Condition numbers of linear system and clock time of PCG at resolution 800 × 800 for example Section 5.3.3.




(a) Estimated order: 1.96 (b) Embedded interface problem solved with discontinuity removal

**Fig. 18.** Numerical results for example Section 5.4.

We chose the parameters $a$, $b$ and $f$ using the exact solution

$$u^- = \cos(y)\sin(x),$$
$$u^+ = 1 - x^2 - y^2.$$

The interface is the segmented 5-pointed star with vertices

$$t_0 = .1243,$$
$$r_i = .35 + .3(i \bmod 2),$$
$$X_i = r_i \cos\left(\frac{\pi i}{5} + t_0\right),$$
$$Y_i = r_i \sin\left(\frac{\pi i}{5} + t_0\right),$$

for $1 \leqslant i \leqslant 10$, represented with a Lagrangian curve. See Fig. 18 for the error plot. A least squares regression estimates the order of accuracy as 1.96.

## 6. Discussion and conclusion

Our method uses a virtual node concept coupled with a Lagrange multiplier formulation to approximate the solution of the elliptic interface problem (1)–(3) and the related embedded Neumann and Dirichlet problems. Notably, the symmetric positive definite discretization and intuitive, geometric nature of the method make it easy to implement. Numerical examples demonstrate second order convergence in $L^\infty$.

Certain FEM approaches that also use virtual nodes [7–10,13,14] parallel our work in spirit and implementation. Moreover, for typical geometric cases, virtual node representations yield the same space as those given by Heaviside enrichment XFEM approaches [9]. Such methods generally use lower-order triangular elements that do not permit the obvious choice of Lagrange multiplier space $\mathbf{\Lambda}^h$ of one constraint per element. Unstructured triangular meshes generally do not achieve second order accuracy in $L^\infty$ either. These facts motivate the interest in regular Cartesian bilinear elements that both achieve second order accuracy in $L^\infty$ and permit straightforward Lagrange multiplier spaces. For instance, the XFEM approach of [58] that uses Cartesian bilinear elements with the Lagrange multiplier space $\mathbf{\Lambda}^h$ also demonstrated second order accuracy in $L^\infty$.

By design, our choice of Lagrange multiplier space eases the computational effort and memory limitations imposed by the saddle-point problem at the cost of accuracy, as the pairing $(\mathbf{V}^h, \mathbf{\Lambda}^{2h})$ results in higher $L^\infty$ error than the choice $(\mathbf{V}^h, \mathbf{\Lambda}^h)$. Our numerical experiments indicate that our choice does not sacrifice second order convergence. Generally the approximations of $\mathbf{H}^{-1/2}(\partial\Omega)$ and $\mathbf{H}^1(\Omega)$ must satisfy an inf–sup condition uniformly in $h$ for the solution to exhibit optimal convergence rates. See [61] for several characterizations of the relevant inf–sup conditions. However, as our numerical experiments indicate and the following argument demonstrates, our choice does not sacrifice such inf–sup stability. Assume the pairing $(\mathbf{V}^h, \mathbf{\Lambda}^h)$ satisfies an inf–sup condition uniformly in $h$, that is, if there exist $\gamma_0$, $h_0 > 0$ such that, for all $h_0 \geqslant h > 0$,

$$\inf_{\mu^h \in \mathbf{\Lambda}^h} \sup_{v^h \in \mathbf{V}^h} \alpha(\mu^h, v^h) \geqslant \gamma_0$$

for some function $\alpha : \mathbf{H}^{-1/2}(\partial\Omega) \times \mathbf{H}^1(\Omega) \to \mathbb{R}$. Then whenever $2h \leqslant h_0$, as $\mathbf{V}^{2h} \subset \mathbf{V}^h$

$$\gamma_0 \leqslant \inf_{\mu^h \in \mathbf{\Lambda}^{2h}} \sup_{v^h \in \mathbf{V}^{2h}} \alpha(\mu^h, v^h) \leqslant \inf_{\mu^h \in \mathbf{\Lambda}^{2h}} \sup_{v^h \in \mathbf{V}^h} \alpha(\mu^h, v^h), \tag{38}$$

so that our pairing $(\mathbf{V}^h, \mathbf{\Lambda}^{2h})$ satisfies the same inf–sup condition uniformly in $h$ as well. Moreover, the above argument holds if we begin with satisfactory constraints on any grid coarser than $\mathcal{G}^h$ and then refine the corresponding space to obtain $\mathbf{V}^h$. In practice, we begin with the matrix $B$ that results from using $(\mathbf{V}^h, \mathbf{\Lambda}^h)$. We then add together any constraints that lie in the same cell $\hat{c}_k \in \mathcal{G}^{2h}$ to arrive at the constraints for the pairing $(\mathbf{V}^h, \mathbf{\Lambda}^{2h})$. The grid $\mathcal{G}^{2h}$ merely serves as an easy means of determining which rows to sum to obtain a diagonal sub-matrix $B_m$. In theory, we could sum rows in some other fashion, so long as the resulting constraint corresponds to an inf–sup stable constraint from a coarser grid.

In our numerical examples we give results using both a level set representation of the interface and a segmented Lagrangian representation of the interface. In principle, our method does not rely upon a particular representation of $\Gamma$. However, for high curvature interfaces such as the examples of Chen and Strain [27], using a Lagrangian representation results in an interface with significantly more detail than the background grid can resolve. This can result in a grid cell $c_k \in \mathcal{G}^h$ that contains two or more disconnected segments of the interface (see Fig. 19). We found that, in this case, enforcing one constraint per cell results in unsatisfactory accuracy, and we have yet to attempt to resolve this issue in full. For smooth interfaces, this will always vanish under refinement, and using a level set representation generally prevents this phenomenon. However, for complex interfaces, the transfer to a level set involves non-negligible regularization. Moreover, our numerical experiments suggest we actually must guarantee none of the cells $\hat{c}_k \in \mathcal{C}_\Gamma^{2h}$ contain disconnected interface segments in order to retain optimal accuracy in the pre-asymptotic regime. As they do not rely on Lagrange multipliers, this does not present a challenge to either embedded Neumann or our discontinuity removal technique. For instance, in example Section 5.4 we used the richer virtual node representation, illustrated in the right column of Fig. 19, to appropriately handle the disconnected interface segments. The case of interface and embedded Dirichlet problems for non-smooth or poorly resolved curves will be treated in future research.

In our numerical examples, we solve the reduced saddle-point problem with a straightforward application of Conjugate Gradient with Jacobi preconditioning on examples Sections 5.1, 5.2 and 5.4, and used PETSc Conjugate Gradient with

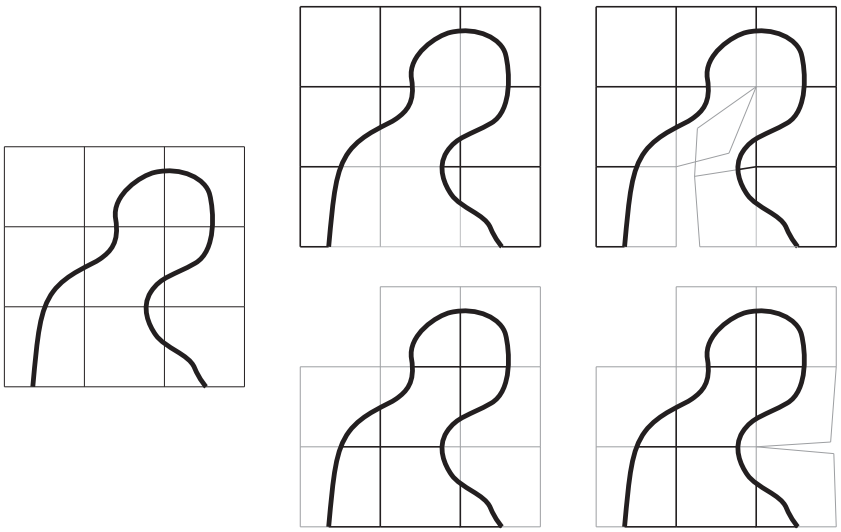


**Fig. 19.** Virtual node representation. On the left, a portion of an interface passes through the unduplicated grid. The images in the center column show the result of applying the duplications schemes of [8–10]. This gives at most two degrees of freedom per original node. The right column shows the richer representation given by the virtual node algorithm. The cell in the center contains two disconnected interface segments. In this case, these segments lie in distinct cells after duplication. Including both cells in the same constraint in $B$ degrades accuracy.

incomplete Cholesky preconditioning [72–74] for examples Sections 5.3, 5.3.2 and 5.3.3. The use of our method in practical applications, such as multiphase fluid dynamics, will require more efficient preconditioning. Moreover, we have not proposed a method for efficiently dealing with large jumps in $\beta$ at the interface (see Sections 5.3.2 and 5.3.3 for the performance of the method in this case). Of course, in the discontinuity removal method of Section 3.3.1, fast Poisson solvers may be applied.

## Acknowledgments

The authors thank Aleka McAdams and Jeffrey Hellrung for the their contributions. We would also like to thank Professor Tom Beale and Professor Ron Fedkiw for their helpful discussions. We used the SuiteSparse [75] numerics library with the Goto BLAS in the course of our research and to perform the extrapolations required for the discontinuity removal example Section 5.4. This work was supported in part by NSF DMS-0502315, NSF DMS-0652427, NSF CCF-0830554, DOE 09-LR-04-116741-BERA and ONR N000140310071.

## References

[1] I. Babuška, The finite element method for elliptic equations with discontinuous coefficients, Computing 5 (1970) 207–213.
[2] J. Bramble, J. King, A finite element method for interface problems in domains with smooth boundaries and interfaces, Adv. Comput. Math. 6 (1996) 109–138.
[3] B. Lamichhane, B. Wohlmuth, Mortar finite elements for interface problems, Computing 72 (2004) 333–348.
[4] J. Huang, J. Zou, A mortar element method for elliptic problems with discontinuous coefficients, IMA J. Numer. Anal. 22 (2002) 549–576.
[5] M. Dryja, A Neumann–Neumann algorithm for a mortar discretization of elliptic problems with discontinuous coefficients, I, J. Numer. Methods Eng. 99 (2005) 645–656.
[6] Z. Chen, J. Zou, Finite element methods and their convergence for elliptic and parabolic interface problems, Numer. Math. 79 (1998) 175–202.
[7] A. Hansbo, P. Hansbo, An unfitted finite element method, based on Nitsche's, method for elliptic interface problems, Comput. Methods Appl. Mech. Eng. 191 (2002) 5537–5552.
[8] A. Hansbo, P. Hansbo, A finite element method for the simulation of strong and weak discontinuities in solid mechanics, Comput. Methods Appl. Mech. Eng. 193 (2004) 3523–3540.
[9] J.-H. Song, P. Areias, T. Belytschko, A method for dynamic crack and shear band propagation with phantom nodes, I, J. Numer. Methods Eng. 67 (2006) 868–893.
[10] J. Dolbow, I. Harari, An efficient finite element method for embedded interface problems, I, J. Numer. Methods Eng. 78 (2009) 229–252.
[11] E. Sifakis, K. Der, R. Fedkiw, Arbitrary cutting of deformable tetrahedralized objects, ACM SIGGRAPH/Eurgraphics Symp. Comput. Anim.
[12] N. Molino, Z. Bao, R. Fedkiw, A virtual node algorithm for changing mesh topology during simulation SIGGRAPH 2004, ACM TOG 23 (2004) 385–392.
[13] Z. Bao, J. Hong, J. Teran, R. Fedkiw, Fracturing rigid materials, IEEE Trans. Vis. Comput. Graph. 13 (2) (2007) 370–378.
[14] C. Richardson, J. Hegemann, E. Sifakis, J. Hellrung, J. Teran, Simulating crack propagation with xfem and a hybrid mesh, I, J. Numer. Methods Eng., submitted for publication.
[15] Z. Li, K. Ito, The Immersed Interface Method: Numerical Solutions of PDEs Involving Interfaces and Irregular Domains (Frontiers in Applied Mathematics), Society for Industrial and Applied Mathematics, 2006.
[16] R.J. LeVeque, Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (1994) 1019–1044.
[17] R.J. LeVeque, Z. Li, Immersed interface methods for stokes flow in elastic boundaries or surface tension, SIAM J. Sci. Comput. 18 (1997) 1019–1044.

[18] Z. Tan, D. Le, Z. Li, K. Lim, B. Khoo, An immersed interface method for solving incompressible viscous flows with piecewise constant viscosity across a moving elastic membrane, J. Comput. Phys. 227 (2008) 9955–9983.
[19] D. Le, B. Khoo, J. Peraire, An immersed interface method for viscous incompressible flows involving rigid and flexible boundaries, J. Comput. Phys. 220 (2006) 109–138.
[20] S. Xu, Z. Wang, A 3d immersed interface method for fluid–solid interaction, Comput. Methods Appl. Mech. Eng. 197 (2008) 2068–2086.
[21] S. Xu, Z. Wang, An immersed interface method for simulating the interaction of a fluid with moving boundaries, J. Comput. Phys. 216 (2006) 454–493.
[22] Z. Li, M.-C. Lai, The immersed interface methods for the Navier–Stokes equations with singular forces, SIAM J. Sci. Comput. 171 (2001) 822–842.
[23] L. Lee, R.J. LeVeque, An immersed interface method for incompressible Navier–Stokes equations, SIAM J. Sci. Comput. 25 (2003) 832–856.
[24] J. Beale, A. Layton, On the accuracy of finite difference methods for elliptic problems with interfaces, Commun. Appl. Math. Comput. Sci. 1 (2006) 207–208.
[25] Z. Li, K. Ito, Maximum principle preserving schemes for interface problems with discontinuous coefficients, SIAM J. Sci. Comput. 23 (2001) 339–361.
[26] S. Deng, K. Ito, Z. Li, Three-dimensional elliptic solvers for interface problems and applications, J. Comput. Phys. 184 (1) (2003) 215–243.
[27] T. Chen, J. Strain, Piecewise-polynomial discretization and Krylov-accelerated multigrid for elliptic interface problems, J. Comput. Phys. 227 (2008) 7503–7542.
[28] A. Weigmann, K. Bube, The explicit-jump immersed interface method: finite difference methods for PDEs with piecewise smooth solutions, SIAM J. Numer. Anal. 37 (2000) 827–862.
[29] P.A. Berthelsen, A decomposed immersed interface method for variable coefficient elliptic equations with non-smooth and discontinuous solutions, J. Comput. Phys. 197 (2004) 364–386.
[30] Z. Li, A fast iterative algorithm for elliptic interface problems, SIAM J. Numer. Anal. 35 (1998) 230–254.
[31] L. Adams, T. Chartier, A comparison of algebraic multigrid and geometric immersed interface multigrid methods for interface problems, SIAM J. Sci. Comput. 26 (2005) 762–784.
[32] X. Liu, R. Fedkiw, M. Kang, A boundary condition capturing method for Poisson's equation on irregular domains, J. Comput. Phys. 160 (1) (2000) 151–178. doi:http://dx.doi.org/10.1006/jcph.2000.6444.
[33] I.-L. Chern, Y.-C. Shu, A coupling interface method for elliptic interface problems, J. Comput. Phys. 225 (2007) 2138–2174.
[34] Y. Zhou, S. Zhao, M. Feig, G. Wei, High order matched interface and boundary method for elliptic equations with discontinuous coefficients and singular sources, J. Comput. Phys. 213 (2006) 1–30.
[35] F. Gibou, R. Fedkiw, L.-T. Cheng, M. Kang, A second-order accurate symmetric discretization of the Poisson equation on irregular domains, J. Comput. Phys. 176 (2002) 205–227.
[36] F. Gibou, R. Fedkiw, A fourth order accurate discretization for the laplace, heat equations on arbitrary domains, with applications to the Stefan problem, J. Comput. Phys. 202 (2005) 577–601.
[37] Z. Jomaa, C. Macaskill, The embedded finite difference method for the poisson equation in a domain with an irregular boundary and Dirichlet boundary conditions, J. Comput. Phys. 202 (2005) 488–506.
[38] X. Liu, T. Sideris, Convergence of the ghost fluid method for elliptic equations with interfaces, Math. Comput. 72 (244) (2003) 1731–1746.
[39] Y. Zhou, G. Wei, On the fictitious-domain and interpolation formulations of the matched interface and boundary (MIB) method, J. Comput. Phys. 219 (2006) 228–246.
[40] S. Yu, G. Wei, Three-dimensional matched interface and boundary (MIB) method for treating geometric singularities, J. Comput. Phys. 227 (2007) 602–632.
[41] S. Hou, X. Liu, A numerical method for solving variable coefficient elliptic equation with interfaces, J. Comput. Phys. 202 (2005) 411–445.
[42] Y. Ng, C. Min, F. Gibou, An efficient fluid–solid coupling algorithm for single-phase flows, J. Comput. Phys. 228 (2009) 8807–8829.
[43] J. Papac, F. Gibou, C. Ratsch, Efficient symmetric discretization for the poisson, heat and Stefan-type problems with robin boundary conditions, J. Comput. Phys. 229 (2010) 875–889.
[44] R. Glowinski, T. Pan, J. Periaux, A fictitious domain method for Dirichlet problem and applications, Comput. Methods Appl. Mech. Eng. 111 (1994) 283–303.
[45] L. Parussini, V. Pediroda, Fictitious Domain approach with hp-finite element approximation for incompressible fluid flow, J. Comput. Phys. 228 (10) (2009) 3891–3910.
[46] H. Mourad, J. Dolbow, I. Harari, A bubble-stabilized finite element method for Dirichlet constraints on embedded interfaces, I, J. Numer. Methods Eng. 69 (2007) 1–21.
[47] A. Almgren, J.B. Bell, P. Colella, T. Marthaler, A cartesian grid projection method for the incompressible Euler equations in complex geometries, SIAM J. Sci. Comput. 18 (5) (1997) 724–725.
[48] D. Young, R. Melvin, M. Bieterman, F. Johnson, S. Samant, J. Bussoletti, A locally refined rectangular grid finite element method: application to computational fluid dynamics and computational physics, J. Comput. Phys. 92 (1991) 1–66.
[49] A. Lew, G. Buscaglia, A discontinuous-Galerkin-based immersed boundary method,I, J. Numer. Methods Eng. 76 (2008) 427–454.
[50] J. Dolbow, L. Franca, Residual-free bubbles for embedded Dirichlet problems, Comput. Methods Appl. Mech. Eng. 197 (2008) 3751–3759.
[51] T. Belytschko, N. Moës, S. Usui, C. Parimi, Arbitrary discontinuities in finite elements, I, J. Numer. Methods Eng. 50 (2001) 993–1013.
[52] N. Moës, J. Dolbow, T. Belytschko, A finite element method for crack growth without remeshing, I, J. Numer. Methods Eng. 46 (1999) 131–150.
[53] C. Daux, N. Moës, J. Dolbow, N. Sukumar, T. Belytschko, Arbitrary branched and intersecting cracks with the extended finite element method, I, J. Numer. Methods Eng. 48 (2000) 1741–1760.
[54] N. Moës, M. Cloirec, P. Cartraud, J. Remacle, A computational approach to handle complex microstructure geometries, Comput. Methods Appl. Mech. Eng. 192 (2000) 3163–3177.
[55] H. Ji, J. Dolbow, On strategies for enforcing interfacial constraints and evaluating jump conditions with extended finite element method,I, J. Numer. Methods Eng. 61 (2004) 2508–2535.
[56] S. Groß, A. Reusken, An extended pressure finite element space for two-phase incompressible flows with surface tension, J. Comput. Phys. 224 (2007) 40–58.
[57] N. Moës, E. Béchet, M. Tourbier, Imposing essential boundary conditions in the extended finite element method, I, J. Numer. Methods Eng. 67 (2006) 1641–1669.
[58] B. Vaughan, B. Smith, D. Chopp, A comparison of the extended finite element method with the immersed interface method for elliptic equations with discontinuous coefficients and singular sources, Commun. Appl. Math. Comput. Sci. 1 (2006) 207–228.
[59] J. Beale, D. Chopp, R. LeVeque, Z. Li, Correction to the article a comparison of the extended finite element method with the immersed interface method for elliptic equations with discontinuous coefficients and singular sources by Vaughan et al., Commun. Appl. Math. Comput. Sci. 3 (2008) 95–101.
[60] I. Babuška, The finite element method with Lagrangian multipliers, Numer. Math. 20 (1973) 179–192.
[61] J. Pitkäranta, Boundary subspaces for the finite element method with Lagrange multipliers, Numer. Math. 33 (1979) 273–289.
[62] D. Chapelle, K. Bathe, The inf–sup test, Comput. Struct. 47 (1993) 537–545.
[63] A. Kumar, S. Padmanabhan, R. Burla, Implicit boundary method for finite element analysis using non-conforming mesh or grid, I, J. Numer. Methods Eng. 74 (2008) 1421–1447.
[64] T. Fries, T. Belytschko, The intrinsic xfem: a method for arbitrary discontinuities without additional unknowns, I, J. Numer. Methods Eng. 68 (2006) 1358–1385.
[65] Z. Li, The immersed interface method using a finite element formulation, Appl. Numer. Math. 27 (1998) 253–267.
[66] Z. Li, T. Lin, X. Wu, New cartesian grid methods for interface problems using the finite element formulation, Numer. Math. 96 (2003) 61–98.
[67] H. Johansen, P. Colella, A cartesian grid embedded boundary method for poisson's equation on irregular domains, J. Comput. Phys. 147 (1998) 60–85.

[68] P. Schwartz, M. Barad, P. Colella, T. Ligocki, A cartesian grid embedded boundary method for the heat equation and Poisson's equation in three dimensions, J. Comput. Phys. 211 (2006) 531–550.
[69] M. Oevermann, R. Klein, A Cartesian grid finite volume method for elliptic equations with variable coefficients and embedded interfaces, J. Comput. Phys. 219 (2) (2006) 749–769.
[70] M. Oevermann, C. Scharfenberg, R. Klein, A sharp interface finite volume method for elliptic equations on Cartesian grids, J. Comput. Phys. 228 (14) (2009) 5184–5206.
[71] M. Benzi, G. Golub, J. Liesen, Numerical solution of saddle point problems, Acta Numer. (2005) 1–137.
[72] S. Balay, K. Buschelman, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, H. Zhang, 2009. PETSc Web page, <http://www.mcs.anl.gov/petsc>.
[73] S. Balay, K. Buschelman, V. Eijkhout, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, H. Zhang, PETSc users manual, Tech. Rep. ANL-95/11 – Revision 3.0.0, Argonne National Laboratory, 2008.
[74] S. Balay, W.D. Gropp, L.C. McInnes, B.F. Smith, Efficient management of parallelism in object oriented numerical software libraries, in: E. Arge, A.M. Bruaset, H.P. Langtangen (Eds.), Modern Software Tools in Scientific Computing, Birkhäuser Press, 1997, pp. 163–202.
[75] T.A. Davis, Suitesparseqr: Algorithm 8xx: Suitesparseqr, a multifrontal multithreaded sparse qr factorization package, ACM TOMS, submitted for publication.