A High Order Cartesian Grid, Finite Volume Method for Elliptic Interface Problems

Will Thacher[a,b,*], Hans Johansen[b], Daniel Martin[b]

[a] *Graduate Student Researcher, Applied Science and Technology Group, University of California Berkeley, Berkeley, CA, 94720, United States*
[b] *Applied Numerical Algorithms Group, Lawrence Berkeley National Laboratory, Berkeley, CA, 94720, United States*

Abstract

We present a higher-order finite volume method for solving elliptic PDEs with jump conditions on interfaces embedded in a 2D Cartesian grid. Second, fourth, and sixth order accuracy is demonstrated on a variety of tests including problems with high-contrast and spatially varying coefficients, large discontinuities in the source term, and complex interface geometries. We include a generalized truncation error analysis based on cell-centered Taylor series expansions, which then define stencils in terms of local discrete solution data and geometric information. In the process, we develop a simple method based on Green's theorem for computing exact geometric moments directly from an implicit function definition of the embedded interface. This approach produces stencils with a simple bilinear representation, where spatially-varying coefficients and jump conditions can be easily included and finite volume conservation can be enforced.

Keywords: Elliptic Interface Problem, High Order, Embedded Boundary, Cut Cell, Finite Volume, Discontinuous Coefficients

1. Introduction

Elliptic PDEs with discontinuities in the source term, coefficients and solution form an important class of equations in computational science and engineering. These equations arise from mathematical models of multi-material systems, multi-phase flows, crystal growth, and many other physical processes [1]. Solving such equations numerically is not straightforward because the accuracy of the scheme is typically based on smoothness assumptions that do not in general apply at the interface.

Numerous schemes have been proposed to solve this problem based on finite difference, finite volume, and finite element formulations. These methods can roughly be classified into those that treat the interface explicitly by creating elements that conform to the shape of the interface, or those that represent the interface implicitly by "embedding" it onto a non-conforming mesh (see [2] for a thorough review and further references). In the finite element realm, methods such as [3] body-fit the mesh to the interface whereas

*Corresponding author: wthacher@lbl.gov

methods such as [4] use a fixed mesh and modify basis functions where the interface crosses elements. A widely used and influential method in the finite difference category is the Immersed Interface Method (IIM) [5]. The IIM uses standard Cartesian grid finite difference stencils away from the interface and modifies stencils near the interface using one-sided Taylor series expansions that incorporate jump conditions. The Ghost Fluid Method [6] extrapolates the solution across the interface to nearby grid points by incorporating jump conditions so that standard stencils can still be used at all grid points. These various finite difference methods are closely related to various schemes for imposing boundary conditions; jump conditions can be thought of as a sort of implicit boundary condition that depends on the solution itself.

This paper is concerned with the third category: finite volume schemes. These methods are conservative (in the sense that the divergence theorem is applied to a control volume), and have well-studied stability properties [7]. The embedded boundary (EB) method of [8] combines the implicit and explicit interface representations: the interface is embedded onto a Cartesian grid, forming cut-cell volumes of arbitrary shapes where it intersects rectangular cells. The elliptic equation is then discretized in flux divergence form using techniques developed in [7] and [9] with appropriate modifications made at the interface to enforce jump conditions. The method developed in [8] is second-order accurate L^∞ and L^1 norm, but is difficult to extend to higher order accuracy. For low-order methods, cell averages can be treated as point values to second-order accuracy, so finite difference type schemes can be employed to create stencils. This is not the case for higher order finite volume methods; integration must be performed over arbitrarily-shaped "cut cells." Techniques such as choosing a midpoint or centroid as a quadrature rule for surface integrals of the flux is not sufficient for higher-order accuracy.

Many of these difficulties are being addressed by recent developments in higher-order finite volume and EB methods, which are summarized in [10]. One example is the use of weighted least-squares interpolation for stencil construction in complex geometries ([11], [12]) as well as the derivation of high-order stencils on Cartesian grids [13]. Given the close relationship between boundary conditions and jump conditions, we propose extending the methodology of [12] to the elliptic interface problem. The primary contribution of this research is a finite volume method for the variable coefficient 2D elliptic interface problem that is 1) high order accurate and 2) conservative. In the process, we have also created approaches for 1) an efficient technique for generating exact geometric information from an implicit function and 2) a method for building high-order finite volume stencils for variable coefficient elliptic operators on arbitrary cut-cell meshes.

The outline of the paper is as follows: In Section 2, we define mesh and geometric quantities and give a general truncation error analysis which allows us to design stencils of arbitrarily high order. In Section 3, we describe in detail our method for constructing stencils. In Section 4, we present results that validate the approach using a series of model problems that test different aspects of the scheme.

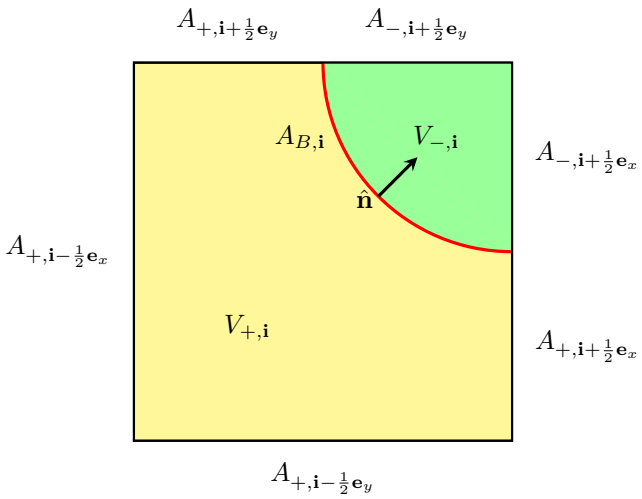


Figure 1: Cut cell geometric quantities that make up the finite volume notation.

2. Discretization

Let Ω be a physical domain that is divided into the subdomains Ω^+ and Ω^- by an interface Γ. We consider the variable coefficient elliptic interface problem for $u(x)$:

$$\alpha u - \nabla \cdot (\beta \nabla u) = f \text{ on } \Omega \tag{1}$$

$$[u] = w \text{ on } \Gamma \tag{2}$$

$$[\beta \partial_{\mathbf{n}} u] = v \text{ on } \Gamma\,. \tag{3}$$

Here, $[\cdot]$ denotes a jump in some quantity at the interface: $[u] = u^+(\mathbf{x}) - u^-(\mathbf{x})$ at some point $\mathbf{x} \in \Gamma$, and the term $\partial_{\mathbf{n}} u \equiv \nabla u \cdot \hat{\mathbf{n}}$ represents a *flux* at this boundary with unit normal $\hat{\mathbf{n}}(\mathbf{x})$. Finally, coefficients $\alpha^{\pm}(\mathbf{x}), \beta^{\pm}(\mathbf{x})$ and the source term $f^{\pm}(\mathbf{x})$ vary in space and may be discontinuous across Γ.

The domain Ω is discretized into a Cartesian mesh of square control volumes (or "cells") $V_{p,\mathbf{i}}$, $\mathbf{i} \in \mathbb{Z}^2$, that have centroids $\mathbf{x}_{p,\mathbf{i}}$ and side lengths of scale h, the grid spacing (see Figure 1). We indicate $p \in \{+,-\}$ to specify a subdomain of Ω_p; p can often be thought of as the phase or material type of a physical quantity. We assume that each cell $V_{p,\mathbf{i}}$ may have up to four grid-aligned faces, which we label $A_{p,\mathbf{i}\pm\frac{1}{2}\mathbf{e}_d}$, where $\mathbf{e}_d$ is the unit vector in direction d.

Any cell that is intersected by Γ, the "embedded boundary" (EB), is called a "cut" cell. We make the following assumptions to simplify the geometric considerations. First, a cut cell consists of only two control volumes $V_{+,\mathbf{i}}$ and $V_{-,\mathbf{i}}$ divided by a portion of the EB, denoted by $A_{B,\mathbf{i}}$. The unit normal vector $\hat{\mathbf{n}}$ on Γ points from Ω^+ to Ω^-. So, along with $A_{p,\mathbf{i}\pm\frac{1}{2}\mathbf{e}_d}$ as the grid-aligned faces of each portion of the cut cell, each cut cell must have a total of at least 3, and at most 5, faces.

Because we are using a finite volume formulation, we should define additional geometric quantities that

will be useful throughout this paper: a geometric "moment" is an integral of a centered monomial over some specified region. We define four moments corresponding to four components of the geometry:

$$m^{\mathbf{q}}_{p,\mathbf{i}} = \int_{V_{p,\mathbf{i}}} (\mathbf{x} - \bar{\mathbf{x}})^{\mathbf{q}} \, dV \tag{4}$$

$$m^{\mathbf{q}}_{p,\mathbf{i}\pm\frac{1}{2}e_d} = \int_{A_{p,\mathbf{i}\pm\frac{1}{2}e_d}} (\mathbf{x} - \bar{\mathbf{x}})^{\mathbf{q}} \, dA \tag{5}$$

$$m^{\mathbf{q}}_{B,\mathbf{i}} = \int_{A_{B,\mathbf{i}}} (\mathbf{x} - \bar{\mathbf{x}})^{\mathbf{q}} \, dA \tag{6}$$

$$m^{\mathbf{q}}_{B,\mathbf{i},d} = \int_{A_{B,\mathbf{i},d}} (\mathbf{x} - \bar{\mathbf{x}})^{\mathbf{q}} \, \hat{n}_d \, dA \ , \tag{7}$$

where $\mathbf{q} = [q_x \ q_y]$ is a vector of non-negative integers, and we use the multi-index notation $(\mathbf{x} - \bar{\mathbf{x}})^{\mathbf{q}} = (x - \bar{x})^{q_x}(y - \bar{y})^{q_y}$. The multi-indices have sum of at most P, $|\mathbf{q}| \leq P$, and are ordered lexicographically: $\{00, 10, 20, ...P0, 01, 11, ...0P\}$. This allows us to refer to $\mathbf{v}[\mathbf{q}]$ as the $\mathbf{q}^{th}$ entry of a vector $\mathbf{v}$. Thus by definition, the volume of cell $\mathbf{i}$ is $|V_{p,\mathbf{i}}| = m^{00}_{p,\mathbf{i}}$, and the centroid $\bar{\mathbf{x}}_{p,\mathbf{i}}$ of $V_{p,\mathbf{i}}$ is $\frac{1}{|V_{p,\mathbf{i}}|}\left[m^{10}_{p,\mathbf{i}}, m^{01}_{p,\mathbf{i}}\right]$. Similarly, $m^{00}_{B,\mathbf{i}}$ is the area of the EB, and $m^{00}_{B,\mathbf{i},x}$ is its x normal component-weighted area, or x direction *cross-section*. For ease of notation, throughout this paper we ignore $\bar{\mathbf{x}}$, although in practice it is the cell-center of each full Cartesian cell.

The interface is represented as the level set of an implicit function, as is common in cut cell literature ([14], [15]). We generate geometric information by creating successively refined piecewise linear approximations to the interface and computing geometric moments over those approximations. The convergence of this sequence of moment approximations is accelerated using Richardson extrapolation, and the resulting geometric moments are exact to within roundoff errors. The geometry algorithm is described in detail in Appendix A. We note that it is not strictly necessary to use an implicit function description of the interface; our method only requires that geometric moments can be computed to sufficient accuracy. However, the implicit function description has several desirable properties: for any point in the domain we can easily determine on which side of the interface that point is on, while points on the interface can be found using a root finder, and the approach is compatible with level set methods for moving boundary problems.

Two types of variables are stored on the mesh: cell-averaged quantities $\langle u \rangle_{p,\mathbf{i}} = \frac{1}{|V_{p,\mathbf{i}}|}\int_{V_{p,\mathbf{i}}} u \, dV$, and centroid-centered quantities $u_{p,\mathbf{i}} = u(\mathbf{x}_{p,\mathbf{i}})$. The coefficients α and β are given as point values at the centroids of cells. The right-hand-side function is provided as cell-averaged values, $\langle f \rangle$, of sufficient accuracy, and we solve for cell-averaged values $\langle u \rangle$.

Within this context, our finite volume scheme solves the discrete system:

$$\langle \alpha u \rangle_{p,\mathbf{i}} - \langle \nabla \cdot \beta \nabla u \rangle_{p,\mathbf{i}} = \langle f \rangle_{p,\mathbf{i}} \tag{8}$$

$$[u]_{\mathbf{i}} = w_{\mathbf{i}} \tag{9}$$

$$[\beta \partial_{\mathbf{n}} u]_{\mathbf{i}} = v_{\mathbf{i}} \ . \tag{10}$$

for $\langle u \rangle_{p,\mathbf{i}}$ in each volume $V_{p,\mathbf{i}}$ in the mesh, subject to problem-specific boundary conditions. The system of equations we solve will have one degree of freedom in full cells and two degrees of freedom in cut cells. $[\cdot]_{\mathbf{i}}$ denotes the integral of the jump of a quantity across the EB in cut cell $\mathbf{i}$:

$$[u]_{\mathbf{i}} = \int_{A_{B,\mathbf{i}}} u^+ - u^- dA \ . \tag{11}$$

The objective of the following section is to provide a general truncation error analysis that will allow us to discretize (8) – (10) to high order accuracy.

2.1. Error Analysis

In the finite volume or finite difference context, a *stencil* approximates some functional $G(u)$ by a linear combination of local information about the function u. This functional is typically point values or integrals over some region of derivative of u. The function information, or data, can include boundary conditions, jump conditions, point values of the function, or averages of the function over some nearby region. Let $\mathbf{d}$ denote this vector of local function data, and $\mathbf{s}$ be the vector of stencil values corresponding to each of the data points in $\mathbf{d}$. The truncation error τ of this stencil is defined as:

$$\tau = \mathbf{s}^T \mathbf{d} - G(u) \ . \tag{12}$$

For the present problem, G will be an integral of u, or some combination of its partial derivatives, over a one or two dimensional region. If we can approximate u using a truncated Taylor series, $G(u)$ can be approximated with a linear combination of Taylor series coefficients of u up to the desired order of accuracy.

Throughout this section we drop the subscripts p and $\mathbf{i}$ except where necessary for clarity. We can express u locally as a Taylor series expansion and remainder term:

$$u(\mathbf{x}) = \sum_{|\mathbf{q}| \leq P} \frac{1}{\mathbf{q}!} u^{(\mathbf{q})}(\bar{\mathbf{x}}) \mathbf{x}^{\mathbf{q}} + O\left(h^{P+1}\right), \tag{13}$$

where, again using multi-index notation, $\mathbf{x}^{\mathbf{q}} = x^{q_x} y^{q_y}$, $\mathbf{q}! = q_x! q_y!$, and $u^{(\mathbf{q})} = \frac{\partial^{q_x} \partial^{q_y}}{\partial x^{q_x} \partial y^{q_y}} u$. The Taylor polynomial is then just

$$u(\mathbf{x}) = \sum_{|\mathbf{q}| \leq P} c_u^{\mathbf{q}} \mathbf{x}^{\mathbf{q}} + O\left(h^{P+1}\right), \text{ where} \tag{14}$$

$$c_u^{\mathbf{q}} = \frac{1}{\mathbf{q}!} u^{(\mathbf{q})}(\bar{\mathbf{x}}) \, .$$

Integrating the flux divergence term in (8) over the discrete volume V and applying Gauss' theorem we obtain:

$$\int_V \nabla \cdot \beta \nabla u \; dV = \left[\sum_{\pm,\; d} (\pm 1) \int_{A_{\pm \frac{1}{2} e_d}} \beta \frac{\partial u}{\partial x_d} \; dA \right] + \int_{A_B} \beta \nabla u \cdot \hat{\mathbf{n}} \; dA \; . \tag{15}$$

Expressing β and u as Taylor expansions, for a surface integral over any (EB or grid-aligned) face A we have:

$$\int_A \beta \nabla u \cdot \hat{\mathbf{n}} \; dA = \int_A \left(\sum_{|\mathbf{r}| \le P} c_\beta^{\mathbf{r}} \mathbf{x}^{\mathbf{r}} \right) \left(\sum_{|\mathbf{q}| \le P} c_u^{\mathbf{q}} \left[\frac{\partial \mathbf{x}^{\mathbf{q}}}{\partial x} \hat{n}_x + \frac{\partial \mathbf{x}^{\mathbf{q}}}{\partial y} \hat{n}_y \right] \right) + O(h^P) \; dA \tag{16}$$

$$= \int_A \left(\sum_{|\mathbf{r}| \le P} c_\beta^{\mathbf{r}} \mathbf{x}^{\mathbf{r}} \right) \left(\sum_{|\mathbf{q}| \le P} c_u^{\mathbf{q}} \left[q_x \mathbf{x}^{\mathbf{q}-\mathbf{e}_x} \hat{n}_x + q_y \mathbf{x}^{\mathbf{q}-\mathbf{e}_y} \hat{n}_y \right] \right) + O(h^P) \; dA \tag{17}$$

$$= \sum_{|\mathbf{q}| \le P} \left[\sum_{|\mathbf{r}| \le (P - |\mathbf{q}|)} c_\beta^{\mathbf{r}} \left(q_x m_{A,x}^{\mathbf{q}+\mathbf{r}-\mathbf{e}_x} + q_y m_{A,y}^{\mathbf{q}+\mathbf{r}-\mathbf{e}_y} \right) \right] c_u^{\mathbf{q}} + O(h^{P+1}) \; . \tag{18}$$

In order to compute $\langle \nabla \cdot \beta \nabla u \rangle$, we apply (18) to each surface integral in (15) and divide by the cell volume. Since $|V|$ is an $O(h^2)$ quantity, the error term for the cell-averaged flux divergence term is $O(h^{P-1})$.

Similarly, for the linear term in (8), we have:

$$\langle \alpha u \rangle = \frac{1}{|V|} \int_V \sum_{|\mathbf{r}| \le P} c_\alpha^{\mathbf{r}} \mathbf{x}^{\mathbf{r}} \sum_{|\mathbf{q}| \le P} c_u^{\mathbf{q}} \mathbf{x}^{\mathbf{q}} + O(h^{P+1}) dV \tag{19}$$

$$= \frac{1}{|V|} \sum_{|\mathbf{q}| \le P} \left[\sum_{|\mathbf{r}| \le (P - |\mathbf{q}| - 2)} c_\alpha^{\mathbf{r}} m^{\mathbf{r}+\mathbf{q}} \right] c_u^{\mathbf{q}} + O(h^{P+1}) \; . \tag{20}$$

We see that our approximation to the functionals G of interest can be written in the general form:

$$G(u) = \sum_{|\mathbf{q}| \le P} g^{\mathbf{q}} c_u^{\mathbf{q}} + O(h^R) = \mathbf{g}^T \mathbf{c}_u + O(h^R) \; , \tag{21}$$

where $\mathbf{c}_u$ is the vector of approximate Taylor series coefficients for u, and $\mathbf{g}$ is the vector of terms that result from operating on these coefficients, shown in brackets in (18) and (20). The order R error term for the functional approximation may be different than the order $P+1$ error term for the Taylor series of u as a result of differentiation and integration.

To calculate the truncation error τ in (12), we must fill in the vector $\mathbf{d}$ with function data. Suppose, for example, that the function data are cell-averaged values of u. Then we can write:

$$\langle u \rangle_{\mathbf{j}} = \frac{1}{|V_{\mathbf{j}}|} \int_{V_{\mathbf{j}}} \sum_{|\mathbf{q}| \le P} c_u^{\mathbf{q}} \mathbf{x}^{\mathbf{q}} + O\left(h^{P+1}\right) dV \tag{22}$$

$$= \sum_{|\mathbf{q}| \le P} \frac{m_{\mathbf{j}}^{\mathbf{q}}}{|V_{\mathbf{j}}|} c_u^{\mathbf{q}} + O\left(h^{P+1}\right) = \mathbf{m}_{\mathbf{j}}^T \mathbf{c}_u + O\left(h^{P+1}\right) \; , \tag{23}$$

where $\mathbf{m_j}$ is the vector of cell-averaged volume moments for any cell $\mathbf{j}$. If we have n such cell-averaged values, then we can write:

$$\mathbf{d} = \begin{bmatrix} \mathbf{m}_{\mathbf{j}_1}^T \\ \mathbf{m}_{\mathbf{j}_2}^T \\ \cdots \\ \mathbf{m}_{\mathbf{j}_n}^T \end{bmatrix} \mathbf{c}_u + O(h^{P+1}) = \mathbf{M}\mathbf{c}_u + O(h^{P+1}) \ , \tag{24}$$

where $\mathbf{M}$ is the "moment matrix" whose rows are the vectors $\mathbf{m}_{\mathbf{j}_i}^T$. Inserting this into (12) we have:

$$\tau = \mathbf{s}^T \mathbf{M}\mathbf{c}_u + O(h^{P+1}) - \left(\mathbf{g}^T \mathbf{c}_u + O(h^R)\right) \ . \tag{25}$$

To ensure that τ is $O(h^{\min(R,P+1)})$, we must have:

$$\mathbf{s}^T \mathbf{M}\mathbf{c}_u = \mathbf{g}^T \mathbf{c}_u \, . \tag{26}$$

This must hold for any u with a Taylor series represented by arbitrary $\mathbf{c}_u$, so that

$$\mathbf{M}^T \mathbf{s} = \mathbf{g} \, . \tag{27}$$

Note that this linear system is a general form for the *method of undetermined coefficients*, as explained in [16] (Chapter 1.2): stencil weights are chosen so that the sum of Taylor series terms of the stencil exactly matches the Taylor series terms of the functional up to some order. In this case, we are working with cell averages rather than pointwise evaluations, but the principle is the same. If the neighborhood of local function data is chosen such that this linear system is exactly determined or undetermined, the least norm solution to this system is given by the pseudoinverse of $\mathbf{M}$:

$$\mathbf{s} = (\mathbf{M}^T)^\dagger \mathbf{g} \, . \tag{28}$$

If this system is undetermined, there are infinitely many stencils that will have the same order of truncation error. However, there is another profitable way to view this stencil construction process that justifies using the least-norm stencil. Suppose we want to interpolate the data stored in the vector $\mathbf{d}$ with a degree P polynomial. If the matrix $\mathbf{M}$ is full rank, we can determine the coefficients of this polynomial by solving, in a least-squares sense, the linear system in (24):

$$\mathbf{c}_u = \mathbf{M}^\dagger \mathbf{d} \ . \tag{29}$$

Inserting this into (21), we have:

$$G(u) = \mathbf{g}^T \mathbf{M}^\dagger \mathbf{d} + O\left(h^{\min(R,P+1)}\right) \ . \tag{30}$$

Finally, inserting this into the truncation error expression (12):

$$\mathbf{s}^T\mathbf{d} = \mathbf{g}^T\mathbf{M}^\dagger\mathbf{d} \implies \tag{31}$$

$$\mathbf{s} = (\mathbf{M}^T)^\dagger\mathbf{g} \ , \tag{32}$$

where the implication follows from the fact this must hold for any $\mathbf{d}$.

Thus we can view our stencils as originating from either the undetermined system $\mathbf{M}^T\mathbf{s} = \mathbf{g}$, in which $\mathbf{s}$ is chosen to cancel lower order Taylor series terms, or the overdetermined system $\mathbf{M}\mathbf{c}_u = \mathbf{d}$, in which we fit an interpolating polynomial to local data. This error analysis is quite general; we have made no mention of the shape of the volumes in the mesh, only that we know their geometric moments to sufficient accuracy. The interface jump conditions and boundary conditions are considered to be pieces of function data that can be used to build stencils near an interface or boundary. Taking the overdetermined perspective, by doing so we will constrain our interpolating polynomials, and therefore the numerical solution, to match boundary and jump conditions.

3. Stencil Construction

In the previous section we showed that if we approximate the solution u with a polynomial whose coefficients are mapped from local function data by $\mathbf{M}^\dagger$, then our stencil will take the simple form (28). In this section we will describe in detail our method for computing the moment matrix $\mathbf{M}$ and the vector $\mathbf{g}$ of terms that result from operating on the Taylor polynomials approximating u. From (18) and (20), we see that to achieve a truncation error of order $P-1$ for the two terms in (8), we need to "calculate" Taylor coefficients $c_\phi^{\mathbf{q}}$ for $\phi \in \{u, \beta, \alpha\}$ up to order P. In this section we again drop the subscript p except where necessary for clarity.

Both the linear term and the flux divergence term can be expressed as linear combinations of the Taylor series terms $c_u^{\mathbf{q}}$, as seen in (20) and (18). For the linear term, let $\mathbf{g}_{\alpha,\mathbf{i}}$ be the vector whose $\mathbf{q}^{th}$ entry is

$$\mathbf{g}_{\alpha,\mathbf{i}}\left[\mathbf{q}\right] = \sum_{|\mathbf{r}|\leq(P-|\mathbf{q}|-2)} c_\alpha^{\mathbf{r}} m_{\mathbf{i}}^{\mathbf{r}+\mathbf{q}} \ . \tag{33}$$

Applying (32), we have that a stencil for the linear term is given by:

$$\mathbf{s}_{\alpha,\mathbf{i}} = (\mathbf{M}_{u,\mathbf{i}}^T)^\dagger \mathbf{g}_{\alpha,\mathbf{i}} \tag{34}$$

where the subscript $\alpha, \mathbf{i}$ indicates a stencil for the linear term, $\langle \alpha u \rangle$, over cell $\mathbf{i}$. For the integral of $\beta \nabla u \cdot \hat{\mathbf{n}}$ over a face A, we let $\mathbf{g}_{\beta,A}$ be the vector whose $\mathbf{q}^{th}$ entry is

$$\mathbf{g}_{\beta,A}\left[\mathbf{q}\right] = \sum_{|\mathbf{r}|\leq(P-|\mathbf{q}|)} c_\beta^{\mathbf{r}} \left(q_x m_{A,x}^{\mathbf{q}+\mathbf{r}-\mathbf{e}_x} + q_y m_{A,y}^{\mathbf{q}+\mathbf{r}-\mathbf{e}_y} \right) \ . \tag{35}$$

A stencil for the flux surface integral (16) is then given by:

$$\mathbf{s}_{\beta,A} = (\mathbf{M}^T_{u,\mathbf{i}})^\dagger \mathbf{g}_{\beta,A} \ . \tag{36}$$

Each entry in $\mathbf{g}_{\alpha,\mathbf{i}}$ involves the Taylor coefficients of the α field, and likewise each $\mathbf{g}_{\beta,A}$ entry of involves the Taylor coefficients of the β field. We must generate these coefficients for each volume V, which can be done with the same moment matrix formulation that we use to compute the Taylor coefficients $\mathbf{c}_u$.

Recall that α is given as point values at the centroids of cells. Let $\mathcal{N}_{p,\mathbf{i}}$ be some neighborhood of cells $\mathbf{j}_k \in \{\mathbf{j}_1, \ldots, \mathbf{j}_n\}$ in phase p around volume $V_{p,\mathbf{i}}$. A cell $V_{p,\mathbf{j}}$ in this neighborhood has centroid $\mathbf{x}_{p,\mathbf{j}}$, with $\alpha_{p,\mathbf{j}} = \alpha(\mathbf{x}_{p,\mathbf{j}})$. Our function data will consist of $\alpha_{p,\mathbf{j}}$ at each point $\mathbf{j}_k$ in the neighborhood, which we compile into the vector $\mathbf{d}_{\alpha,p,\mathbf{i}}$. Let $\mathbf{M}_{\alpha,p,\mathbf{i}}$ be the variable coefficient interpolation matrix whose rows consist of monomials in the Taylor expansion of α evaluated at $\mathbf{x}_{p,\mathbf{j}}$:

$$\mathbf{M}_{\alpha,p,\mathbf{i}}\,[k, \mathbf{q}] = \mathbf{x}^{\mathbf{q}}_{p,\mathbf{j}_k} \ . \tag{37}$$

Letting $\mathbf{c}_{\alpha,p}$ be the vector of Taylor series coefficients, we have

$$\mathbf{c}_{\alpha,p} = \mathbf{M}^\dagger_{\alpha,p,\mathbf{i}} \mathbf{d}_{\alpha,p,\mathbf{i}} \ . \tag{38}$$

We can do the same for β to calculate $\mathbf{c}_{\beta,p}$.

Therefore we have that:

$$\mathbf{g}_{\alpha,\mathbf{i}} = \mathbf{G}_{\alpha,\mathbf{i}} \mathbf{M}^\dagger_{\alpha,\mathbf{i}} \mathbf{d}_{\alpha,\mathbf{i}} \ , \tag{39}$$

where $\mathbf{G}_{\alpha,\mathbf{i}}$ is the matrix whose $\mathbf{q}, \mathbf{r}$ entry is $m_{\mathbf{i}}^{\mathbf{r}+\mathbf{q}}$. Combining this with (34), we can finally write:

$$\mathbf{s}^T_{\alpha,\mathbf{i}} = \mathbf{d}^T_{\alpha,\mathbf{i}} (\mathbf{M}^\dagger_{\alpha,\mathbf{i}})^T \mathbf{G}^T_{\alpha,\mathbf{i}} \mathbf{M}^\dagger_{u,\mathbf{i}} \ . \tag{40}$$

This stencil is bilinear in α and u, which is appropriate for the bilinear functional we are trying to approximate. We can obtain a similar formulation for the flux stencil $\mathbf{s}^T_{\beta,A}$ for each face A of the cell. Thus constructing stencils is just a matter of constructing the moment matrices for u, β and α. We now describe how to construct moment matrices at, near, and away from the interface.

3.1. Moment Matrices

We partition our cells into three subsets: cut cells Ω_C, irregular cells Ω_I, and regular cells Ω_R. Cut cells are intersected by the EB. Irregular cells are not intersected by the EB, but at least one cell in the stencil footprint for a regular cell is intersected by the EB. See Figure 2. The method for constructing the moment matrices is different for each of these three types of cells.

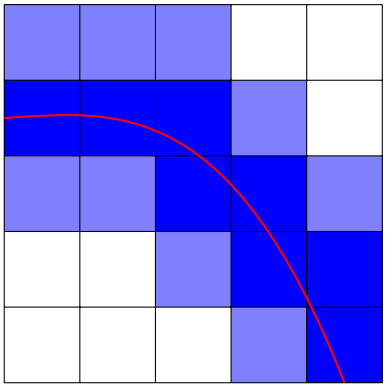
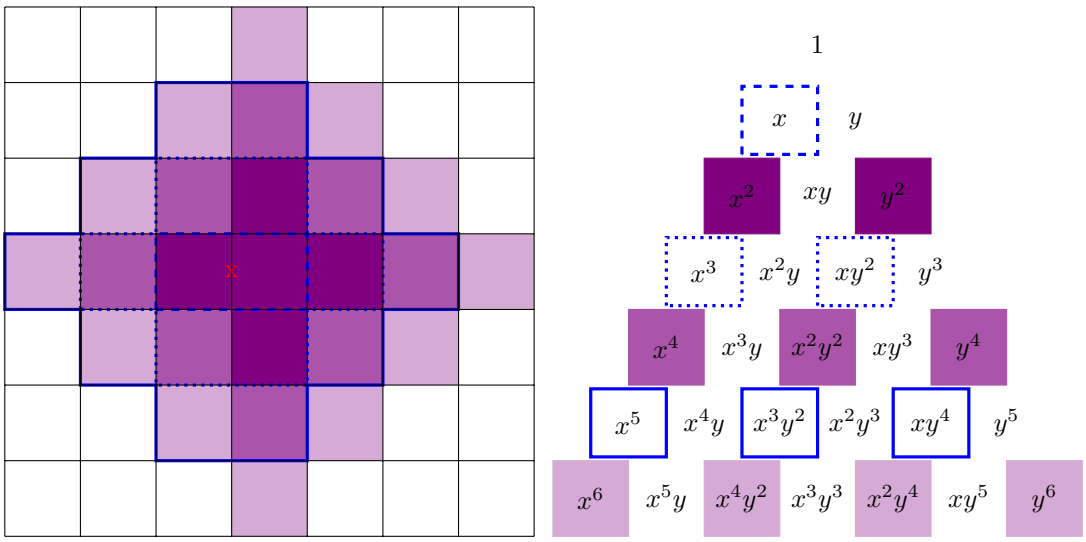


Figure 2: For the $P = 2$ scheme, the regular cell footprint is a standard five-point Laplacian, and if any point in the footprint is a cut cell, it is then "irregular." Cut cells are shown with dark shading, irregular cells with light shading, and the remaining white cells are "regular."



(a) (b)

Figure 3: Figure (a) shows increasing footprints for the moment matrix for $P = 2, 4, 6$ schemes with increasingly lighter shades. The footprints "support" the corresponding (same shading) even monomial terms in Pascal's triangle, (b). As explained in section 3.1.1, odd moments of order P are not needed to achieve order P truncation error. In addition, we require at most order $P - 1$ monomial terms for the coefficient β. The cell face $A_{\mathbf{i}-\mathbf{e}_x}$ is marked with a red x, and the the dashed, dotted, and solid blue outlines (for $P = 2, 4, 6$, respectively), show the footprint for the modified moment matrix $\mathbf{M}_{\beta, A_{\mathbf{i}-\mathbf{e}_x}}$ in (42). The corresponding dashed, dotted, and solid blue boxed monomials in (b) show the order $P - 1$ terms that are supported by these footprints.

3.1.1. Regular Cells

The vast majority of cells will be regular and will all have the same bilinear stencil, meaning we only have to solve for this stencil once. Since regular cells are squares, the integral of any monomial error term with odd degree over a regular cell is 0. This means that for the the flux divergence term, in regular cells we can achieve a truncation error of order P using an order P polynomial. We can achieve an order P cell averaged linear term with an order $P-2$ polynomial.

Furthermore, we do not need to calculate all of the Taylor polynomial coefficients up to a given order: if P is even, then the highest order Taylor series coefficients that we need are those $c^{\mathbf{q}}$ such that $|\mathbf{q}| = P$ and q_x, q_y are both even. This follows from the fact that our operator does not involve any mixed derivatives: if $|\mathbf{q}| = P$ for P even, then if q_x is odd, q_y must be as well, so taking derivatives in only one of the dimensions will leave at least one of these powers odd. Although we will get even cross terms from the flux divergence, the even cross terms that arise from combining derivatives of odd order P moments with lower order moments are on the order P and are therefore not needed. These cancellations are typical for centered finite differences, but here they arrive through symmetries in moments and polynomial coefficients. As shown in Figure 3, these coefficients can be supported with a stencil footprint consisting of cells whose centroids are a Manhattan distance of $\frac{P}{2}h$ from the center cell. The columns of the moment matrix $\mathbf{M}_{u,\mathbf{i}}$ correspond to all monomials with either $|\mathbf{q}| < P$ or $|\mathbf{q}| = P$ and q_x, q_y are both even. Each row is simply the cell-averaged moments $\mathbf{m}_{\mathbf{j}}^T$ for each cell $\mathbf{j}$ in the stencil, and the resulting matrix is square.

For the matrices $\mathbf{M}_{\alpha,\mathbf{i}}$ and $\mathbf{M}_{\beta,\mathbf{i}}$ we use the same footprint as $\mathbf{M}_{u,\mathbf{i}}$. Construction of $\mathbf{s}_{\alpha,\mathbf{i}}$ is then straightforward. The flux divergence term is slightly more complicated. For each of the four faces of the cell $A_{\mathbf{i}\pm\mathbf{e}_d}$, if we can use the same flux stencil on each cell's face, then the stencil will guarantee conservation, that is that its contribution to one cell will be the negative of its contribution to its neighbor sharing the same face. For example, consider the face $A_{\mathbf{i}-\mathbf{e}_x}$, the left hand vertical face of the cell. For the unit normal, we have $\hat{n}_x = -1$ and $\hat{n}_y = 0$, so (18) reduces to the simpler form:

$$\int_{A_{\mathbf{i}-\mathbf{e}_x}} \beta \nabla u \cdot \hat{\mathbf{n}} \; dA \;=\; \sum_{|\mathbf{q}|\le P} \left(\sum_{|\mathbf{r}|\le (P-|\mathbf{q}|)} c_\beta^{\mathbf{r}} \left(-\int_{-\frac{h}{2}}^{\frac{h}{2}} q_x \left[-\frac{h}{2}, y \right]^{\mathbf{q}+\mathbf{r}-\mathbf{e}_x} dy \right) \right) c_u^{\mathbf{q}} \;+\; O(h^{P+1}) \quad , \quad (41)$$

meaning the integral in parentheses is only non-zero if $q_x > 0$ and $q_y + r_y$ is even. In addition, for accuracy requirements we only need moments with $|\mathbf{r} + \mathbf{q} - 1| < P$. This means we do not need to calculate any coefficients $\mathbf{c}_\beta^{\mathbf{r}}$ such that $|\mathbf{r}| = P$; in other words, an order $P-1$ Taylor approximation to β will suffice. Formally, we can multiply the matrix $\mathbf{M}_{\beta,\mathbf{i}}$ on the left and right by matrices that zero out the proper columns and rows, giving us the modified moment matrix:

$$\mathbf{M}_{\beta,A_{\mathbf{i}-\mathbf{e}_x}} = \mathbf{P}_L \mathbf{M}_{\beta,\mathbf{i}} \mathbf{P}_R \; , \quad (42)$$

where $\mathbf{P}_R$ eliminates all columns corresponding to moments with order greater than $P-1$, as well as columns corresponding to moments of order $P-1$ such that q_y is odd. $\mathbf{P}_L$ eliminates rows that are not necessary

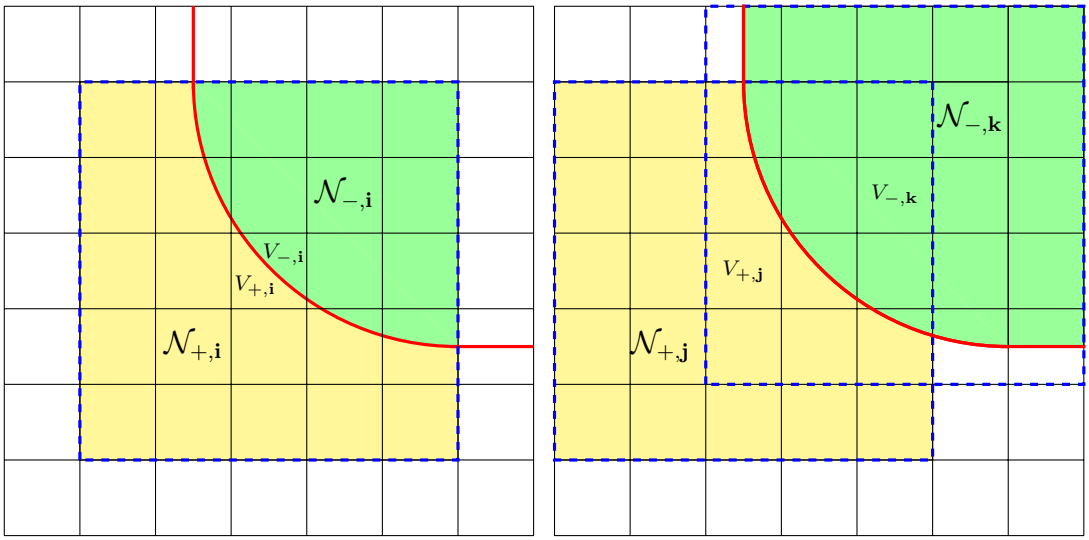


Figure 4: Figure (a) shows neighborhoods $\mathcal{N}_{\pm,\mathbf{i}}$ used to construct interpolation matrices for order $P = 2$, around cut cell $\mathbf{i}$, which contains two volumes $V_{\pm,\mathbf{i}}$ bordering the interface. Figure (b) shows neighborhoods surrounding full cells $\mathbf{j}$ and $\mathbf{k}$ that don't contain the interface, but are "irregular," meaning the regular stencil for order $P = 2$ would be inconsistent (Oth-order).

to support these moments. We can likewise adjust $\mathbf{d}_{\beta,\mathbf{i}}$ and $\mathbf{G}_{\beta,\mathbf{i}}$ to account for these modifications. The contributions from $c_u^{\mathbf{q}}$ are already symmetric about the face because they all represent centered differences. This process results in a flux stencil footprint consisting of all cells whose centroids are a Manhattan distance of $\frac{P-1}{2}h$ from the centroid of the face $A_{\mathbf{i}-\mathbf{e}_x}$ (see Figure 3). However, these simplifications to achieve the minimal stencil footprint rely on symmetry arguments, so they do not apply generally to irregular and cut cells.

3.1.2. Irregular Cells

Let cell $\mathbf{i}$ be an irregular cell in phase p, meaning its regular cell footprint contains at least one cell which is intersected by the EB. This invalidates the truncation error analysis for the regular cell stencil, so we adopt a more general method for construction of moment matrices for irregular cells. Let $\mathcal{N}_{p,\mathbf{i}}$ be a neighborhood of cells in phase p around cell $\mathbf{i}$ (see Figure 4(b)). Our data vector $\mathbf{d}_{u,p,\mathbf{i}}$ will consist of cell averaged values $\langle u \rangle_{\mathbf{j}}$ for each cell $\mathbf{j} \in \mathcal{N}_{p,\mathbf{i}}$. Each row of the corresponding moment matrix $\mathbf{M}_{u,p,\mathbf{i}}$ will simply be $\mathbf{m}_{\mathbf{j}}^T$, the row of cell averaged volume moments for each cell $\mathbf{j}$:

$$\mathbf{M}_{u,p,\mathbf{i}}[\mathbf{j},\mathbf{q}] = m_{p,\mathbf{j}}^{\mathbf{q}} \quad , \quad \mathbf{d}_{u,p,\mathbf{i}}[\mathbf{j}] = \langle u \rangle_{\mathbf{j}} \ . \tag{43}$$

For irregular cells, the moment matrices $\mathbf{M}_{\alpha,p,\mathbf{i}}$ and $\mathbf{M}_{\beta,p,\mathbf{i}}$ are identical, and are given by:

$$\mathbf{M}_{\beta,p,\mathbf{i}}[\mathbf{j},\mathbf{q}] = \mathbf{M}_{\alpha,p,\mathbf{i}}[\mathbf{j},\mathbf{q}] = \mathbf{x}_{p,\mathbf{j}}^{\mathbf{q}} \ , \tag{44}$$

$$\mathbf{d}_{\alpha,p,\mathbf{i}}[\mathbf{j}] = \alpha(\mathbf{x}_{p,\mathbf{j}}) \ , \ \mathbf{d}_{\beta,p,\mathbf{i}}[\mathbf{j}] = \beta(\mathbf{x}_{p,\mathbf{j}}) \ . \tag{45}$$

The columns of these moment matrices consist of all moments up to order P. We form one of each of the matrices $\mathbf{M}_{u,p,\mathbf{i}}$ and $\mathbf{M}_{\alpha,p,\mathbf{i}}$ per irregular cell and use a different matrix $\mathbf{G}_{\beta,p,\mathbf{i}\pm\mathbf{e}_d}$ for each face to create a stencil $\mathbf{s}_{\beta,\mathbf{i}}^T$ for the flux integral along a face, where:

$$\mathbf{G}_{\beta,p,\mathbf{i}\pm\mathbf{e}_d}[\mathbf{q},\mathbf{r}] = q_x m_{p,\mathbf{i}\pm\mathbf{e}_d,x}^{\mathbf{q}+\mathbf{r}-\mathbf{e}_x} + q_y m_{p,\mathbf{i}\pm\mathbf{e}_d,y}^{\mathbf{q}+\mathbf{r}-\mathbf{e}_y} \tag{46}$$

$$\mathbf{s}_{\beta,\mathbf{i}}^T = \mathbf{d}_{\beta,p,\mathbf{i}}^T (\mathbf{M}_{\beta,p,\mathbf{i}}^\dagger)^T \mathbf{G}_{\beta,p,\mathbf{i}\pm\mathbf{e}_d}^T \mathbf{M}_{u,p,\mathbf{i}} \ . \tag{47}$$

For the linear term, our stencil is given by (34).

3.1.3. Cut Cells

In cut cells, the moment matrix is additionally used to enforce jump conditions. When interpreted as an overdetermined system, our interpolating polynomials are being constrained to satisfy interface matching conditions. Alternatively as an underdetermined system, we use jump conditions as data to cancel truncation error terms. It is necessary to enforce these jump conditions so that our discrete operator is not degenerate.

Let $\mathcal{N}_{p,\mathbf{i}}$ be two neighborhoods of cells in their respective phases around the cut cell $\mathbf{i}$. See Figure 4 (a). As with irregular cells, we can form two moment matrices $\mathbf{M}_{u,p,\mathbf{i}}$ and data vectors $\mathbf{d}_{u,p,\mathbf{i}}$ of cell averaged values. For each cut cell $\mathbf{j}$ in $\mathcal{N}_{+,\mathbf{i}} \cup \mathcal{N}_{-,\mathbf{i}}$, we want to enforce the two jump conditions (9) and (10). Expressing these in terms of moments and Taylor coefficients, we have

$$\int_{A_{B,\mathbf{j}}} u^+ - u^- \ dA = \sum_{|\mathbf{q}|\leq P} (c_{u,+}^{\mathbf{q}} - c_{u,-}^{\mathbf{q}}) m_{B,\mathbf{j}}^{\mathbf{q}} + O(h^{P+2}) \ , \tag{48}$$

and for the jump in the flux:

$$\int_{A_{B,\mathbf{j}}} (\beta^+ \nabla u^+ - \beta^- \nabla u^-) \cdot \hat{\mathbf{n}} \ dA = \sum_{|\mathbf{r}|,|\mathbf{q}|\leq P} \left(c_{\beta,+}^{\mathbf{r}} c_{u,+}^{\mathbf{q}} - c_{\beta,-}^{\mathbf{r}} c_{u,-}^{\mathbf{q}}\right) \left[q_x m_{B,\mathbf{j},x}^{\mathbf{q}+\mathbf{r}-\mathbf{e}_x} + q_y m_{B,\mathbf{j},y}^{\mathbf{q}+\mathbf{r}-\mathbf{e}_y}\right] + O(h^{P+1}) \ . \tag{49}$$

These expressions are both linear in the coefficients $c_{u,p}^{\mathbf{q}}$, and they couple both sets of coefficients by interpolating u^+ and u^- simultaneously. The resulting matrices $\mathbf{M}_{J,+,\mathbf{i}}$ and $\mathbf{M}_{J,-,\mathbf{i}}$ have rows with the jump condition expressions and columns corresponding to $\mathbf{c}_{u,+}$ and $\mathbf{c}_{u,-}$, respectively. The vector $\mathbf{d}_{J,\mathbf{i}}$ consists of the given jump condition data. Finally we form the moment matrix $\mathbf{M}_{\mathbf{i}}$ which is defined as:

$$\mathbf{M}_{\mathbf{i}} = \begin{bmatrix} \mathbf{M}_{u,+,\mathbf{i}} & 0 \\ 0 & \mathbf{M}_{u,-,\mathbf{i}} \\ \mathbf{M}_{J,+,\mathbf{i}} & \mathbf{M}_{J,-,\mathbf{i}} \end{bmatrix} \ , \tag{50}$$

and we solve for both sets of coefficients simultaneously:

$$\begin{bmatrix} \mathbf{c}_{u,+} \\ \mathbf{c}_{u,-} \end{bmatrix} = \mathbf{M}_{\mathbf{i}}^{\dagger} \begin{bmatrix} \mathbf{d}_{u,+,\mathbf{i}} \\ \mathbf{d}_{u,-,\mathbf{i}} \\ \mathbf{d}_{J,\mathbf{i}} \end{bmatrix} = \begin{bmatrix} (\mathbf{M}_{\mathbf{i}}^{\dagger})_V & (\mathbf{M}_{\mathbf{i}}^{\dagger})_J \end{bmatrix} \begin{bmatrix} \mathbf{d}_{u,+,\mathbf{i}} \\ \mathbf{d}_{u,-,\mathbf{i}} \\ \mathbf{d}_{J,\mathbf{i}} \end{bmatrix} = (\mathbf{M}_{\mathbf{i}}^{\dagger})_V \begin{bmatrix} \mathbf{d}_{u,+,\mathbf{i}} \\ \mathbf{d}_{u,-,\mathbf{i}} \end{bmatrix} + (\mathbf{M}_{\mathbf{i}}^{\dagger})_J \mathbf{d}_{J,\mathbf{i}} \ . \tag{51}$$

If values in $\mathbf{d}_{J,\mathbf{i}}$ are nonzero, then forming a stencil using these coefficients will result in adding a scalar to the right hand side $\langle f \rangle_{p,\mathbf{i}}$. For example, for the linear term we would have:

$$\mathbf{s}_{\alpha,p,\mathbf{i}}^T = \left[\mathbf{d}_{\alpha,p,\mathbf{i}}^T (\mathbf{M}_{\alpha,p,\mathbf{i}}^{\dagger})^T \mathbf{G}_{\alpha,p,\mathbf{i}}^T (\mathbf{M}_{\mathbf{i}}^{\dagger})_V \right] + \left[\mathbf{d}_{\alpha,p,\mathbf{i}}^T (\mathbf{M}_{\alpha,p,\mathbf{i}}^{\dagger})^T \mathbf{G}_{\alpha,p,\mathbf{i}}^T (\mathbf{M}_{\mathbf{i}}^{\dagger})_J \mathbf{d}_{J,\mathbf{i}} \right] \ , \tag{52}$$

where the second term in brackets is a scalar. To form the moment matrices $\mathbf{M}_{\alpha,p,\mathbf{i}}$ and $\mathbf{M}_{\beta,p,\mathbf{i}}$, we can just use the same neighborhoods as the $\mathbf{M}_{u,p,\mathbf{i}}$; we do not need to couple these systems because there are no external constraints on the jumps in coefficients.

3.1.4. Conservation

For each cut cell and irregular cell, we have shown how to obtain a stencil for $\int_A \beta \nabla u \cdot \hat{\mathbf{n}} \, dA$ on each face A of any cell. However, in order to have a conservative method, we must have only one flux stencil per non-EB face. A simple solution to this problem is to average the flux stencils between pairs of neighboring irregular and cut cells. Although this results in larger stencils, it has the advantage of coupling a layer of irregular cells to the interface jump conditions. This is because the irregular cell stencils that border cut cells will share stencil information with cut cells, which incorporate interface jump conditions. We reiterate that this is not a significant issue because the density of the linear system is dominated by the size of the regular cell stencil. For regular cells we do not have to average with neighbors because our flux stencils were created individually for each face and are symmetric about that face. At a cell face which is shared between an irregular and regular cell, we use the regular cell flux stencil.

3.2. Neighborhood Selection and Weighting

In general, neighborhoods need to be chosen so that resulting moment matrices are overdetermined. So as to not perform some sort of search based on local geometry, we opt to make the neighborhood sufficiently large to accommodate a reasonably smooth geometry. For any irregular or cut cell in phase p, we let $\mathcal{N}_{p,\mathbf{i}}$ be those cells in phase p that lie in the the square of cells with side length $2P+1$ surrounding cell i. See Figure 4. As is done in [12] and [17], we employ a weighted least-squares approach to force stencil weights to decay with distance faster than the growth of the highest polynomial term. To each piece of data in $\mathbf{d}$ we assign a weight that is inversely related to its distance from the centroid of volume $V_{p,\mathbf{i}}$. If $\delta_{\mathbf{j}}$ is this distance, then the corresponding row of the moment matrix and $\mathbf{d}$ are multiplied by $w_{\mathbf{j}}$, where

$$w_{\mathbf{j}} = \frac{1}{(1+\delta_{\mathbf{j}})^{P+1}} \ , \tag{53}$$

where P is the order of the scheme. This forms a diagonal weight matrix $\mathbf{W}$, which gives us the weighted least-squares solution:

$$\mathbf{c}_u = (\mathbf{WM})^{\dagger}\mathbf{Wd} \ . \tag{54}$$

This weighting does not affect the truncation error: with some matrix algebra we can see that this weighted least-squares solution is equivalent to a change of basis in the undetermined formulation:

$$\mathbf{s} = \mathbf{W}\left(\mathbf{M}^T\mathbf{W}\right)^{\dagger}\mathbf{g} \implies \tag{55}$$

$$\mathbf{M}^T\mathbf{W}\mathbf{W}^{-1}\mathbf{s} = \mathbf{g} \ . \tag{56}$$

Since $\mathbf{M}^T\mathbf{s} = \mathbf{g}$ still holds, the truncation error is unaffected by the weighting. This has proved to be an effective tool for controlling the spectrum and conditioning of the discrete operator; see [12]. We compute the pseudoinverse using the SVD algorithm in *LAPACK* [18].

3.3. Solver and Software Implementation

We assemble the stencils to form the linear system

$$\mathbf{Lu} = \mathbf{f} + \mathbf{r} \ , \tag{57}$$

where $\mathbf{r}$ represents the contribution to the right hand side from the jump conditions. Following [12], we precondition this system by left multiplying with the diagonal matrix whose (i,i) entry is $\frac{|V_i|}{h^2}$; i.e. we multiply each row by the volume fraction of that cell. This simple preconditioner eliminates the volume scaling associated with very small volume fractions.

We solve the linear system using Krylov subspace methods and preconditioners provided by the *PETSc* library [19], [20]. Since the linear system is non-symmetric, we use BiCG-Stab or GMRES. We have experimented with the *PETSc* algebraic multigrid and block Jacobi preconditioners. One of these options is typically sufficient, but if they fail, we use the direct solver *SuperLU* [21]. In future research we will develop a geometric multigrid preconditioner similar to that in [8] or [22]. Our method is well-suited for geometric multigrid because the Taylor series formulation makes interpolation straightforward. However, we emphasize that an efficient solver is not the focus of this particular paper. The algorithm is implemented using the *Chombo* software library [23], which allows for large-scale parallelization of the algorithm. Visualizations are created using *VisIt* [24].

4. Numerical Tests

We validate our method with a series of numerical tests. The goals of this section are to:

1. Validate the truncation error analysis and measure the solution error,

2. Demonstrate consistent accuracy for problems with large coefficient and solution jumps, and

3. Demonstrate convergence on non-trivial interface geometries.

Although our scheme is designed to be have arbitrary order of accuracy, we have evaluated it for just $P \in \{2, 4, 6\}$. We measure the error $\langle e \rangle_{\mathbf{i}}$ as discrete cell averages, and evaluate it using discrete L^p norms:

$$\|e\|_1 = \int_\Omega |e| dV = \sum_{p,\mathbf{i}} \left| \langle e \rangle_{p,\mathbf{i}} \right| |V_{p,\mathbf{i}}| \tag{58}$$

$$\|e\|_\infty = \max_{p,\mathbf{i}} \langle e \rangle_{p,\mathbf{i}} \ . \tag{59}$$

We note that we are primarily interested in the convergence of the error; the actual magnitude of the error is dependent on the scaling of the coefficients, solution, and source term, so it can vary dramatically between tests.

4.1. Truncation and Solution Error Validation

Our physical domain for all tests is $\Omega = [-1, 1]^2$, intersected by some interface Γ. Our first interface Γ is an ellipse with major axis of length $\frac{\pi}{4}$ and minor axis of length $\frac{\pi}{8}$. The superscript $+$ refers to quantities enclosed by the interface and the superscript $-$ refers to quantities on the exterior of the interface. We test our discretization using the method of manufactured solutions, such that $\alpha^\pm, \beta^\pm, u^\pm$ are all constructed as linear combinations of periodic functions p_{k_x,k_y} on the square:

$$p_{k_x,k_y} = \cos^2(\pi k_x x) \sin^2(\pi k_y y) \ , \tag{60}$$

and let $k_x, k_y \in \{-2, -1, 1, 2\}$, creating 20 total basis functions. We randomly generate different sets of coefficients $c_{k_x,k_y} \in [-1, 1]$ for each of the six functions $\alpha^\pm, \beta^\pm, u^\pm$. An exact $f^\pm$ is formed by applying the exact differential operator to $u^\pm$. The variable coefficient fields are offset by a positive constant so that they are nonnegative everywhere. Figure 5(a) shows the solution for one particular example.

The global truncation error $\mathbf{e}$ and solution error $\mathbf{t}$ are defined as

$$\mathbf{e} = \mathbf{u}^e - \mathbf{u} \tag{61}$$

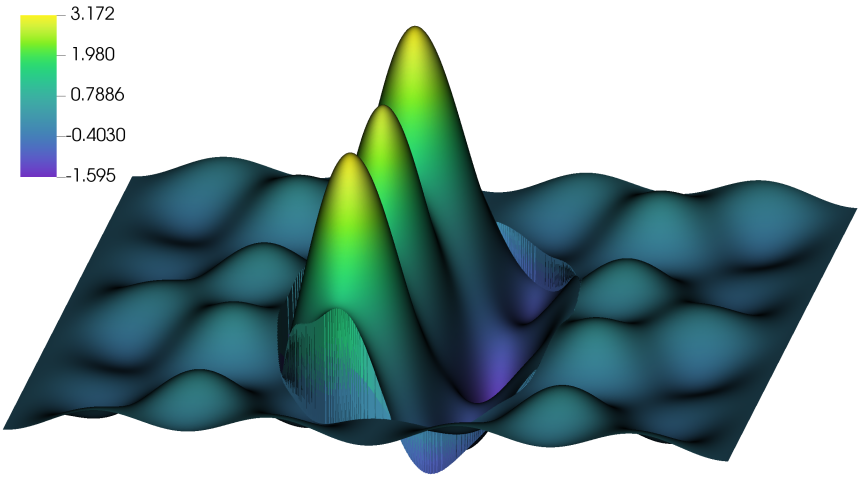
$$\mathbf{t} = (\mathbf{L}\mathbf{u}^e - \mathbf{r}) - \mathbf{L}^e\mathbf{u}^e \ , \tag{62}$$

where $\mathbf{u}^e$ is the exact solution and $\mathbf{L}^e$ is the exact operator, so that the solution error satisfies the equation

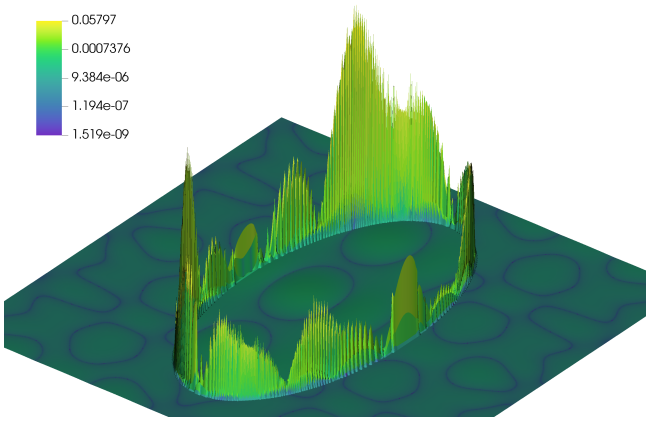
$$\mathbf{L}\mathbf{e} = \mathbf{t} \ . \tag{63}$$

Note that (63) implies homogeneous jump conditions on the error; there is no contribution to the right-hand-side of this system from jump conditions.

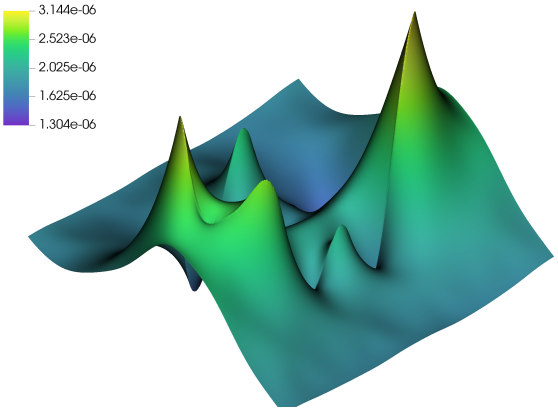
Our truncation error analysis predicts an order $P - 1$ truncation error in cut and irregular cells and an order P truncation error in regular cells. In Figure 5(b) we see that truncation error (for $P = 4$) is almost



(a)



(b)



(c)

Figure 5: Results of the ellipse boundary tests for $P = 4$, $n = 512$. Plots of the (a) exact solution, (b) absolute value of truncation error (log scale), and (c) absolute value of solution error (log scale). Note that the truncation error is concentrated at the interface, and its influence on the solution error is much smoother after inverting the elliptic operator.

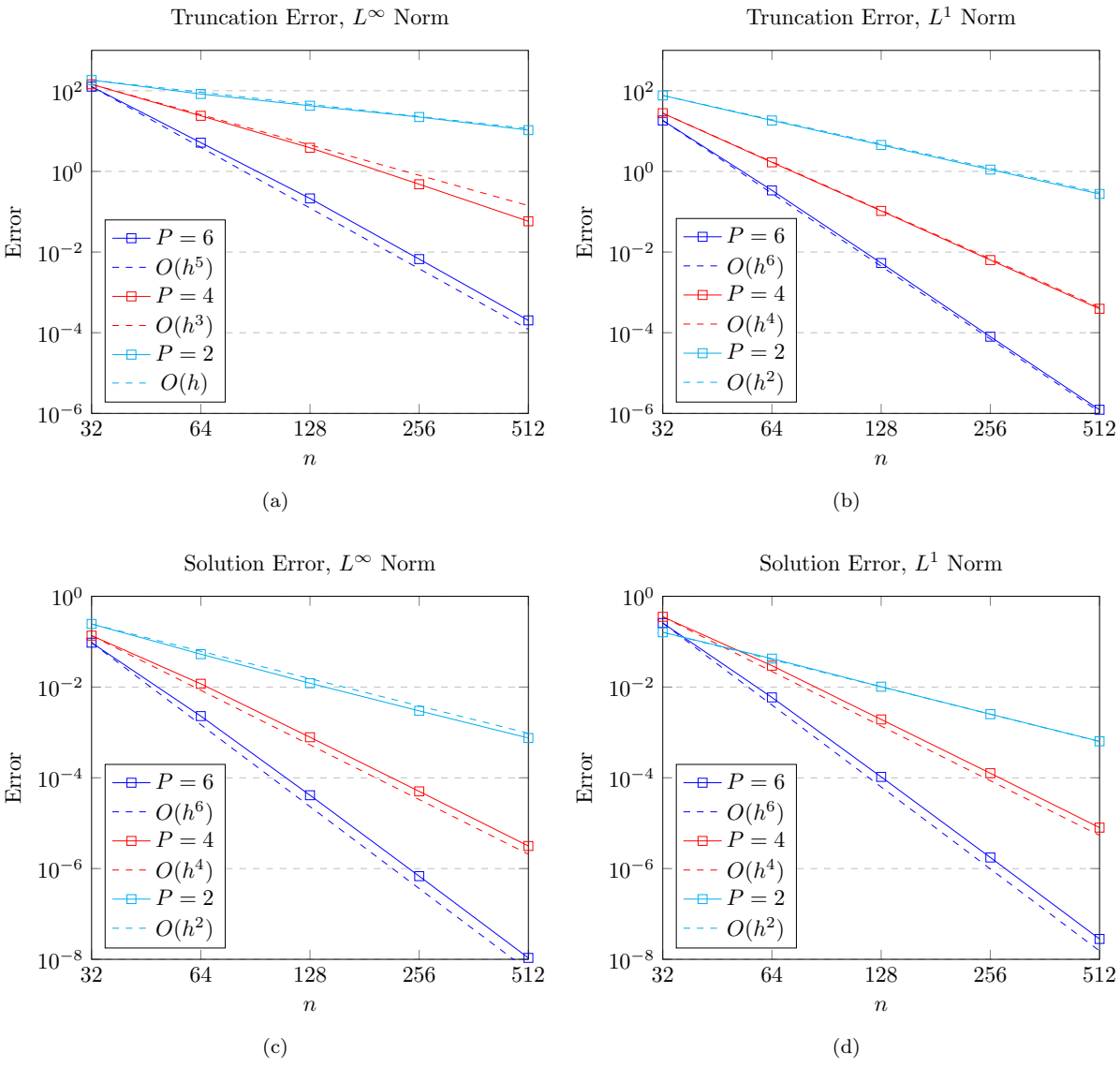


Figure 6: Truncation error L^∞ norm (a), L^1 norm (b), and solution error norms (c) and (d), respectively, for the ellipse geometry. We observe the expected rate of convergence for both quantities in both norms. The truncation error is dominated by behavior at the interface, which because it is codomension-1 smaller, achieves one order higher in L^1 norm (see Figure 5). With regards to the magnitude of the error, for this problem the L^∞ norm of the manufactured solution is $O(1)$, so the errors plotted in (c) and (d) can be interpreted as relative errors. The L^∞ norm of the right-hand-side $\mathbf{f}$ is $O(10^3)$, meaning the relative truncation error at $n = 32$ is the same order of magnitude as the solution error at $n = 32$.

entirely concentrated in cut and irregular cells. This is reflected in Figure 6(a), as the max norm of the truncation error converges at order $P-1$, the expected rate for cells at and near the interface. However, the number of cut and irregular cells is of order h^{-1} because it is a codimension one smaller region, while the number of regular cells is scales like h^{-2}. Therefore for the L^1 norm we have:

$$\|t\|_1 = \sum_{\mathbf{i}\in\Omega_C\cup\Omega_I} \left|\langle t\rangle_{p,\mathbf{i}}\right| |V_{p,\mathbf{i}}| + \sum_{\mathbf{i}\in\Omega_R} \left|\langle t\rangle_{p,\mathbf{i}}\right| |V_{p,\mathbf{i}}| = \sum_{\mathbf{i}\in\Omega_C\cup\Omega_I} O(h^{P+1}) + \sum_{\mathbf{i}\in\Omega_R} O(h^{P+2}) \tag{64}$$

$$= O(h^{-1})O(h^{P+1}) + O(h^{-2})O(h^{P+2}) = O(h^P)\ . \tag{65}$$

This is confirmed in 6 (b); we see clean order P convergence for the L^1 norm of the truncation error. Given that the truncation error at the interface is orders of magnitude greater than truncation error elsewhere, the L^1 norm of the truncation error is also dominated by behavior at the interface.

Although we do not have an analytical bound on $\left\|\mathbf{L}^{-1}\right\|$, based on the analysis in [7] and the results in [12], [8], [14] and others, we expect the solution error to converge at order P in both norms. This behavior is shown in Figures 5(b) and (c); the solution error is roughly of the same order of magnitude everywhere in the domain. We plan on further analysis to explore the combined effects of the homogeneous jump conditions imposed on the error equation and the regularity of the elliptic operator, but the empirical results demonstrate the desired convergence rates.

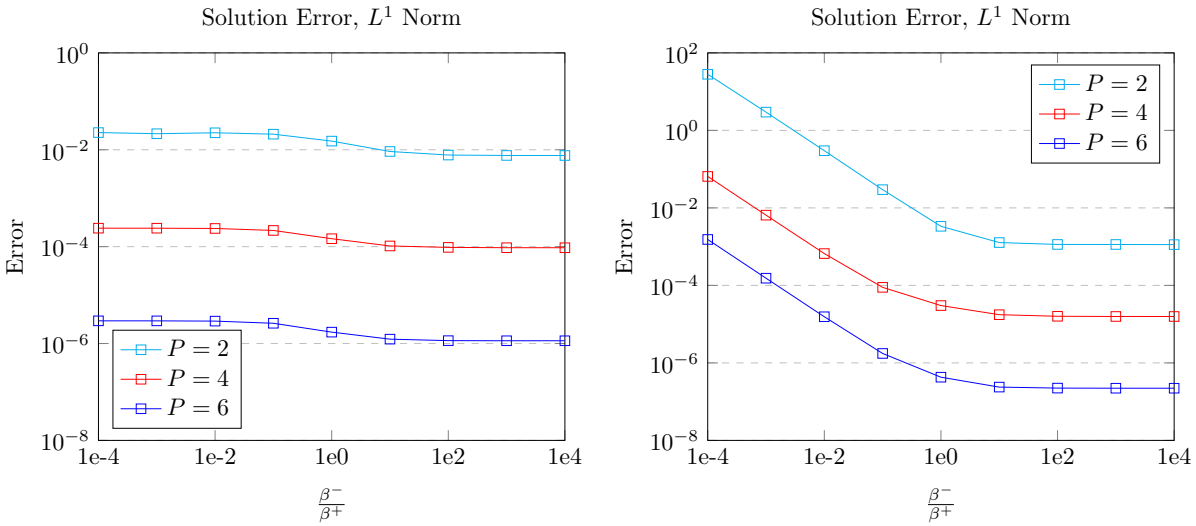
4.2. Discontinuous Diffusion Coefficient

Next we test the robustness of our scheme on problems with large jumps in the diffusion coefficient, as is in common in the literature (see [8], [14], [11], [5]). We set the linear term coefficient $\alpha^{\pm} = 0$, and let the diffusion coefficients be constant, varying the ratio $\frac{\beta^-}{\beta^+}$ from 10^4 to 10^{-4}. Specifically, we fix $\beta^+ = 1$ and vary β^- from 1 to 10^4, and vice versa. The manufactured solution is the same as in the truncation and solution error tests. We are interested in studying the relationship between the solution error and the ratio of diffusion coefficients, so we fix the grid spacing at $h = 256^{-1}$. However, we introduce two geometries that expose different error characteristics.

The first interface geometry is the zero level set of the function

$$\psi(x,y) = \frac{1}{4}\cos(\pi y) + x + \frac{\pi}{100}\ , \tag{66}$$

which is simply a cosine in the xy plane. We impose periodic boundary conditions in the y direction and Dirichlet boundary conditions in the x direction. The + region is the to the right of the interface for this geometry. We impose Dirichlet boundary conditions by filling layers of ghost cells with exact solution values. In this case both phases are tied to boundary conditions, so we do not expect any significant difference between large and small value of $\frac{\beta^-}{\beta^+}$. For the second test, we use the "annulus" interface geometry given in section 3.1 of [14]. This interface shape can be seen in Figures 8(c) and (e). In the former case, the +



(a) (b)

Figure 7: Solution error for discontinuous diffusion coefficient tests. Figure (a) is for the cosine geometry, whereas (b) is the annulus geometry (we only plot L^1 error because the L^∞ norms behave nearly identically).

(interior) phase has a relatively large coefficient ratio (10^4), while in the latter it is the inverse (10^{-4}). Given that in these cases the $-$ phase has domain boundary conditions while the $+$ phase does not, we expect to see different error behavior as $\frac{\beta^-}{\beta^+}$ varies.

For the cosine geometry, we see that accuracy is mostly unaffected by changes in the diffusion coefficient ratio (Figure 7(a)). In Figure 8(a) and (b), the truncation error again is larger at the interface but the solution error is smooth, similar to the previous tests. We observed no impact of the conditioning of the discretization matrix on the solution accuracy as the ratio of coefficients varies over 8 orders of magnitude.

For the second test, instead we see that the error is about 4 orders of magnitude higher when we have the diffusion coefficient on the interior of the domain is much larger than the diffusion coefficient on the exterior of the domain ($\beta^-/\beta^+ = 10^{-4}$). This result is consistent with results reported in Figure 2 of [8], as well as Figure 17b of [11]. Through potential theoretic arguments, we believe this is a result of solution errors in the interior region not being "tied down" to any domain boundary condition, as in the cosine test. Because of the gradient jump condition (3), any gradient *errors* in the interior are multiplied by $\beta^+/\beta^- = 10^4$ in their contribution to the exterior domain gradients at the interface, forcing the interior solution there to "drift" in proportion. However, with this scaling the convergence rates are still retained, but with an error constant reflecting this ratio in diffusion coefficients.

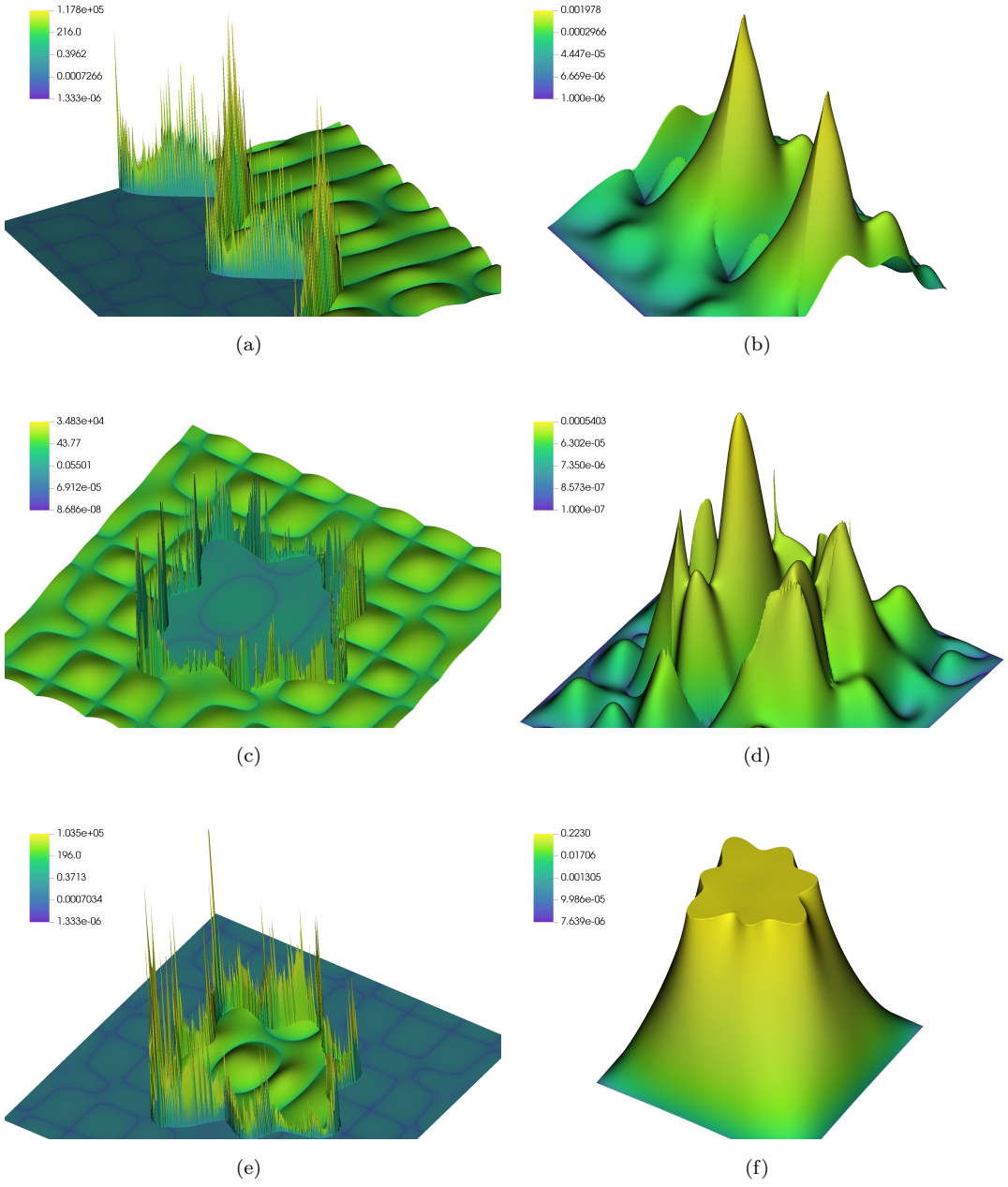


Figure 8: Plots of the absolute value of truncation error (left column) and absolute value of solution error (right column) for $P = 2$, $n = 512$, on a log scale. Figures show: (a) and (b) cosine geometry with $\frac{\beta^-}{\beta^+} = 10^{-4}$; (c) and (d) annulus geometry with $\frac{\beta^-}{\beta^+} = 10^{4}$; and (e) and (f) annulus geometry with $\frac{\beta^-}{\beta^+} = 10^{-4}$. Note that the larger, rough trunction error near the interface becomes smoother, rapidly decaying error in the solution (except (f), see text for discussion).

4.3. Discontinuous Solution

We perform a similar test with a solution that has large jumps at the interface. $\beta^{\pm}$ and $u^{\pm}$ are the same as in the solution and truncation error test, and we again set $\alpha^{\pm} = 0$. The $u^{\pm}$ fields are multiplied by scaling factors $s^{\pm}$ to create large jumps in the solution, and we use the same two geometries as the discontinuous coefficients test. We observe L^1 and L^∞ errors that are proportional to the larger of the two scaling coefficients (see Figure 9). This scaling does not magnify the error because it appears as a large discontinuity in the source term, as well as in the jump conditions, which both contribute only to the right hand side of the linear system. This test highlights the importance of the having two separate degrees of freedom in each cut cell, from which we are able to accurately reproduce a solution and gradients which jump by up to 4 orders of magnitude across the interface.

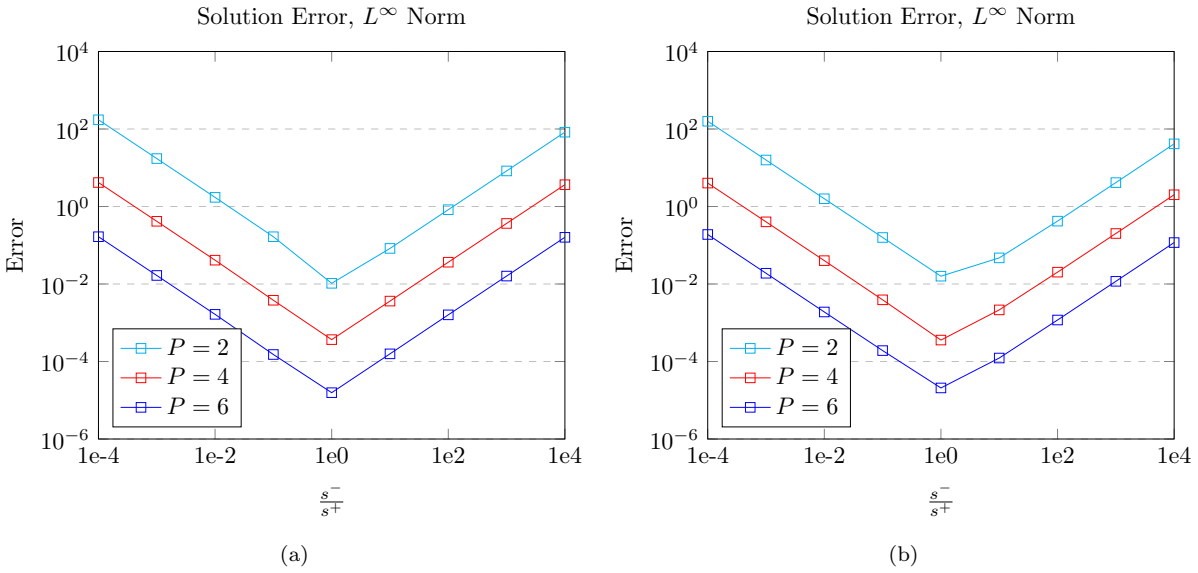
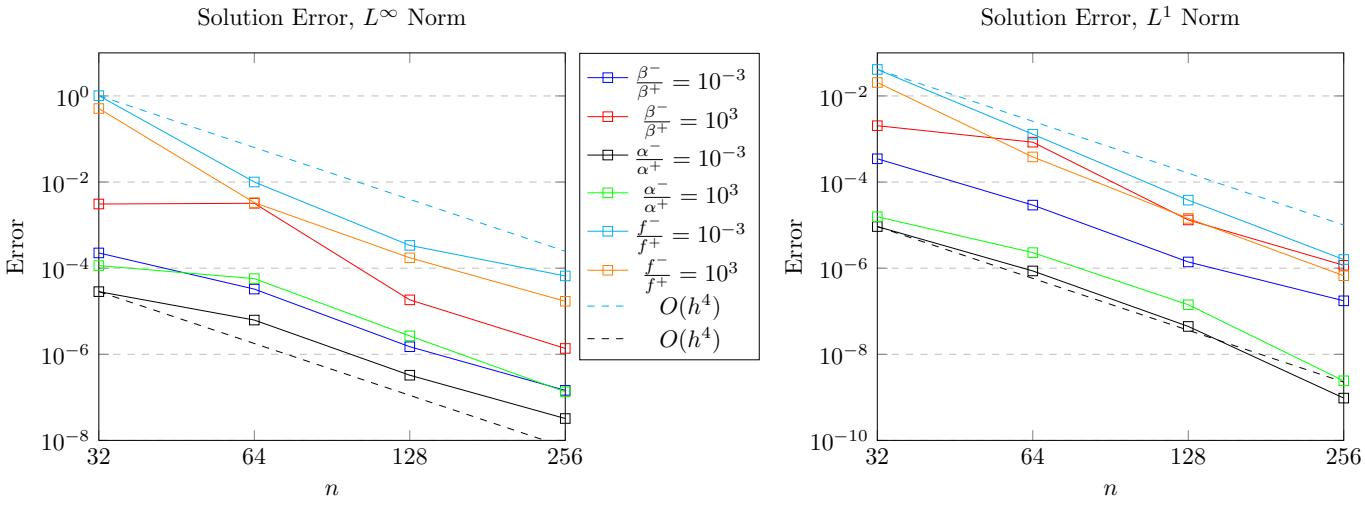


Figure 9: Comparison of solution maximum errors for two geometries (a) cosine, and (b) annulus, and over a range of solution scaling ratios, fixing grid size at $n = 128$ cells and varying the order of accuracy P (note that the L^1 error is nearly identical). For this test, we set $s^- = 1$ for ratios less than 1, and $s^+ = 1$ for ratios greater than 1. Thus the solution error scales with the larger of the two coefficients. Since the unscaled exact solution is $O(1)$, the error relative to the scaled solution is, for example, $O(10^{-5})$ for $P = 6$ for all scaling ratios.

4.4. Imposing Homogeneous Jump Conditions

Lastly, we test the ability of the method to impose jump conditions as a constraint. We let $\alpha^{\pm}, \beta^{\pm}$ be the same as in the truncation and solution error test, and use the manufactured solution $u^{\pm}$ from that test as the source term $f^{\pm}$. We impose homogeneous (zero) jump conditions and focus on the annulus geometry. For this test we fix $P = 4$ and test the scheme with a variety of coefficient and source term scalings. The error is measured by using the $n = 512$ numerical solution as the exact solution, with the results in Figure 10.



(a) (b)

Figure 10: L^∞ and L^1 norm of errors for Richardson convergence test for a variety of scaling factor ratios in linear term coefficient, diffusion coefficient, and source term. We fix $P = 4$ and observe fourth order convergence in the L^1 norm and nearly fourth order convergence in the max norm. For this test, the L^∞ norm of the right hand side $\mathbf{f}$ is $O(1)$, so the errors for the coefficient ratio scaling tests ($\frac{\beta^-}{\beta^+}, \frac{\alpha^-}{\alpha^+}$) can be interpreted as relative errors. For the source term scaling tests ($\frac{f^-}{f^+}$), the magnitude of the relative error is 10^{-3} times the plotted error.

We observe roughly fourth order convergence for all tests, although there is more variation in convergence compared to the manufactured solution tests.

5. Conclusion

We have developed a finite volume method for the variable coefficient elliptic interface problem and demonstrated up to sixth order accurate on a variety of test problems. In developing this method we gave a general truncation error analysis that justifies the use of stencils based on least-squares interpolation. Our stencils are derived from cell-centered Taylor polynomials which are implicitly defined in terms of local values of the solution and interface jumps or boundary conditions, where appropriate. To enforce conservation, we choose a single flux on each face which is an average of the flux calculated in neighboring cells from the respective Taylor polynomials. In cells away from the interface, we take advantage of standard finite volume symmetries to build stencils with a minimal footprint.

Future research will involve 1) extending our method to three dimensions, 2) building an efficient geometric multigrid solver along the lines of [22],~~1) building an efficient geometric multigrid solver along the lines of [22], 2) extending our method to three dimensions~~ and 3) incorporating adaptive mesh refinement (AMR). The methodology of this paper is, in general, dimension independent because stencils are expressed through multidimensional Taylor expansions and geometric moments. Geometric quantities in three dimen-

sions can be calculated using the algorithm in [15]; however, we plan to extend the geometry generation method described in Appendix A to three dimensions as well. However, we expect significantly increased computational cost: there to be $O(n^2)$ cut and irregular cells in three dimensions, and for each of these cells we must solve a least-squares system of size $O(P^3)$. Fortunately, the set up stage of the problem (calculating geometric moments, solving least-squares problems, compiling stencils) is embarrassingly parallel and is well-suited for GPU acceleration. A significant difficulty for extending the method to three dimensions will be the efficiency of the solver. One possible research direction could be building a geometric multigrid solver. Prior works including [11] and [25] describe various strategies for multigrid smoothing, coarsening, and interpolation for the elliptic interface problem in the finite difference and finite volume context. Lastly, to apply the method to realistic problems, particularly in three dimensions, we aim to reduce computational cost by incorporating AMR. The authors of [26] present a methodology for pairing AMR with *PETSC* algebraic multigrid solvers and [22] provides details on a hybrid geometric-algebraic multigrid solver for a high-order EB, adaptive mesh discretization of the Poisson equation. The prior method presented in [12] accomplishes these three objectives for Poisson's equation in more complex geometries, including boundaries with kinks, using "smoothed" constructive solid geometry capabilities of the Chombo software library [23]. This would enable this method to be used in discretizations for large scale science applications.

Finally, further exploration is also needed of the theory of undetermined stencil systems (using the moment matrix transpose, $\mathbf{M}^T$). This paper has shown that building a stencil of a given order and truncation error still allows infinitely many valid stencils; this fact could be exploited to promote sparsity or alter the conditioning or stability of the operator, and we are drafting a paper with analysis that may provide specific algorithmic guidance. This research direction will be particularly useful for extending the present method to parabolic problems, where jumps in coefficients at the interface might lead to different numerical boundary errors in time.

Funding:

This work is supported by the U.S. Department of Energy, Office of Science, Advanced Scientific Computing Research, Base Math Program, under contract number DE-AC02-05CH11231.

Appendix A. Geometry Generation Algorithm

We specify the interface as a zero level set of an implicit function $\psi(\mathbf{x})$. Therefore, for sufficiently smooth ψ compared to the grid resolution, we assume we can identify cut cells by evaluating ψ at the four corners of each cell. If any of these values have different signs, the cell is tagged as a cut cell. If the interface intersects one face of the cell multiple times, or there are more than two faces intersected by the interface, we consider the geometry to be under-resolved and could refine the mesh or adjust the boundary without

inducing significant errors. Given these assumptions, when the interface intersects a cell it creates a region which is bounded on one side by the interface and on two or three sides by the edges of a square cell (see Figure 1). Volume moments (4) are defined as integrals over this region. Area moments, defined in (6) and (7), are integrals over the portion of the EB that intersects the cut cell. We compute these integrals by approximating the interface with piecewise line segments, and then apply a formula for the integral of monomials along line segments. The vertices of the line segments are roots of ψ, which we find using a simple root finder such as the secant method. By refining this interface iteratively into 2^n line segments, we can calculate a convergent sequence m_n of moment approximations that stops when $|m_{n+1} - m_n|$ reaches machine precision. The convergence of this sequence is accelerated using Richardson extrapolation, which in this case is often referred to as Romberg integration.

A formula for the integral of $x^p y^q$ over an arbitrary polygon can be derived from Green's theorem:

$$\int_P \frac{\partial f(x,y)}{\partial x} dV = \int_C f(x,y)\hat{n}_x dA = \int_C f(x,y) dy \ , \tag{A.1}$$

where C is the boundary of the polygon P. Let $f = \frac{x^{p+1}}{p+1} y^q$, giving us:

$$\int_P x^p y^q dV = \int_C \frac{x^{p+1}}{p+1} y^q dy \ . \tag{A.2}$$

We parameterize each edge segment by:

$$x(t) = (x_{k+1} - x_k)t + x_k = \Delta x t + x_k \tag{A.3}$$

$$y(t) = (y_{k+1} - y_k)t + y_k = \Delta y t + y_k \ , \tag{A.4}$$

where t goes from 0 to 1 and (x_k, y_k) are the ordered vertices of the polygon. The formula for the right hand side of (A.2) along a single line segment $C_k(t)$ is obtained by a binomial expansion:

$$\int_{C_k} \frac{x^{p+1}}{p+1} y^q dy = \frac{1}{p+1} \int_0^1 (\Delta x t + x_k)^{p+1} (\Delta y t + y_k)^q \Delta y dt \tag{A.5}$$

$$= \sum_{i=0}^{p+1} \sum_{j=0}^{q} \frac{\binom{p+1}{i}\binom{q}{j}}{(p+1)(p+2+q-i-j)} (x_k^i y_k^j)(\Delta x)^{p+1-i} (\Delta y)^{q+1-j} \ . \tag{A.6}$$

We follow a similar procedure for area integrals:

$$\int_C x^p y^q = \int_0^1 x(t)^p y(t)^q dC \tag{A.7}$$

$$= \sum_{i=0}^{p} \sum_{j=0}^{q} \frac{\binom{p}{i}\binom{q}{j}}{(p+1+q-i-j)} (x_k^i y_k^j)(\Delta x)^{p-i} (\Delta y)^{q-j} \Delta C \ , \tag{A.8}$$

and to calculate area integrals times unit normals we multiply equation (A.8) by $n_x = \dfrac{\Delta y}{\Delta C}$ or $n_y = \dfrac{-\Delta x}{\Delta C}$, where $\Delta C = \sqrt{\Delta x^2 + \Delta y^2}$. (The tangent vector is rotated 90 degrees clockwise). The integrals of interest are obtained by adding the integrals along all line segments of the polygon in the case of volume moments, or just along the interface in the case of area moments.

References

[1] Z. Li, An overview of the immersed interface method and its applications, Taiwanese Journal of Mathematics 7 (2003) 1–49.

[2] F. Gibou, C. Min, R. Fedkiw, High resolution sharp computational methods for elliptic and parabolic problems in complex geometries, J. Sci. Comput. 54 (2013) 369–413.

[3] I. Babuska, The finite element method for elliptic equations with discontinuous coefficients, Computing 5 (1970) 207–213.

[4] Z. Li, The immersed interface method using a finite element formulation, Applied Numerical Mathematics 27 (1998) 253–267.

[5] R. J. Leveque, Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM Journal on Numerical Analysis 31 (1994) 1019–1044.

[6] 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.

[7] H. Johansen, P. Colella, A cartesian grid embedded boundary method for poisson's equation on irregular domains, Journal of Computational Physics 147 (1998) 60–85.

[8] R. Crockett, P. Colella, D. Graves, A cartesian grid embedded boundary method for solving the poisson and heat equations with discontinuous coefficients in three dimensions, Journal of Computational Physics 230 (2011) 2451–2469.

[9] 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, Journal of Computational Physics 211 (2006) 531–550.

[10] P. Colella, High-order finite-volume methods on locally-structured grids, Discrete and Continuous Dynamical Systems 36 (2016) 4247–4270.

[11] T. Chen, J. Strain, Piecewise-polynomial discretization and krylov-accelerated multigrid for elliptic interface problems, Journal of Computational Physics 227 (2008) 7503–7542.

[12] D. Devendran, D. Graves, H. Johansen, T. Ligocki, A fourth-order Cartesian grid embedded boundary method for Poisson's equation, Communications in Applied Mathematics and Computational Science 12 (2017) 51 – 79.

[13] Q. Zhang, H. Johansen, P. Colella, A fourth-order accurate finite-volume method with structured adaptive mesh refinement for solving the advection-diffusion equation, SIAM Journal on Scientific Computing 34 (2012) B179–B201.

[14] D. Bochkov, F. Gibou, Solving elliptic interface problems with jump conditions on cartesian grids, Journal of Computational Physics 407 (2020) 109269.

[15] P. Schwartz, J. Percelay, T. J. Ligocki, H. Johansen, D. T. Graves, D. Devendran, P. Colella, E. Ateljevich, High-accuracy embedded boundary grid generation using the divergence theorem, Commun. Appl. Math. Comput. Sci. 10 (2015) 83–96.

[16] R. J. LeVeque, Finite Difference Methods for Ordinary and Partial Differential Equations, Society for Industrial and Applied Mathematics, 2007.

[17] N. Overton-Katz, X. Gao, S. Guzik, O. Antepara, D. T. Graves, H. Johansen, A fourth-order embedded boundary finite volume method for the unsteady stokes equations with complex geometries, arXiv (2022).

[18] V. A. Barker, L. S. Blackford, J. Dongarra, J. D. Croz, S. Hammarling, M. Marinova, J. Waśniewski, P. Yalamov, LAPACK95 Users' Guide, Society for Industrial and Applied Mathematics, 2001.

[19] S. Balay, S. Abhyankar, M. F. Adams, S. Benson, J. Brown, P. Brune, K. Buschelman, E. Constantinescu, L. Dalcin, A. Dener, V. Eijkhout, W. D. Gropp, V. Hapla, T. Isaac, P. Jolivet, D. Karpeev, D. Kaushik, M. G. Knepley, F. Kong, S. Kruger, D. A. May, L. C. McInnes, R. T. Mills, L. Mitchell, T. Munson, J. E. Roman, K. Rupp, P. Sanan, J. Sarich, B. F. Smith, S. Zampini, H. Zhang, H. Zhang, J. Zhang, PETSc/TAO Users Manual, Technical Report ANL-21/39 - Revision 3.17, Argonne National Laboratory, 2022.

[20] 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.

[21] X. S. Li, J. W. Demmel, SuperLU_DIST: A scalable distributed-memory sparse direct solver for unsymmetric linear systems, ACM Trans. Mathematical Software 29 (2003) 110–140.

[22] D. Devendran, D. T. Graves, H. Johansen, A hybrid multigrid algorithm for poisson's equation using an adaptive, fourth order treatment of cut cells, Technical Report LBNL-1004329, LBNL (2014).

[23] M. Adams, P. Colella, D. Graves, J. Johnson, H. Johansen, N. Keen, T. Ligocki, D. Martin, P. McCorquodale, D. Modiano, P. Schwartz, T. Sternberg, B. V. Straalen, Chombo software package for AMR applications: design document, Technical Report, April 2021.

[24] H. Childs, E. Brugger, B. Whitlock, J. Meredith, S. Ahern, D. Pugmire, K. Biagas, M. C. Miller, C. Harrison, G. H. Weber, H. Krishnan, T. Fogal, A. Sanderson, C. Garth, E. W. Bethel, D. Camp, O. Rubel, M. Durant, J. M. Favre, P. Navratil, VisIt: An End-User Tool For Visualizing and Analyzing Very Large Data, 2012.

[25] A. Coco, G. Russo, Second order finite-difference ghost-point multigrid methods for elliptic problems with discontinuous coefficients on an arbitrary interface, Journal of Computational Physics 361 (2018) 299–330.

[26] M. F. Adams, S. L. Cornford, D. F. Martin, P. McCorquodale, Composite matrix construction for structured grid adaptive mesh refinement, Computer Physics Communications 244 (2019) 35–39.