Journal of Scientific Computing (2023) 97:40
https://doi.org/10.1007/s10915-023-02353-9

Universal AMG Accelerated Embedded Boundary Method Without Small Cell Stiffness

Zhichao Peng[1] · **Daniel Appelö**[2] · **Shuang Liu**[3]

Received: 12 April 2022 / Revised: 15 August 2023 / Accepted: 7 September 2023 /
Published online: 28 September 2023

Abstract
We develop a universally applicable embedded boundary finite difference method, which results in a symmetric positive definite linear system and does not suffer from small cell stiffness. Our discretization is efficient for the wave, heat and Poisson equation with Dirichlet boundary conditions. When the system needs to be inverted we can use the conjugate gradient method, accelerated by algebraic multigrid techniques. A series of numerical tests for the wave, heat and Poisson equation and applications to shape optimization problems verify the accuracy, stability, and efficiency of our method. Our fast computational techniques can be extended to moving boundary problems (e.g. Stefan problem), to the Navier–Stokes equations, and to the Grad-Shafranov equations for which problems are posed on domains with complex geometry and fast simulations are of great interest.

Keywords Algebraic multigrid · Embedded boundary method · Line-by-line interpolation · Radial basis function interpolation

1 Introduction

Fast and accurate simulation of problems arising in engineering and the sciences is of great importance. Often such problems are posed on domains with complex geometry and the numerical methods used in the simulations must account for this. There are numerous methods that are capable of handling geometry, among them are the finite element method [11] and

✉ Shuang Liu
Shuang.Liu@unt.edu

Zhichao Peng
pengzhic@ust.hk

Daniel Appelö
appelo@vt.edu

[1] Department of Mathematics, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, China

[2] Department of Mathematics, Virginia Tech, 225 Stanger Street, Blacksburg, VA 24061-1026, USA

[3] Department of Mathematics, University of North Texas, Denton, TX 76201, USA

other methods using unstructured grids, overset grid methods [9, 37, 39] and embedded boundary methods.

In element based methods a volumetric grid must be generated and in both two and three dimensions this can be a time consuming task, especially if the grids are to be of high quality. On the other hand there are well developed open source unstructured mesh generators available. In methods that use the overset grid (also known as composite grid, or overlapping grid) framework the grids are overset and a Cartesian background grid is coupled through interpolation to local boundary fitted narrow grids near the geometry. The locality of the boundary fitted grids makes the grid generation easier and the quality of the grids is typically very high. The hole-cutting, the process where the interpolation operators between grids are constructed, can be done efficiently [9] and in parallel but there are relatively few software packages available.

In embedded boundary (EB) methods, which is the topic of this paper, the geometry is represented by curves in two dimensions and surfaces in three dimensions. These curves or surfaces are, as the name suggests, embedded in a uniform Cartesian grid that covers the computational domain. In an embedded boundary method there is no need to generate a grid and the geometry is instead incorporated through modified stencils near the boundary that explicitly incorporate boundary conditions. It is straightforward to write an EB mesh generator for the simple geometries we consider in this work. However, while the CAD description of a geometry can be used directly in an EB mesh generator there are, to our knowledge, no such freely available EB mesh generators. This is a drawback for EB, compared to methods using unstructured meshes. In applications such as uncertainty quantification and model order reduction for problems with geometric parameters or parameters determining geometry [19], using a fixed mesh for different geometries to avoid interpolation between solutions associated with different meshes is highly desired and the EB method then becomes an attractive choice.

The purpose of this work is to introduce a universal embedded boundary method that can be used to efficiently solve the wave, heat or Poisson equation with Dirichlet boundary conditions. This is achieved by designing a method that:

- is symmetric and positive definite, so that the conjugate gradient (CG) method can be used,
- is diagonally dominant and with eigenvalues and eigenvectors that closely resemble those of the periodic problem, so that the conjugate gradient method can be accelerated by algebraic multigrid (AMG) techniques,
- and, does not suffer from small cell stiffness so that the wave equation can be marched in time by an explicit method.

The basic ingredient to obtain a symmetric embedded boundary discretization is to use an approximation for the boundary condition that only modifies the diagonal element in the matrix approximating the second derivative. The most straightforward approach is to approximate the second derivative dimension-by-dimension and use linear extrapolation based on the boundary condition and the numerical solution at the interior point to assign the value of the numerical solution at an outside ghost-point. In one dimension this modifies the diagonal element from -2 to $-(1/\theta + 1)$ where $\theta \in (0, 1]$ depends on how the boundary cuts the grid. This does not change the symmetry of the matrix and it also does not change its definiteness. It does however change the spectrum of the matrix, introducing an eigenvalue that scale as $1/\theta$. If used directly for the wave equation this approach will thus suffer from small cell stiffness, forcing the stable time-step to be excessively small. This can be mitigated by the local time-stepping proposed by Kreiss, Petersson and Ystrom in [21] resulting in a provably

stable and efficient embedded boundary method. The same approach was also introduced by Gibou et al. [16] for the Poisson equation and the heat equation with implicit time-stepping. Then, as the time-stepping is already implicit the small cell does not reduce the time-step, however the very large eigenvalues that result from the small cells can result in matrices with large condition numbers and care has to be taken when choosing an iterative solver.

Here we expand on the ideas in [16, 21] but just as in our earlier work [1] we enforce the Dirichlet boundary conditions by interpolating to interior boundary points rather than extrapolate to exterior ghost-points. This subtle yet crucial difference improves previous second-order accurate approaches by removing the small-cell stiffness problem. Moreover, placing boundary points inside the computational domain allows the solution to be "single-valued" for slender geometries, leading to significant algorithmic simplifications. For geometry that is convex it is still possible to use a line-by-line linear interpolation to enforce boundary conditions, but when the geometry is concave this procedure can break down. For such cases we introduce a combined polynomial and radial basis function interpolation that prevents breakdown. We also propose a simple criterion that can be used to test if the system matrix is SPD.

Embedded boundary methods have been used to successfully solve a variety of problems from elasticity [41] to incompressible [34] and compressible flows [32]. Here we do not aim to provide a complete literature survey but rather to mention contributions relevant to the method we introduce. As mentioned above [16] develops a symmetric second order method by imposing the boundary condition through linear extrapolation. This method is analyzed and expanded to quadratic boundary treatment in [18]. To obtain second order accuracy in both the solution and its gradient, [31] proposes a non-symmetric discretization based on a quadratic extrapolation. Adaptive mesh refinement (AMR) techniques for the method in [31] were considered in [6, 7]. Finite volume solvers with embedded boundaries using bilinear interpolation were considered in [17, 36]. Higher order accurate methods for the Poisson and the heat equation can be found in [10, 15]. For wave equations the early works by Kreiss, Petersson and Ystrom [20–22] provided analysis of second order accurate methods with external ghost-points and Dirichlet, Neumann and interface conditions. Higher order accurate methods for the wave equation include [1, 5, 23, 26–29, 38, 40].

The rest of the paper is organized as follows. In Sect. 2, we first present the overall algorithm. Then, we show the details of how interior boundary points are located, the formulation of the line-by-line and radial basis function (RBF)-based interpolation as well as the symmetric positive definite (SPD) checking criteria. In Sect. 3, the performance of the proposed method is demonstrated through a series of numerical experiments. In Sect. 4, we summarize and conclude.

2 Universal Embedded Boundary Discretization of the Laplacian

Our goal is to design an embedded boundary finite difference method, which results in a symmetric positive definite linear system and does not suffer from small cell stiffness. Our discretization can thus be efficient for solving wave, heat and Poisson equation. We now describe the different components of our method one at a time but note that the entire method is summarized in Algorithm 2.

To demonstrate the discretization of the Laplacian operator, consider Poisson's equation in an irregular two-dimensional domain $(x, y) \in \Omega$:

$$\nabla \cdot (\beta(x, y)\nabla u) = f(x, y), \quad (x, y) \in \Omega, \tag{1}$$

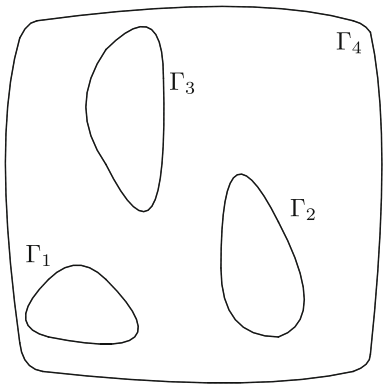

Fig. 1 An illustration of an interior problem with the boundary consisting of four disconnected curves

closed by Dirichlet boundary conditions

$$u(x, y) = u_{\mathcal{D}}^{(l)}(x, y), \quad (x, y) \in \Gamma_l, \ \ l = 1, \ldots, n_{\text{tot}}. \tag{2}$$

The boundary of the domain Ω is a collection of n_{tot} smooth curves Γ_l. A possible set-up for an interior problem is shown in Fig. 1, where one curve encloses the other $n_{\text{tot}} - 1$ curves.

Without loss of generality, we assume that all the boundary interfaces describing the geometry Ω are contained inside the uniform Cartesian grid (see Fig. 2)

$$(x_i, y_j) = (x_L + (i-1)h, y_L + (j-1)h), \ \ i = 1, \ldots, N_x, \ j = 1, \ldots, N_y,$$

discretizing the rectangular domain $[x_L, x_R] \times [y_L, y_R]$ where x_L and y_L are given and $x_{N_x} = x_R$ and $y_{N_y} = y_R$ are determined so that they align with the grid.

To this end we will approximate the Laplacian operator by the central difference scheme:

$$\begin{aligned} \nabla \cdot (\beta(x_i, y_j)\nabla u(x_i, y_j)) \approx & \frac{\beta_{i+\frac{1}{2},j} u_{i+1,j} - \left(\beta_{i+\frac{1}{2},j} + \beta_{i-\frac{1}{2},j}\right) u_{i,j} + \beta_{i-\frac{1}{2},j} u_{i-1,j}}{h^2} \\ & + \frac{\beta_{i,j+\frac{1}{2}} u_{i,j+1} - \left(\beta_{i,j+\frac{1}{2}} + \beta_{i,j-\frac{1}{2}}\right) u_{i,j} + \beta_{i,j-\frac{1}{2}} u_{i,j-1}}{h^2}. \end{aligned} \tag{3}$$

As mentioned in the introduction, a novelty of our method is to enforce boundary conditions through interpolation to interior boundary points. This avoids small cell stiffness. To identify interior boundary points, we first set up a mask grid function $m_{i,j}$ defined to be one inside the geometry and zero outside. That is:

$$m_{i,j} = \begin{cases} 1, & (x_i, y_j) \in \Omega, \\ 0, & \text{otherwise}. \end{cases}$$

For example, if the boundary of Ω is determined by the signed level-set function $\psi(x, y)$=0, then the value of the mask $m_{i,j}$ follows by the sign of $\psi(x_i, y_j)$. An example of a mask grid function is shown in Fig. 2.

We denote grid points inside Ω but adjacent to the boundary as *boundary points* and we denote the remaining interior grid points as *computational points*. Precisely we define:

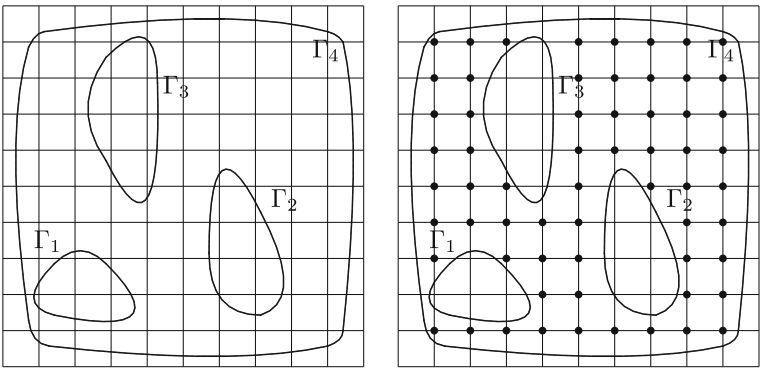

Fig. 2 Embedded boundaries of an interior problem in a rectangular mesh. **a** Discretization of the geometry without the mask shown. **b** Discretization of the geometry with the mask shown, grid points with a filled circle have $m_{ij} = 1$, grid points without a filled circle have $m_{ij} = 0$

- A **boundary point** (x_i, y_j) satisfies:

  (1) $m_{i,j} = 1$,
  (2) $m_{i+1,j} + m_{i-1,j} + m_{i,j+1} + m_{i,j-1} < 4$.

  In other words, (x_i, y_j) is inside Ω, but at least one of its nearest neighbors is outside.
- A **computational point** (x_i, y_j) satisfies:

  (1) $m_{i,j} = 1$,
  (2) $m_{i+1,j} + m_{i-1,j} + m_{i,j+1} + m_{i,j-1} = 4$.

  In other words, (x_i, y_j) and all its nearest neighbors are all inside Ω.

2.1 Imposing the Boundary Condition at Boundary Points

We now describe how we use interpolation together with the boundary conditions to assign values to the solution at interior boundary points. We use two strategies, line-by-line interpolation and radial basis interpolation. We first describe the line-by-line approach.

2.1.1 Line-by-Line Interpolation

We describe the line-by-line approach for a case such as the one depicted in the left image of Fig. 3.

Let (x_i, y_j) be a computational point and $(x_{\text{BP}}, y_{\text{BP}}) = (x_{i-1}, y_j)$ be a boundary point associated with the boundary Γ. If the point (x_{i-2}, y_j) is outside Ω, we introduce a local one-dimensional coordinate system ξ along the grid line in x passing through $(x_{\text{BP}}, y_{\text{BP}})$. We denote the intersection of the horizontal line $y = y_j$ and the boundary Γ by ξ_Γ, and the boundary value at ξ_Γ by u_Γ. The ξ_Γ satisfies the scalar equation $\psi(\xi_\Gamma, y_j) = 0$ and can be found by a root-finding algorithm such as the secant method.

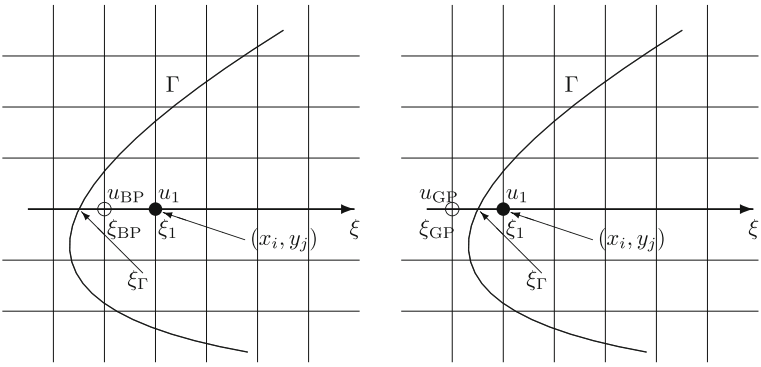

Fig. 3 Enforcing Dirichlet boundary conditions by a line by line approach using interior boundary points (left) or exterior ghost points (right)

Let u_Γ be the value of the boundary condition at (ξ_Γ, y_j). We introduce an interpolating polynomial

$$\mathcal{I}_P u(\xi) = u_\Gamma g_\Gamma(\xi) + u_1 g_1(\xi), \tag{4}$$

where g_Γ and g_1 are the Lagrange polynomials

$$g_\Gamma(\xi) = \frac{\xi - \xi_1}{\xi_\Gamma - \xi_1}, \; g_1(\xi) = \frac{\xi - \xi_\Gamma}{\xi_1 - \xi_\Gamma}. \tag{5}$$

Then, the value of the solution at the boundary point u_{BP} can be approximated to second order accuracy by evaluating the interpolant

$$u_{BP} = \mathcal{I}_P u(\xi_{BP}) = u_\Gamma g_\Gamma(\xi_{BP}) + u_1 g_1(\xi_{BP}) = u_\Gamma \frac{\xi_{BP} - \xi_1}{\xi_\Gamma - \xi_1} - u_1 \frac{\xi_{BP} - \xi_\Gamma}{\xi_\Gamma - \xi_1}. \tag{6}$$

The placement of the boundary point inside the boundary is the subtle yet important distinction from previous methods like those in [16, 20–22]. In previous work, the point is placed outside (and is usually referred to as a ghost point), see the right image of Fig. 3. Then the linear interpolant will contain a factor

$$\frac{\xi_{GP} - \xi_1}{\xi_\Gamma - \xi_1}$$

which can be arbitrarily large when ξ_Γ is close to ξ_1. This causes small-cell stiffness or numerical overflow in the assembly process of the system of equations.

For the Poisson equation the boundary interface can be moved to the interior points if $\frac{\xi_{GP} - \xi_1}{\xi_\Gamma - \xi_1} <$ threshold $\approx O(h)$, [16], however, this will introduce an eigenvalue that scale as $O(\frac{1}{h})$. Such a large eigenvalue will lead to a very restrictive time step for the wave equation.

In contrast, for the approach suggested above, $\xi_\Gamma \leq \xi_{BP} < \xi_1 = \xi_{BP} + h$, and thus

$$|g_\Gamma(\xi_{BP})| = \left| \frac{\xi_{BP} - \xi_1}{\xi_\Gamma - \xi_1} \right| \leq 1$$
$$\text{and } |g_1(\xi_{BP})| = \left| \frac{\xi_{BP} - \xi_\Gamma}{\xi_\Gamma - \xi_1} \right| \leq 1.$$

Table 1 Comparison between the Kreiss/Gibou and our approach to determine values near the boundary and the finite difference approximation to the second derivative in the first available interior point

γ	10 (−5)	10 (−4)	10 (−3)	10 (−2)	10 (−1)	0.9
u_{GP} [Kreiss/Gibou]	7.18 (−1)	7.18 (−1)	7.17 (−1)	7.06 (−1)	6.03 (−1)	3.92 (−2)
u_{BP} [Our method]	3.68 (−6)	3.68 (−5)	3.68 (−4)	3.66 (−3)	3.45 (−2)	1.91 (−1)
$D_+D_-u_1$ [Kreiss/Gibou]	6.32 (−1)	6.32 (−1)	6.31 (−1)	6.21 (−1)	5.25 (−1)	4.20 (−3)
$D_+D_-u_2$ [Our method]	3.17 (−2)	3.17 (−2)	3.20 (−2)	3.51 (−2)	6.32 (−2)	2.03 (−1)

Clearly the errors in the Kreiss/Gibou approach saturate quickly for small values of γ while the new approach is more robustly accurate over the entire range of values of γ. See the text for details

Substituting the value of u_{BP} into the central difference approximation for $(\beta u_x)_x$

$$\frac{\beta_{i-\frac{1}{2},j}u_{BP} - (\beta_{i-\frac{1}{2},j} + \beta_{i+\frac{1}{2},j})u_{ij} + \beta_{i+\frac{1}{2},j}u_{i+1,j}}{h^2},$$

we have

$$\frac{1}{h^2}\Big((-1 + g_1(\xi_{BP}))\beta_{i-\frac{1}{2},j}u_{ij} - \beta_{i+\frac{1}{2},j}(u_{ij} - u_{i+1,j}) + g_\Gamma(\xi_{BP})u_\Gamma\beta_{i-\frac{1}{2},j}\Big).$$

Because only the diagonal element is modified and $|g_1(\xi_{BP})| \leq 1$, the resulting linear system is still symmetric and diagonally dominant with correct sign. As a result, the SPD structure of the discrete Laplacian operator is preserved, and the method also avoids small cell stiffness.

Consider a one-dimensional example with the exact solution $u(x) = e^{-x/h}$ with a boundary at $x = 0$ and with a grid point just outside the boundary at $x_0 = (\gamma - 1)h$ and gridpoints $x_1 = \gamma h$, $x_2 = (1 + \gamma)h$, ... inside. Using the Kreiss / Gibou approach we would then find the exterior ghost-point value $u_0 = \frac{u(0)+(\gamma-1)u_1}{\gamma}$ and using our approach we would find the interior boundary point value $u_1 = \frac{u(0)+\gamma u_2}{1+\gamma}$. For illustration purposes we consider the case $h = 1$ and evaluate the errors in these values as well as in the approximation to the second derivative at the points x_1 and x_2 respectively. Numerical errors for various values of γ corresponding to different numerical approaches are displayed in Table 1. It is clear that the errors in the Kreiss / Gibou approach saturate very quickly for small values of γ while our new approach is more robustly accurate over the entire range of values of γ.

2.2 Radial Basis Function (RBF) Interpolation

Unfortunately there are some cases when the line-by-line approach cannot be used. For example, when the geometry is non-convex (there is an inward pointing smooth corner), as Fig. 4, it can happen that the intersection between the grid line and the boundary does not exist, or it is far away. In Fig. 4, the stencil (3) requires that the leftmost interior boundary point is determined by the interpolant in the x-direction but the intersection with the boundary along the grid-line may be far away and would result in an inaccurate approximation. Of course, the value at the boundary point can be specified by interpolating along the y-direction. However, this strategy would result in a non-diagonal modification of the system matrix and break its symmetry, so we propose an alternative approach.

For geometries where the interior boundary point cannot be accurately determined by line-by-line interpolation, we instead use the radial basis function (RBF) interpolation. To do this we find two suitable distinct points on the boundary of Ω, and utilize these two points and the

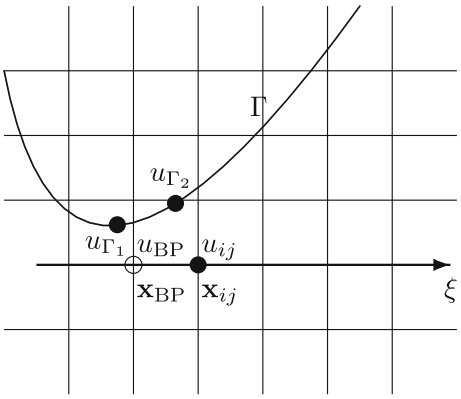

Fig. 4 An illustration of the points used to construct the RBF interpolant for evaluating u_{BP}

interior computational point to interpolate at the interior boundary point with a polynomial augmented RBF interpolant [42]. In this section, we first present the associated RBF-based interpolation and then discuss how to choose the two distinct points on the boundary.

Without loss of generality, we consider the case presented in Fig. 4. Let $\mathbf{x}_{ij} = (x_i, y_j)$ be the computational point where we want the approximation to the Laplacian operator. Let $\mathbf{x}_{\mathrm{BP}} = (x_{\mathrm{BP}}, y_{\mathrm{BP}})$ be the boundary point needed in the stencil in the x-direction, and let $\mathbf{x}_{\Gamma_1}$ and $\mathbf{x}_{\Gamma_2}$ be the two points on the boundary that we have selected. At these points, the boundary conditions are u_{Γ_1} and u_{Γ_2} respectively.

We use the following RBF and linear polynomial augmentation to interpolate at $\mathbf{x}_{\mathrm{BP}}$:

$$\begin{aligned}\mathcal{I}_{\mathrm{RBF}}u(\mathbf{x}) = &\lambda_{ij}\phi(||\mathbf{x}-\mathbf{x}_{ij}||) + \lambda_{\Gamma_1}\phi(||\mathbf{x}-\mathbf{x}_{\Gamma_1}||)\\ &+ \lambda_{\Gamma_2}\phi(||\mathbf{x}-\mathbf{x}_{\Gamma_2}||) + \mu_1 + \mu_2 x + \mu_3 y,\end{aligned} \tag{7}$$

where $\mathbf{x} = (x, y)$, $||\cdot||$ is the standard l_2 norm and $\phi(\cdot)$ is a radial basis function. The linear polynomial augumentation is required to obtain second order accuracy [3, 12]. The coefficients $\boldsymbol{\lambda} = (\lambda_{ij}, \lambda_{\Gamma_1}, \lambda_{\Gamma_2})^T$ and $\boldsymbol{\mu} = (\mu_1, \mu_2, \mu_3)^T$ are determined by solving the linear system

$$B\begin{pmatrix}\boldsymbol{\lambda}\\ \boldsymbol{\mu}\end{pmatrix} = \begin{pmatrix}A & \Pi^T\\ \Pi & 0\end{pmatrix}\begin{pmatrix}\boldsymbol{\lambda}\\ \boldsymbol{\mu}\end{pmatrix} = \begin{pmatrix}\mathbf{u}\\ \mathbf{0}\end{pmatrix}. \tag{8}$$

Here, $\mathbf{u} = (u_{ij}, u_{\Gamma_1}, u_{\Gamma_2})$,

$$A = \begin{pmatrix}\phi(0) & \phi(||\mathbf{x}_{ij}-\mathbf{x}_{\Gamma_1}||) & \phi(||\mathbf{x}_{ij}-\mathbf{x}_{\Gamma_2}||)\\ \phi(||\mathbf{x}_{\Gamma_1}-\mathbf{x}_{ij}||) & \phi(0) & \phi(||\mathbf{x}_{\Gamma_1}-\mathbf{x}_{\Gamma_2}||)\\ \phi(||\mathbf{x}_{\Gamma_2}-\mathbf{x}_{ij}||) & \phi(||\mathbf{x}_{\Gamma_2}-\mathbf{x}_{\Gamma_1}||) & \phi(0)\end{pmatrix} \quad \text{and} \quad \Pi = \begin{pmatrix}1 & 1 & 1\\ x_{ij} & x_{\Gamma_1} & x_{\Gamma_2}\\ y_{ij} & y_{\Gamma_1} & y_{\Gamma_2}\end{pmatrix}.$$

The purpose of the last equation $\Pi\boldsymbol{\lambda} = \mathbf{0}$ is to minimize the far-field growth [13]. The value at the boundary point $\mathbf{x}_{\mathrm{BP}}$ is then

$$u_{\mathrm{BP}} = \mathcal{I}_{\mathrm{RBF}}u(\mathbf{x}_{\mathrm{BP}}) = \left(\boldsymbol{\phi}_{\mathrm{BP}}^T, \mathbf{p}_{\mathrm{BP}}^T\right)B^{-1}\begin{pmatrix}\mathbf{u}\\ \mathbf{0}\end{pmatrix} \tag{9}$$

with $\boldsymbol{\phi}_{\mathrm{BP}} = \left(\phi(||\mathbf{x}_{\mathrm{BP}}-\mathbf{x}_{ij}||), \phi(||\mathbf{x}_{\mathrm{BP}}-\mathbf{x}_{\Gamma_1}||), \phi(||\mathbf{x}_{\mathrm{BP}}-\mathbf{x}_{\Gamma_2}||)\right)^T$ and $\mathbf{p}_{\mathrm{BP}} = (1, x_{\mathrm{BP}}, y_{\mathrm{BP}})^T$. Then the central difference approximation for $\partial_x(\beta\partial_x u)_{ij}$ becomes

$$\frac{1}{h^2}\Big((c_1 u_{ij} + s_{\mathrm{BC}})\beta_{i-\frac{1}{2},j} - (\beta_{i-\frac{1}{2},j} + \beta_{i+\frac{1}{2},j})u_{ij} + u_{i+1,j}\Big), \tag{10}$$

where $c_1 = \left(\boldsymbol{\phi}_{\mathrm{BP}}^T, \mathbf{p}_{\mathrm{BP}}^T\right) B^{-1}[:, 1]$ and $s_{\mathrm{BC}} = \left(\boldsymbol{\phi}_{\mathrm{BP}}^T, \mathbf{p}_{\mathrm{BP}}^T\right) B^{-1}[:, 2:3](u_{\Gamma_1}, u_{\Gamma_2})^T$ with $B^{-1}[:, 1]$ being the first column of B^{-1} and $B^{-1}[:, 2:3]$ being the matrix consisting of the second and the third column of B^{-1}. The only interior computational point involved in (9) is $\mathbf{x}_{ij}$, hence only the diagonal elements of the discrete Laplacian operator are modified and the symmetry is preserved. The primary computational cost associated with imposing boundary conditions is the inversion of the 6 by 6 matrix B during the assembly of the discrete Laplacian operator. However, this cost does not significantly impact the overall computational efficiency.

We now describe how the two points $\mathbf{x}_{\Gamma_i}$ $(i = 1, 2)$ on the boundary are determined (this is also described in Algorithm 1.) Our procedure is simple. If the points closest to $\mathbf{x}_{\mathrm{BP}}$ and $\mathbf{x}_{ij}$ on the boundary are distinguishable, we choose these two points. Otherwise, we choose the point closest to $\mathbf{x}_{\mathrm{BP}}$ as $\mathbf{x}_{\Gamma_1}$ and pick $\mathbf{x}_{\Gamma_2}$ such that the angle between $\overrightarrow{\mathbf{x}_{\mathrm{BP}}\mathbf{x}_{\Gamma_1}}$ and $\overrightarrow{\mathbf{x}_{\mathrm{BP}}\mathbf{x}_{\Gamma_2}}$ is $\frac{\pi}{4}$. These two cases are geometrically demonstrated in Fig. 5. The rotation angle $\frac{\pi}{4}$ consistently yields good performance in all tested numerical examples. If the rotated line does not intersect with the boundary, one can rotate with a smaller angle or refine the mesh to better resolve the geometry.

Algorithm 1: Given the computational point $\mathbf{x}_{ij}$, its neighboring boundary point $\mathbf{x}_{\mathrm{BP}}$ and a pre-selected tolerance ϵ, find the interpolation points $\mathbf{x}_{\Gamma_1}$ and $\mathbf{x}_{\Gamma_2}$.

Find the point $\mathbf{x}_{\Gamma_1}$ and the $\mathbf{x}_{\Gamma_2}^{(0)}$ such that

$$\mathbf{x}_{\Gamma_1} = \arg\min_{\mathbf{x}\in\Gamma_l} ||\mathbf{x} - \mathbf{x}_{\mathrm{BP}}|| \quad \text{and} \quad \mathbf{x}_{\Gamma_2}^{(0)} = \arg\min_{\mathbf{x}\in\Gamma_l} ||\mathbf{x} - \mathbf{x}_{ij}||. \tag{11}$$

If

$$||\mathbf{x}_{\Gamma_1} - \mathbf{x}_{\Gamma_2}^{(0)}|| > \epsilon h, \tag{12}$$

then $\mathbf{x}_{\Gamma_2} = \mathbf{x}_{\Gamma_2}^{(0)}$ (see the left picture in Fig. 5).
Otherwise, rotate the line determined by $\mathbf{x}_{\Gamma_1}$ and $\mathbf{x}_{\mathrm{BP}}$ counterclockwise by $\frac{\pi}{4}$ and find the
intersection of this line with Γ. Choose the intersection point as $\mathbf{x}_{\Gamma_2}$ (see the right picture in Fig. 5).

2.3 Choice of Interpolation Strategy

We note that the RBF interpolation can always be applied. However, there are two advantages of the line-by-line interpolation over the RBF interpolation. First, it is more straightforward and second, it is possible to prove that it will preserve diagonal dominance. In what follows, if we apply the line-by-line interpolation wherever possible and only use the RBF interpolation when the line-by-line approach breaks down, we say the embedded boundary method is mixed. If the RBF interpolation is applied everywhere, we say the embedded boundary method is RBF-based.

Numerically, we have observed that the condition (inequality (12)) that the two points on the boundary used in the RBF interpolant always are determined to be distinct for $\epsilon = 0.025$ if the mixed EB method is used. However, inequality (12) may not always hold if the RBF-based EB method is used. An often occurring case when this situation may arise is when the closest points to $\mathbf{x}_{\mathrm{BP}}$ and $\mathbf{x}_{ij}$ are the same point (see the right picture of Fig. 5). Then, in the mixed EB method, the line-by-line interpolation is used. We emphasize that although we

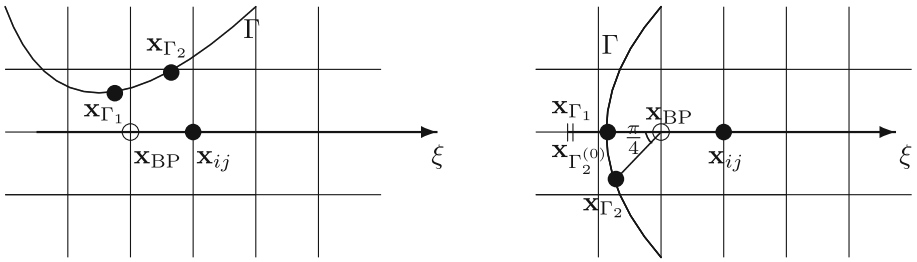

Fig. 5 Geometry demonstration of the two cases in Algorithm 1

have not been able to prove that the mixed EB method or the RBF-based method always lead to a SPD system we have not encountered any numerical examples where the SPD property is lost. Below in Sect. 2.5, we describe a simple algorithm to a-priori check whether the SPD structure is preserved. This algorithm does not require the computation of eigenvalues.

2.4 Extension to 3D

Here we briefly discuss how to extend the proposed method from 2D to 3D.

In 3D, the polynomial part of the augmented RBF interpolation is $\mu_1 + \mu_2 x + \mu_3 y + \mu_4 z$. As a result, four interpolation points are needed to get an invertible matrix B in (8). Hence we need an additional point on the boundary, namely $\mathbf{x}_{\Gamma_3}$. It can be found as follows.

Let the interpolation point inside Ω be $\mathbf{x}_{ijk}$. We first find $\mathbf{x}_{\Gamma_1}$ and $\mathbf{x}_{\Gamma_2}$ following Algorithm 1. Define $\mathbf{x}_{\text{bary}} = (\mathbf{x}_{\Gamma_1} + \mathbf{x}_{\Gamma_2} + \mathbf{x}_{ijk})/3$ as the barycenter of $\mathbf{x}_{\Gamma_1}$, $\mathbf{x}_{\Gamma_2}$, and $\mathbf{x}_{ijk}$, then the normal direction of the plane determined by $\mathbf{x}_{\Gamma_1}$, $\mathbf{x}_{\Gamma_2}$, and $\mathbf{x}_{ijk}$ is

$$\mathbf{n} = (\mathbf{x}_{\Gamma_1} - \mathbf{x}_{ijk}) \times (\mathbf{x}_{\Gamma_2} - \mathbf{x}_{ijk}).$$

We seek $\mathbf{x}_{\Gamma_3}$ along the direction

$$\mathbf{v} = \cos(\theta)\mathbf{n}/||\mathbf{n}|| + \sin(\theta)(\mathbf{x}_{\Gamma_1} - \mathbf{x}_{\text{bary}}) \tag{13}$$

where θ is a pre-selected rotation angle. Specifically, $\mathbf{x}_{\Gamma_3}$ is given by

$$\mathbf{x}_{\Gamma_3} = \mathbf{x}_{\text{bary}} + t_0\mathbf{v}_0 \quad \text{subject to } \mathbf{v}_0 = h\mathbf{v}/||\mathbf{v}|| \quad \text{and} \quad \psi(\mathbf{x}_{\text{bary}} + t_0\mathbf{v}_0) = 0. \tag{14}$$

The above procedure can be seen as rotating the line determined by the vector $\mathbf{x}_{\Gamma_1} - \mathbf{x}_{\text{bary}}$ and finding the intersection of the rotated line and the boundary. In practice we solve the scalar nonlinear equation $\psi(\mathbf{x}_{\text{bary}} + t_0\mathbf{v}_0) = 0$ with Newton's method and zero as the initial guess. In our numerical tests, we set $\theta = \frac{\pi}{6}$, which has been found to be effective for all tested 3D examples. Note that the main focus of the current paper is on the method in 2D and a more systematic study for the extension of the proposed method to 3D, such as how to determine θ, is left for future investigations.

Algorithm 2: Given a computational domain Ω, a Cartesian mesh, source $f(x, y)$, and boundary conditions, compute the numerical solution to (1).

Step 1: Locate the interior computational points and the interior boundary points. Obtain the total number of computational points N_C.
Step 2: Assemble the discrete Laplacian operator $\mathbf{L} \in \mathbb{R}^{N_C \times N_C}$ and the right hand side $\mathbf{b} \in \mathbb{R}^{N_C}$:
apply the difference approximation (3) at the computational points $\mathbf{x}_k^C = (x_{i^{(k)}}, y_{j^{(k)}})$, $1 \le k \le N_C$. If $(x_{i^{(k)}\pm 1}, y_{j^{(k)}\pm 1})$ is a boundary point, temporarily neglect its contribution.
Step 3: Impose the boundary condition to correct $\mathbf{L}$ and $\mathbf{b}$:
If $(x_{i^{(k)}-1}, y_{j^{(k)}})$ is a boundary point:

If $i^{(k)} - 2 \le 0$ or $(x_{i^{(k)}-2}, y_{j^{(k)}})$ is outside Ω, **then** apply the line-by-line interpolation to impose boundary condition and correct $\mathbf{L}_{kk}$ and $\mathbf{b}_k$.

Otherwise, apply the RBF interpolation to impose the boundary condition and correct $\mathbf{L}_{kk}$ and $\mathbf{b}_k$.
Endif
If $(x_{i^{(k)}+1}, y_{j^{(k)}})$ or $(x_{i^{(k)}}, y_{j^{(k)}\pm 1})$ is a boundary point, impose the boundary condition similarly.
If the RBF interpolation was used, use the algorithm in Sect. 2.5 to check whether the SPD structure is preserved.
Step 4: Solve the linear equation $\mathbf{Lu} = \mathbf{b}$ with the conjugate gradient (CG) method and an AMG preconditioner.

2.5 An Algorithm to Check the SPD Structure for Constant β

It is well known that the conjugate gradient method (CG) with the classical algebraic multigrid preconditioner (AMG) [35] is efficient for SPD matrices. The proposed embedded boundary method always results in a symmetric linear system. For many cases, the resulting linear system is also diagonally dominant, which is a sufficient condition for symmetric positive definiteness. However, for certain geometric configurations the matrix may not be diagonally dominant. For these (rarely occurring) cases we present a simple algorithm to a-priori check whether the matrix is positive definite.

To derive conditions guaranteeing that the discrete matrix corresponding to our discretization of the Laplacian is SPD, we need a result for the one dimensional three-point central difference discretization for u_{xx} with the same type of boundary modification as in the embedded boundary methods described above.

Consider a discretization along a grid-line in the x-direction with n grid points. Assume that the boundary conditions on each side have been imposed by modifying the first and last diagonal element in the matrix (and the right hand side vector). Then the resulting matrix

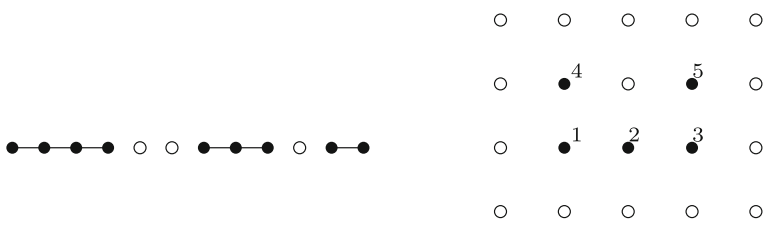

Fig. 6 Left: computational points and non-computational points along a grid line. Here the ordering of the unknowns would be increasing from left to right. Right: lexicographic ordering of the computational points results in a block diagonal discretization matrix (here with block sizes 3,1,1. Solid points: computational points. Empty points: non-computational points

can be written

$$\mathcal{D}^{(n)}(a,b) = \begin{cases} \begin{pmatrix} a \end{pmatrix}, & n = 1, \\ \begin{pmatrix} a & -1 \\ -1 & b \end{pmatrix}, & n = 2, \\ \begin{pmatrix} a & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 2 & -1 \\ 0 & 0 & 0 & \dots & -1 & b \end{pmatrix} \in \mathbb{R}^{n\times n}, & n \geq 3. \end{cases} \tag{15}$$

The following Lemma (whose proof is given in Appendix A) gives conditions on a and b which guarantee that $\mathcal{D}^{(n)}(a,b)$ is SPD.

Lemma 2.1 *The matrix $\mathcal{D}^{(n)}(a,b)$ is symmetric positive definite, if one of the following conditions is satisfied:*

1.
$$n = 1, \text{ and } a > 0. \tag{16}$$

2.
$$n = 2,\ a > 0, \text{ and } ab > 1. \tag{17}$$

3.
$$n \geq 3, \quad a > \frac{n-2}{n-1}, \quad b > \frac{n-2}{n-1} \quad \textit{and} \quad a > \frac{(n-2)b-(n-3)}{(n-1)b-(n-2)}. \tag{18}$$

As a result of Lemma 2.1, the following corollary gives us two conditions that are more straightforward to check.

Corollary 1 *The matrix $\mathcal{D}^{(n)}(a,b)$ is symmetric positive definite if $a > 1$ and $b > 1$.*

In general there may be holes in the computational domain and the discretization along a grid line is then divided into S segments (see the example in the left picture in Fig. 6). Suppose that the k-th segment contains n_k computational points ordered from left to right, then the

matrices corresponding to the discretization on each grid line segment can be arranged in the following block diagonal form

$$\mathcal{D}^{\text{line}} = \begin{pmatrix} \mathcal{D}^{(n_1)}(a_1, b_1) & & & \\ & \mathcal{D}^{(n_2)}(a_2, b_2) & & \\ & & \ddots & \\ & & & \mathcal{D}^{(n_S)}(a_S, b_S) \end{pmatrix}. \tag{19}$$

The matrix $\mathcal{D}^{\text{line}}$ thus contains the discretization of the second derivative along a single, say, horizontal grid line. As a direct result of Lemma 2.1 and the block structure of (19), we have the following Theorem.

Theorem 1 *The matrix $\mathcal{D}^{\text{line}}$, defined in (19), is symmetric positive definite, if for each of the blocks, $\mathcal{D}^{(n_k)}(a_k, b_k)$, the numbers a_k, b_k, n_k satisfy one of the conditions in Lemma 2.1.*

In higher dimensions there will be many grid lines and many segments but, assuming lexicographic ordering (see the right picture in Fig. 6), the discretization matrix for the second derivative in x will still be block diagonal with each block being a tridiagonal matrix on the form (15). Let this "two dimensional" block diagonal matrix with tridiagonal blocks be called $\mathcal{D}_{xx}$. Similarly let the matrix $\mathcal{D}_{yy}$ be a block diagonal matrix with tridiagonal blocks corresponding to the discretization of the second derivative in y along all grid lines but now with an ordering of the degrees of freedom that is fast in the y-index. Finally let P be the permutation matrix that converts between the fast-in-x (lexicographic) and fast-in-y ordering. Then, using the lexicographic ordering the approximation of the Laplacian is $-\mathcal{L} = \mathcal{D}_{xx} + P^T \mathcal{D}_{yy} P$ and we have the following theorem.

Theorem 2 *Consider a two dimensional embedded boundary discretization on a grid with N_c computational points and resulting in block diagonal matrices $\mathcal{D}_{xx} \in \mathbb{R}^{N_c \times N_c}$ and $\mathcal{D}_{yy} \in \mathbb{R}^{N_c \times N_c}$ in the fast-in-x and fast-in-y orderings, respectively. Assume that all of the one dimensional one segment discretization matrices (in both the x and y direction) satisfy the conditions in Lemma 2.1, then the matrix $-\mathcal{L} = \mathcal{D}_{xx} + P^T \mathcal{D}_{yy} P$ is symmetric positive definite.*

Proof The matrix is manifestly symmetric. Let $v \in \mathbb{R}^{N_C}$ be an arbitrary non-zero vector. With $\mathcal{D}_{xx}$ and $\mathcal{D}_{yy}$ being symmetric positive definite, $v^T \mathcal{D}_{xx} v > 0$ and $v^T P^T \mathcal{D}_{yy} P v > 0$ and so

$$-v^T \mathcal{L} v = v^T \mathcal{D}_{xx} v + v^T P^T \mathcal{D}_{yy} P v > 0 + 0 = 0. \tag{20}$$

□

At the implementation level, when we check the conditions in Lemma 2.1, the width of each 1D segments can be found based on the mask matrix, and the values of a_k and b_k can be computed based on the boundary corrections along the horizontal (or vertical) direction.

3 Numerical Results

We now demonstrate performance of our method through a series of numerical examples including the Poisson equation, the heat equation and the wave equation. Throughout the l_2

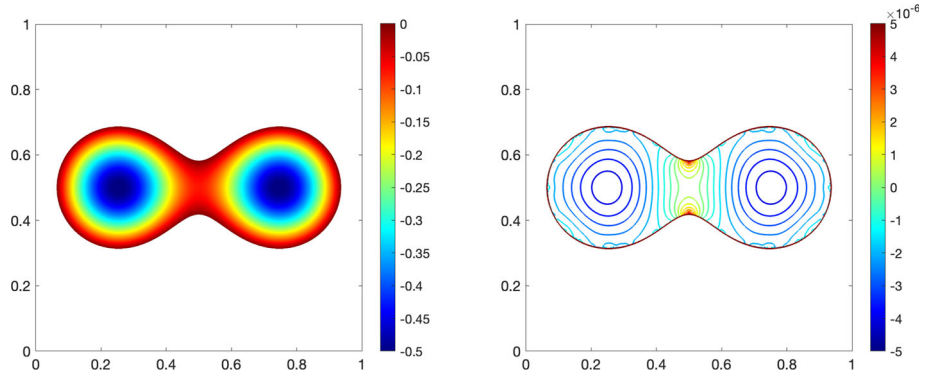

Fig. 7 Poisson's equation in two dimensions, $\nabla \cdot (\beta \nabla u) = f$, with Dirichlet boundary conditions on a glass-shaped domain. Left: Numerical solution with $N = 1280$. Right: The error between the numerical and the exact solution

error and l_∞ error are computed by

$$\begin{aligned} \mathcal{E}_{l_2} &= \sqrt{\sum_{i,j} h^2 \left(u_{ij} - u_{\text{exact}}(x_i, y_j)\right)^2}, \ \mathcal{E}_{l_\infty} \\ &= \max_{i,j} \left|u_{ij} - u_{\text{exact}}(x_i, y_j)\right| \end{aligned} \tag{21}$$

in 2D. Throughout this section, DOF stands for the degrees of freedom. In the RBF interpolation, we choose the polyharmonic spline $\phi(r) = r^3$ as the radial basis function. For the Poisson and the heat equation with implicit time stepper, the linear solver is chosen as the conjugate gradient (CG) method with a classic algebraic multigrid (AMG) preconditioner [35]. Both the V-cycle and W-cycle are considered in the AMG method. The iterative solver is considered to have converged when the relative residual is smaller than 10^{-12}. In all of our numerical experiments, we observe that the resulting discrete Laplacian operator is always SPD.

Our code is implemented in Julia, and we use the AMG preconditioner and CG solver from the open source packages `AlgebraicMultigrid.jl` and `IterativeSolvers.jl`.

3.1 Poisson's Equation in a Non-convex Geometry

We consider a glass-shaped and non-convex geometry determined by the level-set function

$$\psi(x, y) = 0.5 - e^{-20((x-0.25)^2+(y-0.5)^2)} - e^{-20((x-0.75)^2+(y-0.5)^2)}. \tag{22}$$

The Dirichlet boundary condition and the source function are chosen such that $\psi(x, y)$ is an exact solution with the coefficient in (1) to be $\beta(x, y) = -8$. An $(N + 1) \times (N + 1)$ uniform mesh partitioning $[0, 1] \times [0, 1]$ is used. With this non-convex geometry, the RBF interpolation will be activated in the mixed EB method.

The numerical solution and error for $N = 1280$ are presented in Fig. 7. In Fig. 8 the errors for $N = 50$ to $N = 500$ are also displayed. For both the mixed and the RBF-based method, the l_2 and l_∞ error converge as $O(h^2)$. The error of the gradient ∇u converges as $O(h^{1.7})$ in l_2 and $O(h)$ in l_∞. The total number of iterations for convergence for the W-cycle are

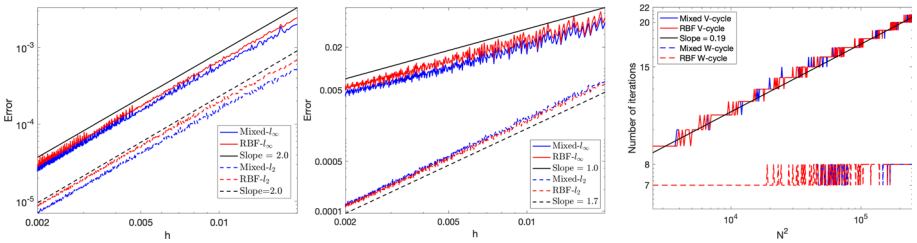

Fig. 8 Convergence check for the AMG preconditioned embedded boundary method on the glass-shapes domain. Left: Numerical errors correspond to different grid size h. Middle: Numerical errors of the gradient for different h. Right: The total number of AMG iterations for convergence for different N^2

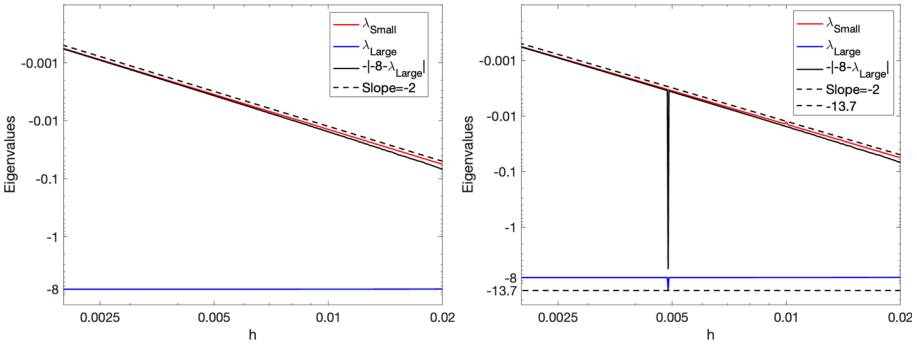

Fig. 9 Glass shaped geometry: the maximum and minimum eigenvalues of the discrete Laplace operator as a function of h. Left: the embedded boundary method using mixed interpolation. Right: the embedded boundary method using RBF-based interpolation

independent of the size of the problem and are always 7 or 8. The number of iterations for the V-cycle scale as $O(DOF^{0.19})$.

We also compute the eigenvalues with the smallest and largest magnitude, λ_{Small} and λ_{Large}, for $50 \leq N \leq 500$, and plot them in Fig. 9. As can be seen they are all negative and as expected λ_{Small} scales approximately as $O(h^{-2})$ while λ_{Large} converges to -8 as the mesh is refined.

3.2 Poisson's Equation on a Tilted Square

We take the following example from [31]. The computational domain is $[-3, 3] \times [-3, 3]$, and Ω is a tilted square determined by the level set function

$$\psi(x, y) = \max\Big(\max\big(\,|\,(\hat{x} - p_x) - (\hat{y} - p_y)\,|\,-1, |\,(\hat{x} - p_x) + (\hat{y} - p_y)\,|\,-1, |\,(\hat{y} - p_y) - (\hat{x} - p_x)\,|\,-1\big)\Big),$$

where $\hat{x}(x, y) = x\cos(\theta\pi) - y\sin(\theta\pi)$ and $\hat{y}(x, y) = x\sin(\theta\pi) + y\cos(\theta\pi)$. We take $p_x = 0.691$ and $p_y = 0.357$ so that the center of the tilted square (p_x, p_y) does not fall exactly on a grid point. We take $\theta = 0.313$ so that the tilted square is not symmetric in the x or y direction. The boundary of the tilted square Ω has a kink. The exact solution for this

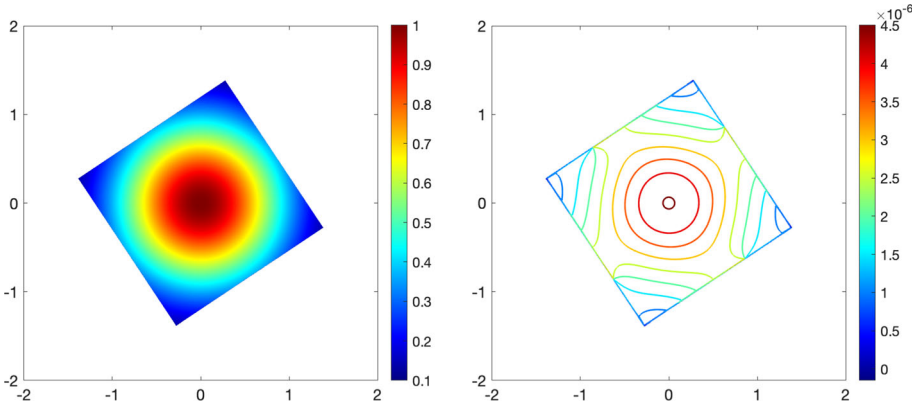

Fig. 10 Poisson's equation in two dimensions, $\nabla \cdot (\beta \nabla u) = f$, with Dirichlet boundary conditions on a tilted-square domain. Left: Numerical solution with $N = 1280$. Right: The contours of the error between the numerical and the exact solution

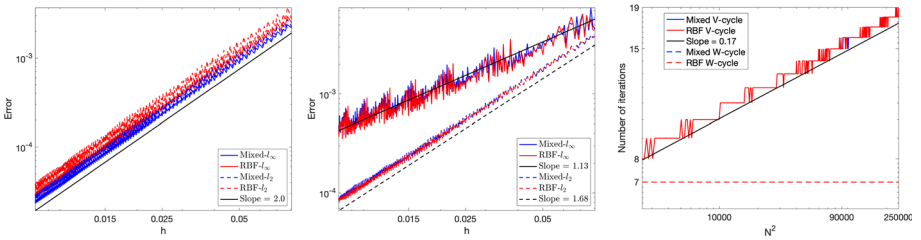

Fig. 11 Convergence check for the AMG preconditioned embedded boundary method on a tilted-square domain. Left: Numerical errors as a function of h. Middle: Numerical errors of the gradient as a function of h. Right: The total number of iterations for convergence for different N^2

example is $u(x, y) = e^{-x^2-y^2}$ and $\beta(x, y) = 1.0$. A $(N + 1) \times (N + 1)$ uniform mesh is used.

The numerical solution and error for $N = 1280$ are presented in Fig. 10. Sweeping from $N = 50$ to $N = 500$, numerical results are presented in Fig. 11. For both the mixed and the RBF-based method, we observe second order accuracy for the solution in both l_∞ and l_2. The error in the gradient ∇u scales as $O(h^{1.13})$ in l_∞ and $O(h^{1.68})$ in l_2. As in the example above, the total number of iteration for convergence stays fixed at 7 when the W-cycle is used and scales as $O(DOF^{0.17})$ when the V-cycle is used.

This geometry can result in stencils that are no longer diagonally dominant. Out of the 451 considered values of N, 25 of them result in a linear system that is not diagonal dominant. For all the system sizes the criteria to check the SPD property from Sect. 2.5 guarantees that the matrix is SPD.

3.3 Comparison with Finite Element Method

As mentioned in the introduction an advantage of using an embedded boundary method is that the mesh generation is local and performed using geometry objects that are one dimension lower than for an unstructured mesh generator. The locality also helps with parallelization and load balancing, but this is not considered here. This said, once the mesh is generated

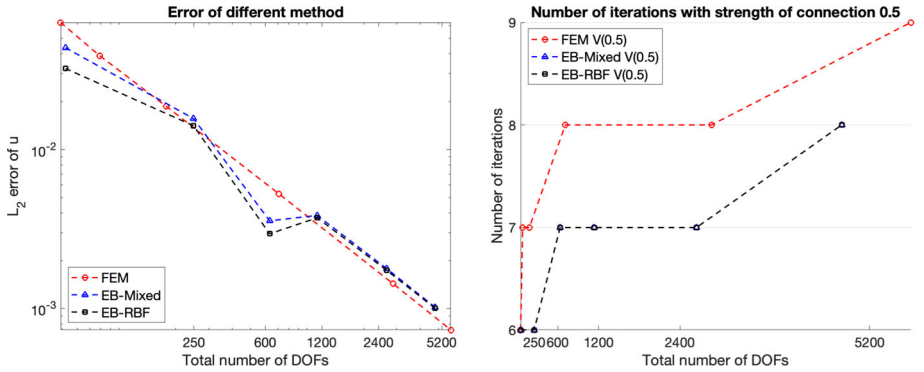

Fig. 12 Comparison of FEM and EB method for the Poisson equation on the unit disk. Left: L_2 error as a function of number of degrees of freedom. Right: total number of AMG iterations for the convergence of V-cycle AMG-CG solver with the threshold to determine the strength of connection 0.5

the algebraic problems have similar structure and thus we expect similar performance. In this example we compare the computational performance of our method with a conforming piecewise linear Galerkin finite element method (FEM) using unstructured meshes. The finite element method is implemented with Julia library `Gridap.jl` [2]. We solve $-\Delta u = 4$ on the unit disk. The exact solution is $u(x, y) = x^2 + y^2 - 1$. For the embedded boundary method, we partition the computational domain $[-1.1, 1.1]^2$ with a uniform mesh. For the finite element method, we use `Gmsh` [14] to generate a series of unstructured meshes by setting the global mesh size factor as 1, $\frac{3}{4}$, $\frac{1}{2}$, $\frac{1}{4}$, $\frac{1}{8}$, and $\frac{1}{16}$.

The comparison of the two methods are done in two ways: the L^2 error and the number of iterations for the convergence of AMG-CG solver, is shown in Fig. 12. We observe that, with roughly the same number of DOFs, errors given by the EB-Mixed, EB-RBF, and the FEM are comparable. With 0.5 as the threshold to determine the strength of connection, the V-cycle AMG-CG solver converges fast for both EB and FEM methods and needs slightly less number of AMG iterations for the EB method. We have also done similar comparisons for other grids and geometries and found that overall, the performance of the EB method and the FEM method with unstructured mesh is comparable.

3.4 Geometry Determined by Parametric Curves

In our previous examples, Ω is determined by a level set function. Here we consider a case when it is determined by a parametric curve:

$$\Gamma = \{(x, y) = (x(\theta), y(\theta)), \ \theta \in [\theta_L, \theta_R]\}. \tag{23}$$

We define the level set function $\psi(x, y)$ as the signed distance function $\psi_{\text{SD}}(x, y)$. Let the closet point on the boundary interface Γ to $\mathbf{x} = (x, y)$ be $\mathbf{x_n} = (x_\mathbf{n}, y_\mathbf{n})$. The amplitude of $\psi_{\text{SD}}(x, y)$ is $||\mathbf{x} - \mathbf{x_n}||$. The sign of $\psi_{\text{SD}}(x, y)$ is determined by the cross-product of the normal vector $\mathbf{n} = \overrightarrow{\mathbf{x}\mathbf{x}_\mathbf{n}}$ and the tangent vector $\boldsymbol{\tau}$ of Γ passing $\mathbf{x_n}$, and $\psi_{\text{SD}}(x, y)$ is negative for $\mathbf{x} = (x, y)$ inside Ω.

To find the closet point to $\mathbf{x} = (x, y)$ on the boundary Γ, a good initial guess is needed. We uniformly partition $[\theta_L, \theta_R]$ with N_{guess} points, and use the point corresponding to $\theta_{\text{guess}} = \arg\min_{1 \le k \le N_{\text{guess}}} \{||(x, y) - (x(\theta_k), y(\theta_k))||\}$ as the initial guess.

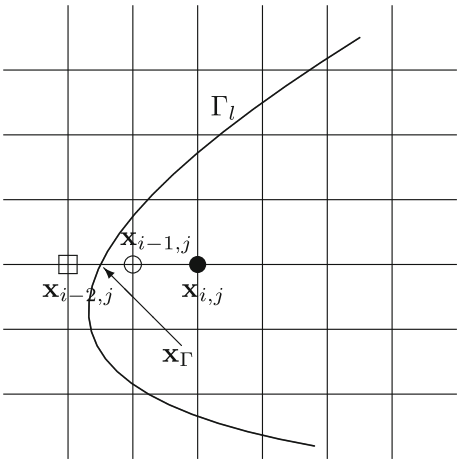

Fig. 13 Two grid points $\mathbf{x}_{i-1,j}$ and $\mathbf{x}_{i-2,j}$ which border $\mathbf{x}_\Gamma$. An approximation of the distance between a boundary point $\mathbf{x}_{i-1,j}$ and the intersection point $\mathbf{x}_\Gamma$ of the interface with the line y_j by the linear interpolation of the level set function

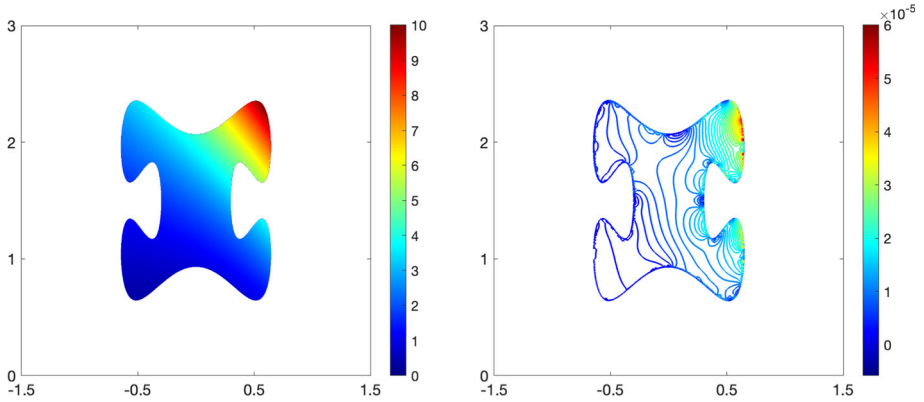

Fig. 14 Poisson's equation in two dimensions, $\nabla \cdot (\beta \nabla u) = f$, with Dirichlet boundary conditions on a bone-shape domain. Left: Numerical solution with $N = 1280$. Right: The contours of the error between the numerical and the exact solution

When the line-by-line interpolation is applied, we need to approximate the horizontal or vertical distance from the interior boundary points to the boundary interface Γ. One can compute this distance exactly, but here we re-use the infrastructure for the levelset description of the geometry and instead use a second order approximation of the desired distance. Take the case in Fig. 13 as an example, $\mathbf{x}_{i,j}$ is a computational point, $\mathbf{x}_{i-1,j}$ is a boundary point and $\mathbf{x}_{i-2,j}$ is outside Ω. $\mathbf{x}_\Gamma$ lies on the boundary interface Γ. Following [8], $||\mathbf{x}_{i-1,j} - \mathbf{x}_\Gamma||$ can be approximated as:

$$||\mathbf{x}_{i-1,j} - \mathbf{x}_\Gamma|| = \frac{\psi_{\mathrm{SD}}(x_{i-1}, y_j)}{\psi_{\mathrm{SD}}(x_{i-1}, y_j) - \psi_{\mathrm{SD}}(x_{i-2}, y_j)} h + O(h^2). \tag{24}$$

3.4.1 Poisson's Equation on a Bone-Shaped Geometry

We take the example from [16, 24] and solve Poisson's equation on a bone-shaped irregular domain in the computational domain $[-1.5, 1.5] \times [0, 3]$. The exact solution is

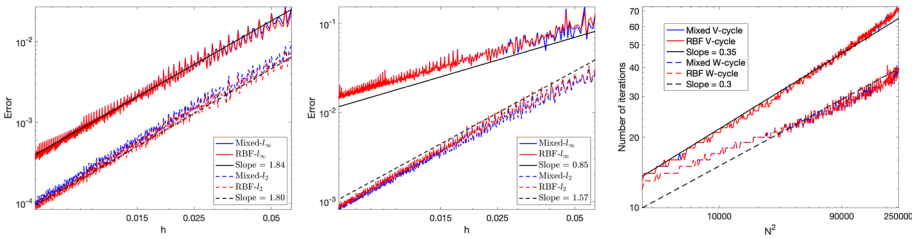

Fig. 15 Convergence check for the AMG preconditioned embedded boundary method for a bone-shaped domain. Left: Numerical errors as a function of h. Middle: Numerical errors of the gradient as a function of h. Right: The total number of iterations for convergence for different N^2

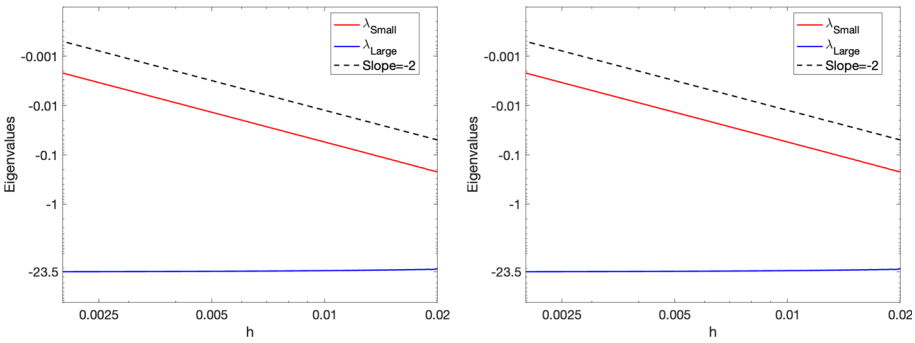

Fig. 16 Bone-shaped geometry: the maximum and minimum eigenvalues of the discrete Laplace operator as a function of h. Left: the embedded boundary method using mixed interpolation. Right: the embedded boundary method using RBF-based interpolation

$u = e^x(x^2\sin(y) + y^2)$ and $\beta = 2 + \sin(xy)$. The boundary Γ is parameterized by

$$\Big(x(\theta), y(\theta)\Big) = \Big(0.6\cos(\theta) - 0.3\cos(3\theta), 1.5 + 0.7\sin(\theta) - 0.07\sin(3\theta) + 0.2\sin(7\theta)\Big)$$

with $\theta \in [0, 2\pi]$. An $(N+1) \times (N+1)$ uniform mesh partitioning $[-2, 2] \times [-1, 3]$ is used.

The numerical solution for $N = 1280$ and its difference from the exact solution are presented in Fig. 14. Sweeping from $N = 50$ to $N = 500$, numerical results are presented in Fig. 15. For both the mixed and the RBF-based EB methods, we observe $O(h^{1.84})$ error in l_∞ and $O(h^{1.80})$ error in l_2 for u. The l_∞ error of the gradient ∇u scale as $O(h^{0.85})$ and the l_2 error scale as $O(h^{1.57})$. The total number of iterations needed by the W-cycle is smaller than the V-cycle. The former scales as $O(DOF^{0.3})$ and the later scales as $O(DOF^{0.35})$. Note that the cost of one W-cycle is larger than one V-cycle. Except for 4 out of the 451 considered discretization sizes, the resulting linear system is diagonally dominant, and the proposed criteria to check the SPD structure is never violated. The eigenvalues with the smallest and largest magnitude λ_{Small} and λ_{Large} are plotted in Fig. 16. They are all negative, and λ_{Small} scales approximately as $O(h^{-2})$. Moreover, $|\lambda_{\text{Large}}|$ is smaller than 24, which is the eigenvalue with largest magnitude of the problem with periodic boundary condition and conductivity coefficient $\widetilde{\beta} = \max(|\beta(x, y|) = 3$.

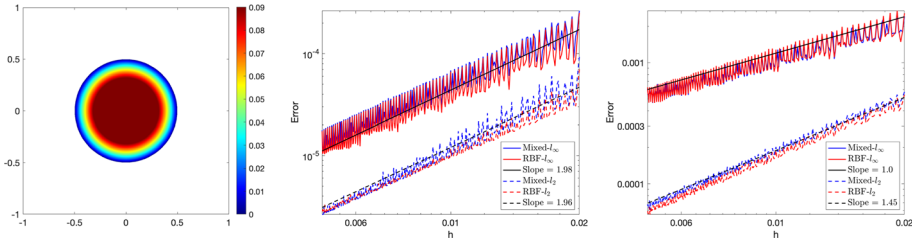

Fig. 17 The heat equation in two dimensions, $u_t = \nabla \cdot (\beta \nabla u) + f$, with Dirichlet boundary conditions on a disk-shaped domain. Left: Numerical solution using $N = 1000$. Middle: Numerical error as a function of h at the final time $T = 0.5$. Right: Numerical errors of the gradient as a function of h at the final time $T = 0.5$

3.5 Heat Equation

In this example, we consider the heat equation with Dirichlet boundary conditions

$$u_t = \nabla \cdot (\beta \nabla u) + f, \quad (x, y) \in \Omega, \tag{25a}$$

$$u(x, y, t) = u_{\mathcal{D}}(x, y), \quad (x, y) \in \Gamma \quad \text{and} \quad u(x, y, 0) = u_{\mathcal{I}}(x, y). \tag{25b}$$

We consider $\beta(x, y) = 0.25 - x^2 - y^2$, and impose a homogenous Dirichlet boundary condition. The geometry Ω is a disk determined by the level set function $\psi(x, y) = 0.25 - x^2 - y^2$. The source $f(x, y, t)$ and the initial condition are chosen such that $u(x, y, t) = e^{-t}(x^2 + y^2 - 0.25)$. The computational domain is $[-1, 1] \times [-1, 1]$ partitioned by a $(N + 1) \times (N + 1)$ uniform mesh.

We use our method as the spatial discretization and the Crank-Nicolson method as the time integrator with a time step size $\Delta t = h$. In this example only the V-cycle is used in the AMG. When inverting the linear system, we use the data from last step as initial guess. We solve the problem for time $t \in [0, 0.5]$. The numerical solution with $N = 1000$ at $t = 0.5$ is presented in Fig. 17. The l_2 and l_∞ errors in u at $t = 0.5$ are converging as $O(h^2)$, the error of the gradient ∇u at $t = 0.5$ converges as $O(h)$ in l_∞ and as $O(h^{1.45})$ in l_2. On average, we need 2 iterations for convergence per time step.

3.6 The Wave Equation

In this example we solve the wave equation with homogenous Dirichlet boundary conditions

$$\frac{\partial^2 u(x,y,t)}{\partial t^2} = \nabla^2 u(x, y, t), \ (x, y) \in \Omega \equiv \{x^2 + y^2 < 1\}, \quad t > 0,$$

$$u(x, y, t) = 0, \qquad (x, y) \in \Gamma \equiv \{x^2 + y^2 = 1\}.$$

The initial data is chosen so that the solution is a standing mode

$$u_{mn}(r, \theta, t) = J_m(r\kappa_{mn}) \cos m\theta \ \cos \kappa_{mn} t. \tag{26}$$

Here $J_m(z)$ is the Bessel function of the first kind of order m, $(m = 0, 1, \ldots)$ and κ_{mn} is the nth zero of J_m. In this problem we set $m = n = 7$. Then $\kappa_{77} = 31.4227941922$ and the period of the solution is $\frac{2\pi}{\kappa_{77}} = 0.1999562886971$.

To discretize in time we use the so-called θ-scheme (see e.g. [4])

$$u^{n+1} - \theta \Delta t^2 \nabla^2 u^{n+1} = 2u^n + (1 - 2\theta)\Delta t^2 u^n - u^{n-1} + \theta \Delta t^2 \nabla^2 u^{n-1}. \tag{27}$$

Table 2 The table displays max-errors, along with computed rates of convergence (using two subsequent refinement levels) and average number of AMG-CG iterations per time step, at the final time T

h	$\theta = 0$	p	$\theta = \frac{1}{2}$	p	iter	$\theta = \frac{1}{4}$	p	iter	$\theta = \frac{1}{12}$	p	iter
2.2 (−2)	3.6 (−2)	∗	2.0 (−1)	∗	6.0	5.7 (−1)	∗	6.0	5.9(−2)	∗	3.0
1.1 (−2)	9.2 (−3)	2.0	4.8 (−1)	−1.3	6.0	7.6 (−3)	6.2	5.2	2.1 (−2)	1.5	3.0
5.5 (−3)	2.5 (−3)	1.9	6.0 (−2)	3.0	6.0	5.1 (−2)	−2.7	5.0	6.1 (−3)	1.8	3.0
2.75 (−3)	6.7 (−4)	1.9	3.5 (−2)	0.8	6.0	1.6 (−2)	1.7	5.0	1.6 (−3)	1.9	3.0
1.375 (−3)	1.7 (−4)	2.0	9.9 (−3)	1.8	5.0	4.3 (−3)	1.9	5.0	4.1 (−4)	2.0	3.0

The results are for the wave equation example

Here $\theta \geq 0$ is a parameter that can be chosen to obtain different schemes. The values we consider here are $\theta = 0$, which corresponds to the classic explicit leap-frog scheme, $\theta = 1/2, 1/4$ which corresponds to second order unconditionally stable implicit methods, and $\theta = 1/12$ which leads to an implicit method that is fourth order accurate but with a stability constraint on the time step. Comparing the explicit method and the fourth order accurate method, the latter can march with a time step that is roughly 50% larger than the second order explicit method. We discretize the domain $(x, y) \in [-1.1, 1.1]^2$ using 100, 200, 400 and 800 grid points and evolve the numerical solution for 10.2 periods. The mixed EB method is used in the spatial discretization. We use $\Delta t/h = (0.7, 2, 2, 0.85)$ for the schemes corresponding to $\theta = (0, 1/2, 1/4, 1/12)$. The initial data are set using the exact solution. In Table 2 we report max-errors, along with computed rates of convergence (using two subsequent refinement levels) and average number of V-cycle AMG-CG iterations per time step, at the final time. It is clear that for this example there is no benefit to use an unconditionally stable time stepping method as the temporal errors are dominating the total error if $\Delta t > h$. On the other hand, the fourth order method achieves an error that is roughly equivalent with the explicit scheme and may be a suitable alternative for problems with solutions that vary rapidly in time. In Figs. 18 and 19 we display the numerical solution and the error for the four different methods.

3.7 Quantity of Interest Determined by Geometry Parameters

In shape optimization problems, a quantity of interest (QOI) or loss may be determined by the parameters of the geometry. In order to find the optimal geometry, the gradient of the QOI or loss with respect to the parameters needs to be computed. If one wants to use an embedded boundary method as a building block for shape optimization the QOI computed by the method should be smooth with respect to the parameter. To show the potential of our method for these type of problems, we consider two prototype examples.

In the first example, we consider Poisson's equation on an elliptical shape with a fixed area

$$\frac{x^2}{a^2} + \frac{y^2}{1/a^2} = 1.$$

Here, $a \in [0.5, 2]$, $\beta = 1$ and zero Dirichlet boundary conditions are used. The source function is $f(x, y) = 3$. The QOI is the value of u at point $(0, 0)$. In Fig. 20, we present the QOI and compute its derivative with respect to a by the central difference method. Both the QOI and computed derivative are smooth. We observe that $u(0, 0)$ has the minimum at

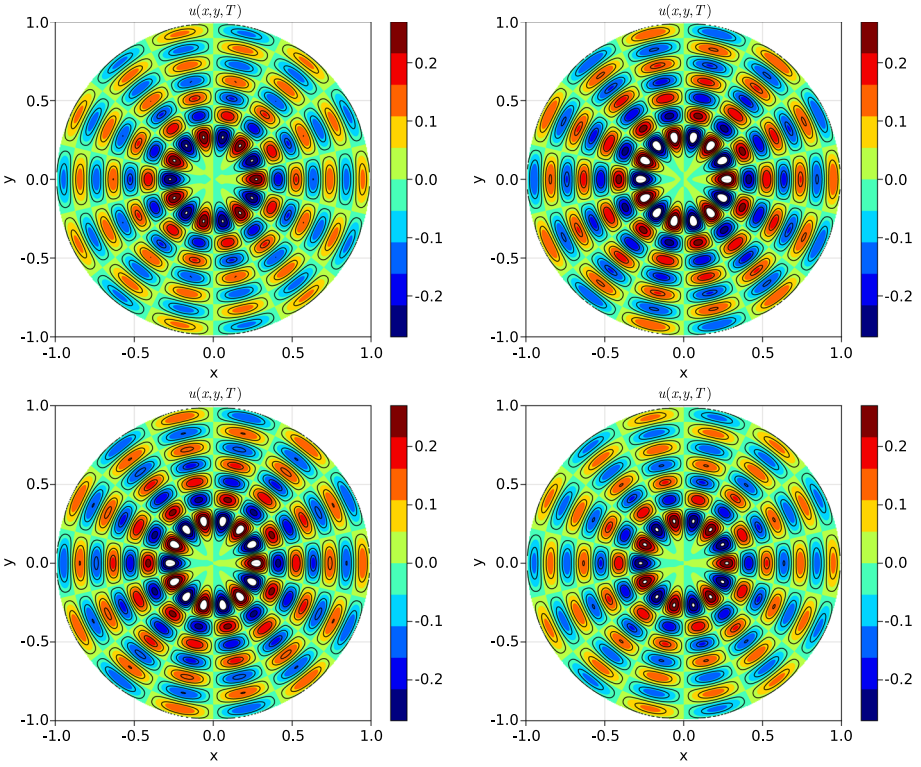

Fig. 18 The wave equation, $u_{tt} = \nabla^2 u$, with zero boundary condition on a disk shaped geometry. Numerical solutions with $\theta = (0, 1/2, 1/4, 1/12)$ correspondingly from left to right, from top to bottom at the final time

$a = 1$, when the elliptical shape becomes a unit circle. We also observe that the value of $u(0, 0)$ corresponding to a and $\frac{1}{a}$ equal to each other.

In the second example, we consider an ellipsoid that is rotated by an angle α and the QOI is the integration of u in the rectangular domain $[-1, 1] \times [-0.5, 0.5]$. The rotated-ellipse-shape geometry is determined by the level-set function

$$\psi(x, y) = \frac{(x\cos(\theta) - y\sin(\theta))^2}{16} + \frac{(x\sin(\theta) + y\cos(\theta))^2}{4} - 1. \tag{28}$$

where $\theta \in [0, 2\pi)$. We consider the Poisson equation with $\beta = 1$ and zero Dirichlet boundary conditions, and the source function is $f(x, y) = 3$. A 501×501 uniform mesh partitioning $[-5, 5] \times [-5, 5]$ is used. In Fig. 20, we present the QOI and compute its derivative with respect to θ by the central difference method. Both of the QOI and the derivative are smooth. Expected symmetry is also observed.

For both examples, the QOI and its derivative computed by our method are smooth, and the underlying geometric symmetry is also respected.

3.8 3D Accuracy Test

We solve Poisson's equation in 3D with the coefficient $\beta(x, y, z) = 1$ on the two geometries displayed in the left column of Fig. 21. The first geometry is the unit ball and the second

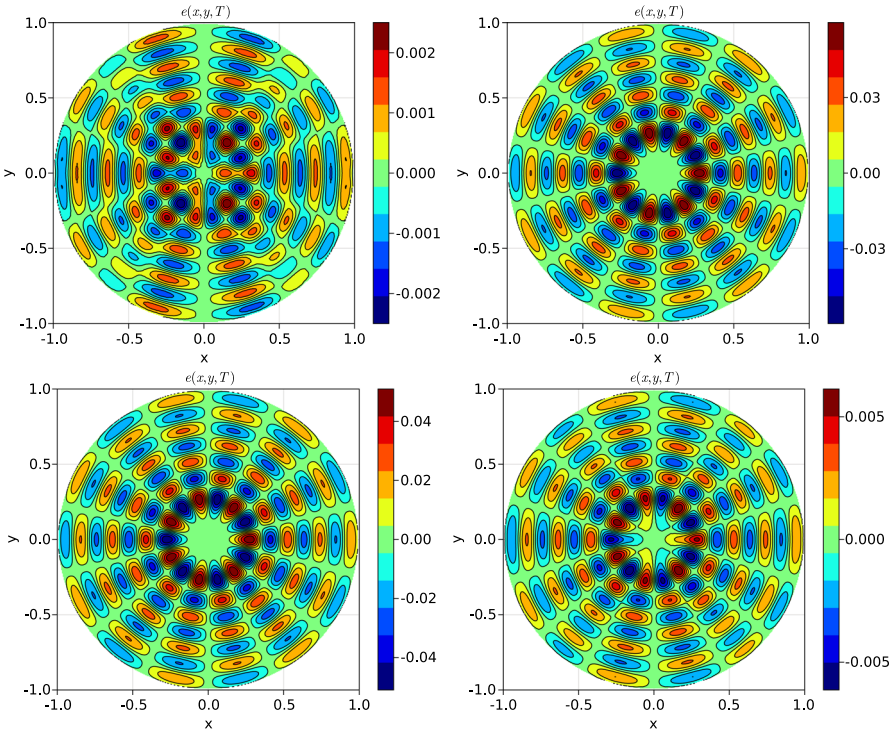

Fig. 19 The wave equation, $u_{tt} = \nabla^2 u$, with zero boundary condition on a disk shaped geometry. Displayed are the errors for $\theta = (0, 1/2, 1/4, 1/12)$ correspondingly from left to right, from top to bottom at the final time

geometry is defined as the interior of the surface defined by the zero level set of

$$\psi_2(x, y, z) = 0.5 - e^{-20((x-0.25)^2+(y-0.5)^2)+(z-0.5)^2)} - e^{-20((x-0.75)^2+(y-0.5)^2+(z-0.5)^2)}. \tag{29}$$

We set the computational domain as $[-1, 1]^3$ for the unit ball and $[0, 1]^3$ for the second (non-convex) geometry. As in two dimensions we use a uniform mesh. We use a V-cycle AMG-CG solver to solve the resulting linear systems. The l_2 errors and number of AMG iterations for convergence are presented in Fig. 21. The l_2 error is computed by

$$\mathcal{E}_{l_2} = \sqrt{\sum_{i,j,k} h^3 (u_{ijk} - u_{\text{exact}}(x_i, y_j, z_k))^2}. \tag{30}$$

The expected second order accuracy is observed and the V-cycle AMG-CG solver converges after a few iterations.

4 Conclusion

In summary, we have developed a universal embedded boundary method to solve the Poisson equation, the heat equation or the wave equation in general two-dimensional domains with

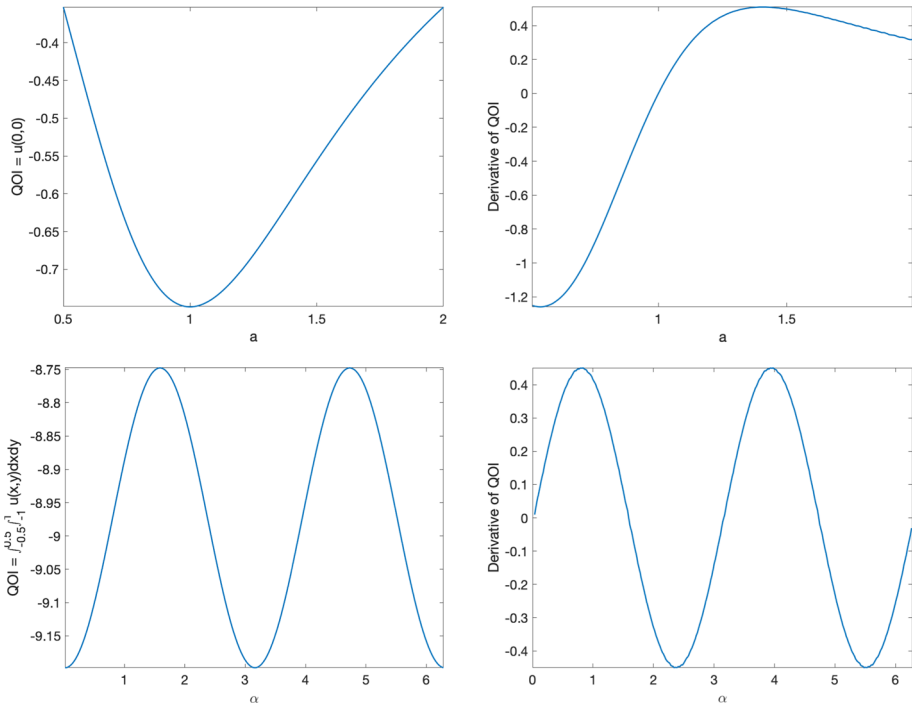

Fig. 20 Computations of QOIs with geometry parameter. Top left: QOI vs a for the first example. Top right: the derivative of QOI with respect to a vs a for the first example. Bottom left: QOI vs the rotation angle α for the second example. Bottom right: the derivative of QOI with respect to α vs the rotation angle α for the second example

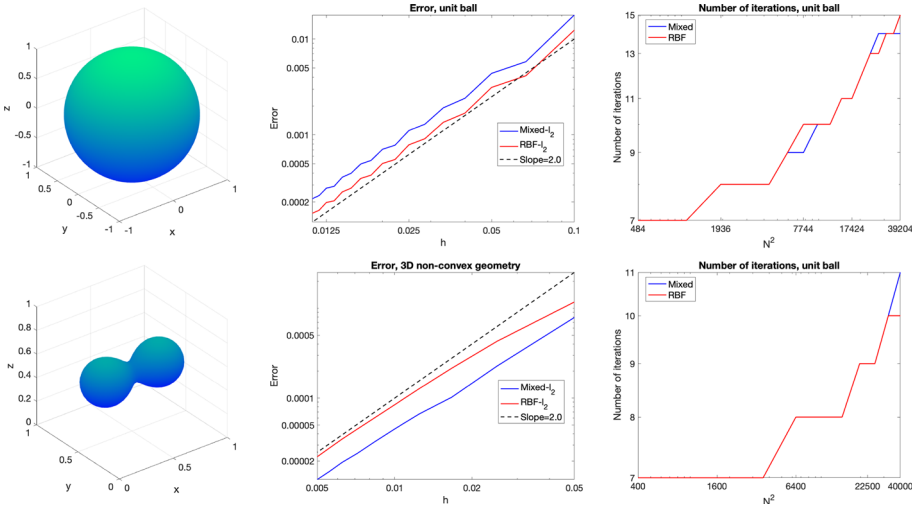

Fig. 21 Geometry, l_2 error of the solution and the number of iterations for convergence in 3D accuracy tests. Top: unit ball. Bottom: the non-convex geometry determined by ψ_2

complex geometries subject to Dirichlet boundary conditions. The two advantages of this method are: (1) by using interior boundary points instead of placing the ghost point outside the computational domain, the small-cell stiffness is avoided. (2) to impose the boundary condition, we apply the line-by-line interpolation or RBF interpolation, which results in a SPD linear system. The SPD structure of the discrete Laplacian operator can be rigorously proved for the convex geometry and is numerically verified for non-convex geometries. A simple criteria to a-priori check the SPD property is also proposed.

An immediate next step is to conduct a more systematic study of extending the proposed method to 3D. A possible future work is to generalize the current method to problems with the Neumann boundary conditions. Other possible extensions include combining the current method with the level set method for moving boundary problems (e.g. Stefan problem) and the generalization of the method to the Navier–Stokes equations, and the Grad-Shafranov equations [25, 33].

Finally, we note that while the method presented here is not able to directly handle singularities due to sharp re-entrant corners, adaptive mesh refinement can be incorporated into EB methods to maintain the accuracy for such a problem, see for example [30].

Author Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Zhichao Peng, Daniel Appelö and Shuang Liu. All authors read and approved the final manuscript.

Funding This material is based upon work supported by the National Science Foundation under Grant No. DMS-1913076, DMS-2210286 & DMS-2208164 (D. Appelö), and in part by an AMS Simons Travel Grant (S. Liu). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation and American Mathematical Society.

Data Availability The participants of this study did not give written consent for their data to be shared publicly, so due to the sensitive nature of the research supporting data is not available.

Declarations

Conflict of interest The authors have no relevant financial or non-financial interests to disclose.

A Proof of Lemma 2.1

For the case $n = 1$ and $n = 2$, the lemma can be verified through direct computations of leading principal minors. We only focus on the case $n \geq 3$.

The matrix $\mathcal{D}^{(n)}(a, b)$ is manifestly symmetric. To prove that $\mathcal{D}^{(n)} \in \mathbb{R}^{n\times n}$ is SPD, we use mathematical induction to prove that its leading principal minors are all positive. When $1 \leq k \leq n-1$, the k-th order leading principal minor of $\mathcal{D}^{(n)} \in \mathbb{R}^{n\times n}$ is

$$Q_k^{(n)}(a) = \det \underbrace{\begin{pmatrix} a & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 2 & -1 \\ 0 & 0 & 0 & \dots & -1 & 2 \end{pmatrix}}_{k\times k}, \qquad 1 \leq k \leq n-1. \tag{31}$$

The n-th order leading principal minor of $\mathcal{D}^{(n)}(a,b) \in \mathbb{R}^{n\times n}$ is

$$P^{(n)}(a,b) = \det \underbrace{\begin{pmatrix} a & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 2 & -1 \\ 0 & 0 & 0 & \dots & -1 & b \end{pmatrix}}_{n\times n}. \tag{32}$$

(1) We first prove that if $a > \frac{n-2}{n-1}$, then the first $n-1$ principal minors of $\mathcal{D}^{(n)} \in \mathbb{R}^{n\times n}$, namely $Q_k^{(n)}(a)$ $(1 \le k \le n-1)$ are all positive.

For $n = 3$, with direct computations, one can check that if $a > \frac{3-2}{3-1} = \frac{1}{2}$ then $Q_1^{(3)}(a)$ and $Q_2^{(3)}(a)$ are positive. Now, for the induction case $n-1$, we assume if $a > \frac{n-3}{n-2}$, then the first $n-2$ principal minors of $\mathcal{D}^{(n-1)}$, namely $Q_k^{(n-1)}(a)$ $(1 \le k \le n-2)$, are all positive. With this induction assumption, we will prove that if $a > \frac{n-2}{n-1}$ then the first $n-1$ principal minors of $\mathcal{D}^{(n)}(a,b)$ are all positive.

The definition in (31) implies that $Q_k^{(n)}(a) = Q_k^{(n-1)}(a)$ when $1 \le k \le n-2$. Moreover, if $a > \frac{n-2}{n-1}$, then $a > \frac{n-3}{n-2}$ and the induction assumption leads to

$$Q_k^{(n)}(a) = Q_k^{(n-1)}(a) > 0, \ 1 \le k \le n-2.$$

Now, we only need to prove that $Q_{n-1}^{(n)}$ is positive:

$$\begin{aligned} Q_{n-1}^{(n)}(a) &= \det \begin{pmatrix} a & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 2 & -1 \\ 0 & 0 & 0 & \dots & -1 & 2 \end{pmatrix} = \det \begin{pmatrix} a & -1 & 0 & \dots & 0 & 0 \\ 0 & 2-\frac{1}{a} & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 2 & -1 \\ 0 & 0 & 0 & \dots & -1 & 2 \end{pmatrix} \\ &= a Q_{n-2}^{(n-1)} \left(2 - \frac{1}{a}\right). \end{aligned} \tag{33}$$

Moreover, if $a > \frac{n-2}{n-1} > 0$, then

$$2 - \frac{1}{a} > 2 - \frac{n-1}{n-2} = \frac{n-3}{n-2}, \tag{34}$$

and together with the induction assumption for $n-1$, we have

$$Q_{n-1}^{(n)}(a) = a Q_{n-2}^{(n-1)} \left(2 - \frac{1}{a}\right) > 0.$$

(2) Finally, we prove if $a > \frac{(n-2)b-(n-3)}{(n-1)b-(n-2)}$ and $b > \frac{n-2}{n-1}$, the last principal minor $P^{(n)}(a,b)$ is positive. We perform an induction proof with respect to n.

For the base case, when $n = 3$, $b > \frac{n-2}{n-1} = \frac{1}{2}$ and $a > \frac{(n-2)b-(n-3)}{(n-1)b-(n-2)} = \frac{b}{2b-1}$. A direct computation shows that $P^{(3)}(a,b)$ is positive. Now, we turn to the induction case, for $n-1$,

assume that

$$b > \frac{n-2}{n-3} \quad \text{and} \quad a > \frac{(n-3)b-(n-4)}{(n-2)b-(n-3)} \tag{35}$$

implies that $P^{(n-1)}(a,b)$ is positive.

We note that $P^{(n)}(a,b)$ can be related to $P^{(n-1)}\left(2-\frac{1}{a},b\right)$ through the rules of determinants as follows:

$$\begin{aligned} P^{(n)}(a,b) &= \det\begin{pmatrix} a & -1 & 0 & \dots & 0 & 0 \\ -1 & 2 & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 2 & -1 \\ 0 & 0 & 0 & \dots & -1 & b \end{pmatrix} = \det\begin{pmatrix} a & -1 & 0 & \dots & 0 & 0 \\ 0 & 2-\frac{1}{a} & -1 & \dots & 0 & 0 \\ 0 & -1 & 2 & \dots & 0 & 0 \\ \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & 2 & -1 \\ 0 & 0 & 0 & \dots & -1 & b \end{pmatrix} \\ &= a P^{(n-1)}\left(2-\frac{1}{a},b\right). \end{aligned} \tag{36}$$

Now, we just need to verify that $a > 0$ and $P^{(n-1)}\left(2-\frac{1}{a},b\right) > 0$ under the assumption that $b > \frac{n-2}{n-1}$ and $a > \frac{(n-2)b-(n-3)}{(n-1)b-(n-2)}$. As $b > \frac{n-2}{n-1}$ and $a > \frac{(n-2)b-(n-3)}{(n-1)b-(n-2)}$, we have

$$\begin{aligned} &(n-1)b-(n-2) > 0, \quad (n-2)b-(n-3) \geq \frac{(n-2)^2-(n-1)(n-3)}{n-1}\frac{1}{n-1} > 0, \\ &0 < \frac{1}{a} < \frac{(n-1)b-(n-2)}{(n-2)b-(n-3)} \quad \text{and} \quad a^* = 2-\frac{1}{a} > 2-\frac{(n-1)b-(n-2)}{(n-2)b-(n-3)} \\ &= \frac{(n-3)b-(n-4)}{(n-2)b-(n-3)}. \end{aligned}$$

Hence, we have $b = \frac{n-2}{n-1} > \frac{n-2}{n-3}$, and $a^* > \frac{(n-3)b-(n-4)}{(n-2)b-(n-3)}$. The induction assumption for the $n-1$ case in (35) is satisfied and $P^{(n-1)}\left(2-\frac{1}{a},b\right) = P^{(n-1)}(a^*,b) > 0$.

References

1. Appelö, D., Petersson, N.A.: A fourth-order embedded boundary method for the wave equation. SIAM J. Sci. Comput. **34**(6), 2982–3008 (2012)
2. Badia, S., Verdugo, F.: Gridap: an extensible finite element toolbox in Julia. J. Open Source Soft. **5**(52), 2520 (2020)
3. Barnett, G.A.: A robust RBF-FD formulation based on polyharmonic splines and polynomials. PhD thesis, University of Colorado Boulder, (2015)
4. Britt, S., Turkel, E., Tsynkov, S.: A high order compact time/space finite difference scheme for the wave equation with variable speed of sound. J. Sci. Comput. **76**(2), 777–811 (2018)
5. Bruno, O.P., Lyon, M.: High-order unconditionally stable FC-AD solvers for general smooth domains I. Basic elements. J. Comput. Phys. **229**(6), 2009–2033 (2010)
6. Chen, H., Min, C., Gibou, F.: A supra-convergent finite difference scheme for the Poisson and heat equations on irregular domains and non-graded adaptive Cartesian grids. J. Sci. Comput. **31**(1), 19–60 (2007)
7. Chen, H., Min, C., Gibou, F.: A numerical scheme for the Stefan problem on adaptive Cartesian grids with supralinear convergence rate. J. Comput. Phys. **228**(16), 5803–5818 (2009)
8. Chen, S., Merriman, B., Osher, S., Smereka, P.: A simple level set method for solving Stefan problems. J. Comput. Phys. **135**(1), 8–29 (1997)
9. Chesshire, G., Henshaw, W.D.: Composite overlapping meshes for the solution of partial-differential equations. J. Comput. Phys. **90**(1), 1–64 (1990)

10. Coco, A., Russo, G.: Finite-difference ghost-point multigrid methods on Cartesian grids for elliptic problems in arbitrary domains. J. Comput. Phys. **241**, 464–501 (2013)
11. Courant, R.: Variational methods for the solution of problems of equilibrium and vibrations. Bull. Am. Math. Soc. **49**(1), 1–23 (1943)
12. Flyer, N., Fornberg, B., Bayona, V., Barnett, G.A.: On the role of polynomials in RBF-FD approximations: I. Interpolation and accuracy. J. Comput. Phys. **321**, 21–38 (2016)
13. Fornberg, B., Driscoll, T.A., Wright, G., Charles, R.: Observations on the behavior of radial basis function approximations near boundaries. Comput. Math. Appl. **43**(3–5), 473–490 (2002)
14. Geuzaine, C., Remacle, J.-F.: GMSH: A 3-D finite element mesh generator with built-in pre-and post-processing facilities. Int. J. Numer. Meth. Eng. **79**(11), 1309–1331 (2009)
15. Gibou, F., Fedkiw, R.: A fourth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problem. J. Comput. Phys. **202**(2), 577–601 (2005)
16. Gibou, F., Fedkiw, R.P., Cheng, L.T., Kang, M.: A second-order-accurate symmetric discretization of the Poisson equation on irregular domains. J. Comput. Phys. **176**(1), 205–227 (2002)
17. Johansen, H., Colella, P.: A Cartesian grid embedded boundary method for Poisson's equation on irregular domains. J. Comput. Phys. **147**(1), 60–85 (1998)
18. Jomaa, Z., Macaskill, C.: The embedded finite difference method for the Poisson equation in a domain with an irregular boundary and Dirichlet boundary conditions. J. Comput. Phys. **202**(2), 488–506 (2005)
19. Karatzas, E.N., Ballarin, F., Rozza, G.: Projection-based reduced order models for a cut finite element method in parametrized domains. Comput. Math. Appl. **79**(3), 833–851 (2020)
20. Kreiss, H.-O., Petersson, N.A.: A second order accurate embedded boundary method for the wave equation with Dirichlet data. SIAM J. Sci. Comput. **27**, 1141–1167 (2006)
21. Kreiss, H.-O., Petersson, N.A., Ystrom, J.: Difference approximations for the second order wave equation. SIAM J. Numer. Anal. **40**(5), 1940–1967 (2002)
22. Kreiss, H.-O., Petersson, N.A., Ystrom, J.: Difference approximations of the Neumann problem for the second order wave equation. SIAM J. Numer. Anal. **42**(3), 1292–1323 (2004)
23. Li, J.-R., Greengard, L.: High order marching schemes for the wave equation in complex geometry. J. Comput. Phys. **198**(1), 295–309 (2004)
24. Li, Z.: A fast iterative algorithm for elliptic interface problems. SIAM J. Numer. Anal. **35**(1), 230–254 (1998)
25. Liu, S., Tang, Q., Tang, X.: A parallel cut-cell algorithm for the free-boundary grad-shafranov problem. SIAM J. Sci. Comput. **43**(6), B1198–B1225 (2021)
26. Lombard, B., Piraux, J.: Numerical treatment of two-dimensional interfaces for acoustic and elastic waves. J. Comput. Phys. **195**(1), 90–116 (2004)
27. Lombard, B., Piraux, J., Gelis, C., Virieux, J.: Free and smooth boundaries in 2-d finite-difference schemes for transient elastic waves. Geophys. J. Int. **172**(1), 252–261 (2008)
28. Lyon, M.: High-order unconditionally-stable FC-AD PDE solvers for general domains. PhD thesis, Caltech, (2009)
29. Lyon, M., Bruno, O.P.: High-order unconditionally stable FC-AD solvers for general smooth domains II. Elliptic, parabolic and hyperbolic PDEs; theoretical considerations. J. Comput. Phys. **229**(9), 3358–3381 (2010)
30. Min, C., Gibou, F., Ceniceros, H.D.: A supra-convergent finite difference scheme for the variable coefficient poisson equation on non-graded grids. J. Comput. Phys. **218**(1), 123–140 (2006)
31. Ng, Y.T., Chen, H., Min, C., Gibou, F.: Guidelines for Poisson solvers on irregular domains with Dirichlet boundary conditions using the ghost fluid method. J. Sci. Comput. **41**(2), 300–320 (2009)
32. Pember, R.B., Bell, J.B., Colella, P., Curtchfield, W.Y., Welcome, M.L.: An adaptive cartesian grid method for unsteady compressible flow in irregular regions. J. Comput. Phys. **120**(2), 278–304 (1995)
33. Peng, Z., Tang, Q., Tang, X.-Z.: An adaptive discontinuous petrov-galerkin method for the grad-shafranov equation. SIAM J. Sci. Comput. **42**, B1227–B1249 (2020)
34. Peskin, C.S.: Flow patterns around heart valves: a numerical method. J. Comput. Phys. **10**(2), 252–271 (1972)
35. Ruge, J.W. and Stüben, K.: Algebraic multigrid. In Multigrid methods, pages 73–130. SIAM, (1987)
36. Schwartz, P., Barad, M., Colella, P., Ligocki, T.: A Cartesian grid embedded boundary method for the heat equation and Poisson's equation in three dimensions. J. Comput. Phys. **211**(2), 531–550 (2006)
37. Starius, G.: Composite mesh difference methods for elliptic boundary value problems. Numer. Math. **28**(2), 243–258 (1977)
38. Visher, J., Wandzura, S., White, A.: Stable, high-order discretization for evolution of the wave equation in 1+1 dimensions. J. Comput. Phys. **194**(2), 395–408 (2004)
39. Volkov, E.A.: The method of composite meshes for finite and infinite regions with piecewise smooth boundary. Trudy Matematicheskogo Instituta imeni VA Steklova **96**, 117–148 (1968)

40. Wandzura, S.: Stable, high-order discretization for evolution of the wave equation in 2 + 1 dimensions. J. Comput. Phys. **199**(2), 763–775 (2004)
41. Weller, R., Shortely, H.: Calculation of stresses within the boundary of photoelastic models. J. Appl. Mech. **6**, A71–A78 (1939)
42. Wright, G.B.: Radial basis function interpolation: numerical and analytical developments. University of Colorado at Boulder, (2003)