ORIGINAL PAPER

Immersed boundary-conformal isogeometric method for linear elliptic problems

Xiaodong Wei[1] · Benjamin Marussig[3] · Pablo Antolin[1] · Annalisa Buffa[1,2]

Abstract

We present a novel isogeometric method, namely the *Immersed Boundary-Conformal Method* (IBCM), that features a layer of discretization conformal to the boundary while employing a simple background mesh for the remaining domain. In this manner, we leverage the geometric flexibility of the immersed boundary method with the advantages of a conformal discretization, such as intuitive control of mesh resolution around the boundary, higher accuracy per degree of freedom, automatic satisfaction of interface kinematic conditions, and the ability to strongly impose Dirichlet boundary conditions. In the proposed method, starting with a boundary representation of a geometric model, we extrude it to obtain a corresponding conformal layer. Next, a given background B-spline mesh is cut with the conformal layer, leading to two disconnected regions: an exterior region and an interior region. Depending on the problem of interest, one of the two regions is selected to be coupled with the conformal layer through Nitsche's method. Such a construction involves Boolean operations such as difference and union, which therefore require proper stabilization to deal with arbitrarily cut elements. In this regard, we follow our precedent work called the minimal stabilization method (Antolin et al in SIAM J Sci Comput 43(1):A330–A354, 2021). In the end, we solve several 2D benchmark problems to demonstrate improved accuracy and expected convergence with IBCM. Two applications that involve complex geometries are also studied to show the potential of IBCM, including a spanner model and a fiber-reinforced composite model. Moreover, we demonstrate the effectiveness of IBCM in an application that exhibits boundary-layer phenomena.

Keywords Immersed method · Conformal boundary/interface · Boolean operations · Stabilized method · Isogeometric analysis · Boundary layer

1 Introduction

Isogeometric analysis (IGA) was proposed to tightly integrate computer-aided design (CAD) with downstream engineering simulation [2,3], which is enabled by employing the same basis of CAD in simulation. While significant advances have been made in almost every aspect of IGA, dealing with boundary representation (B-rep) remains one of the biggest challenges in the field. In many current commercial CAD

systems, solid modeling is built upon B-reps, where a solid model is represented by its boundary only. However, B-reps are not immediately ready to be used in IGA. A key issue is the lack of a true volumetric description. They need to be processed through either volume parameterization (from the geometry point of view) or the immersed method (from the analysis point of view). Volume parameterization conformal to a given B-rep is appealing but remains an open problem at large [4,5]. Alternatively, the immersed boundary method, which embeds a given B-rep into a simple background mesh (e.g., a Cartesian grid), eliminates the need for conformal and quality volume parameterization and has gained wide popularity in IGA [6–11].

The term "immersed boundary method" indicates a large family of methods and is sometimes used interchangeably with other terms such as "non-boundary-fitted method", "finite cell method", "fictitious domain method" and "embedded domain method". Despite their various origins, all of such

✉ Xiaodong Wei
  xiaodong.wei@epfl.ch

1 Institute of Mathematics, École Polytechnique Fédérale de Lausanne, 1015 Lausanne, Switzerland

2 Istituto di Matematica Applicata e Tecnologie Informatiche "Enrico Magenes" del CNR, Pavia, Italy

3 Institute of Applied Mechanics, Graz University of Technology, 8010 Graz, Austria

methods share the same idea: embedding a B-rep into a simple background mesh. We use the term only to reflect this key idea. Interested readers may refer to [12] and references therein for a comprehensive review.

In the context of IGA, the immersed idea is also closely related to Boolean operations (i.e., intersection, difference, and union) and the related trimming operations, which allow the specification of arbitrary interfaces within the elements of a tensor product surface. Indeed, an immersed method needs to address the same challenging problems encountered in dealing with Boolean operations, such as numerical integration of cut elements [13–17], stabilization for small cut elements [18–21], and imposition of boundary/interface conditions [22,23].

Motivated to accurately capture boundary/interface features while retaining geometric flexibility of immersed methods, we propose a novel method that combines the immersed idea with a conformal discretization. We call it the *immersed boundary-conformal method* (IBCM). The geometric construction of IBCM is conceptually simple. It starts with a B-rep of the domain of interest. The B-rep is first extruded to yield a boundary layer that is clearly conformal to the input B-rep, so we call it the *conformal layer*. Next, we embed the conformal layer into a sufficiently large background mesh such as a Cartesian grid. As a result, the background mesh is cut into two disconnected regions: an exterior region and an interior region. Depending on the problem of interest, one of the two regions is selected and coupled with the conformal layer to constitute (a part of) underlying computation domain. It is noted that we focus on 2D planar domains in this paper.

An IBCM construction involves Boolean operations such as difference and union. Properly dealing with them in analysis is the key to the success of IBCM, which includes numerical integration on cut elements and interfaces, and stabilized formulation regardless of how elements are cut. To address the integration issue, we present an improved decomposition method to minimize the number of resulting quadrature cells. On the other hand, we adopt our previous work, the minimal stabilization method [1,21], to guarantee stability for its simplicity and effectiveness.

Two kinds of applications are studied with IBCM: (1) accuracy enhancement via adding conformal layers to geometric features (e.g., holes, reentrant corners, etc), and (2) stress analysis on domains composed of multiple materials (e.g., inclusions). It is straightforward for IBCM to control the mesh resolution near boundaries/interfaces due to the conformal layers. As singularities and/or large gradients often happen close to the boundary, it is often vital to have a sufficiently fine mesh near the boundary to obtain accurate solutions. The conformal layer also provides the possibility to strongly impose Dirichlet boundary conditions, which, in contrast, is not possible in immersed methods. In

the case of modeling composite materials, material interfaces can be represented with conformal meshes in IBCM, and thus kinematic constraints are automatically satisfied on the interfaces by construction. While strong imposition of boundary/interface conditions is enabled by the conformal layer, coupling it with the background mesh is done weakly with the control over where to put the coupling interface. In addition to all these analysis benefits, we will further show that IBCM indeed can easily model complex geometries via the immersed idea.

It is worth mentioning that there exist several methods that IBCM shares the essential idea with. First, a popular group of methods in computational fluid dynamics, called overset grid methods [24–29], also employ boundary-fitted meshes near geometric features while keeping discretization simple elsewhere. However, the two methods differ in the key building blocks such as geometric representations and treatment of coupling interfaces. Overset grid methods are typically developed on linear elements, and different meshes are coupled via, for instances, hybrid meshes [25], and interpolation schemes [24,27] that solve the problem iteratively to reach a convergent solution. Second, in [30], disk-shaped patches (represented by degenerated NURBS) are added on top of a large domain (represented by a Cartesian grid) to model holes or inclusions. Different domains are coupled in a non-intrusive way that requires tens or even hundreds of iterations to obtain a converged solution for the linear elasticity problem. Moreover, the method primarily aims to capture local features inside a rectangle-shaped domain and thus does not support complex exterior geometries. Third, a multi-mesh finite element method was proposed in [31] to optimize the shapes of disk- or ring-like objects that are embedded in a large background domain. Compared to the multi-mesh approach [31–33], the main differences of our proposed method lie in four aspects: (1) smooth discretization with IGA, (2) emphasis on parameterizations of conformal layers, (3) a minimal stabilization approach, and (4) use of conformal layers to improve accuracy and provide flexible control near the boundary/interface.

The paper is organized as follows. We first introduce several basic concepts and notations in Sect. 2.1. Section 2 introduces how to deal with Boolean operations in IGA, which lays the foundation to our proposed method. Section 3 presents the proposed immersed boundary-conformal method. In Sect. 4, we present several numerical examples to demonstrate the proposed method in terms of improved accuracy, ease of imposing Dirichlet boundary conditions, and flexibility in modeling complex geometries. Section 5 concludes the paper and suggests future work.

2 Analysis-aware treatment of Boolean operations

In this section, we introduce how to handle difference and union operations in IGA. Their analysis-aware treatment lays the foundation to our proposed method. As the difference operation is built upon trimming, we discuss trimming in the following without loss of generality. Our description is based on 2D planar domains unless stated otherwise.

2.1 B-spline/NURBS surfaces

We first introduce several basic notations of B-spline/NURBS surfaces to facilitate our discussion. Given a control mesh $\{P_{ij}\}_{i=1,j=1}^{n,m}$ ($P_{ij} \in \mathbb{R}^2$), degrees p and q, and knot vectors $\{\xi_i\}_{i=1}^{n+p+1}$ and $\{\eta_j\}_{j=1}^{m+q+1}$, a bivariate B-spline basis function is defined as

$$B_{i,p}(\xi) = N_{i,p}(\xi) M_{j,q}(\eta), \tag{1}$$

where $i = (i,j)$, $p = (p,q)$, $\xi = (\xi, \eta)$, and $N_{i,p}(\xi)$ and $M_{j,q}(\eta)$ are univariate B-spline basis functions defined on $\{\xi_i\}_{i=1}^{n+p+1}$ and $\{\eta_j\}_{j=1}^{m+q+1}$, respectively. The geometric mapping of a B-spline surface, $F : \hat{\Omega} \to \Omega$, is defined as

$$F(\xi) = \sum_{i \in \mathcal{I}} P_i B_{i,p}(\xi), \quad \mathcal{I} = \{(i,j)\}_{i=1,j=1}^{n,m}, \tag{2}$$

where $\hat{\Omega} := (\xi_1, \xi_{n+p+1}) \times (\eta_1, \eta_{m+q+1})$ and Ω are called the *parametric domain* and the *physical domain*, respectively. We also call a surface a *patch*, especially when multiple surfaces are involved. $\hat{\Omega}$ is partitioned by the knot vectors, leading to a parametric mesh $\hat{\mathcal{M}}$,

$$\hat{\mathcal{M}} = \{(\xi_i, \xi_{i+1}) \times (\eta_j, \eta_{j+1}) : \quad \xi_i < \xi_{i+1} \text{ and } \eta_j < \eta_{j+1} \tag{3}$$
$$i = 1, \ldots, n+p, \quad j = 1, \ldots, m+q\}.$$

The corresponding physical mesh $\mathcal{M}$ is obtained as the images of elements in $\hat{\mathcal{M}}$ through the geometric mapping,

$$\mathcal{M} = \{F(K) : K \in \hat{\mathcal{M}}\}. \tag{4}$$

The element boundaries in $\hat{\mathcal{M}}$ are represented by knot lines, i.e., $\{\xi_i\} \times [\eta_1, \eta_{m+q+1}]$ and $\{\eta_j\} \times [\xi_1, \xi_{n+p+1}]$. Their images are called knot-line curves and they are the element boundaries in $\mathcal{M}$.

NURBS are a generalization of B-splines and they are able to exactly represent conic sections such as circles and ellipses. A weight $w_i \in \mathbb{R}$ is associated to each control point in addition to its coordinates. The NURBS basis function

corresponding to the B-spline $B_{i,p}(\xi)$ is defined as

$$R_{i,p}(\xi) = \frac{B_{i,p}(\xi) w_i}{\sum_{j \in \mathcal{I}} B_{j,p}(\xi) w_j}. \tag{5}$$

Note that NURBS and B-spline basis functions are equivalent when $w_i = 1 \; \forall i \in \mathcal{I}$, i.e., $R_{i,p} = B_{i,p}(\xi)$. A NURBS surface representation is obtained by replacing $B_{i,p}(\xi)$ with $R_{i,p}(\xi)$ in Eq. 2. To simplify notations, we use $B_{i,p}(\xi)$ to generically denote a B-spline or NURBS basis function in the rest of the paper.

The spline space on the physical domain Ω is defined as

$$\mathcal{B} = \text{span}\{B_{i,p} \circ F^{-1}(x) : i \in \mathcal{I}\}. \tag{6}$$

Let $\Omega_a \subset \Omega$ be a subdomain. We further define the restriction of $\mathcal{B}$ to Ω_a as

$$\mathcal{B}|_{\Omega_a} = \text{span}\{B_{i,p} \circ F^{-1}(x) : i \in \mathcal{I}, \\ \text{supp} B_{i,p} \cap F^{-1}(\Omega_a) \neq \emptyset\}, \tag{7}$$

where $\text{supp} B_{i,p} := (\xi_i, \xi_{i+p+1}) \times (\eta_j, \eta_{j+q+1})$ is the support of $B_{i,p}(\xi)$. $\mathcal{B}|_{\Omega_a}$ will be used in trimmed geometries.

2.2 Trimming

From the geometric point of view, trimming provides a powerful and flexible tool to represent complex shapes that do not have a tensor-product structure. However, it is merely a visualization means to make part of a geometry invisible to users without changing anything else. More specifically, we consider a NURBS surface $F : \hat{\Omega} \to \Omega$. Applying trimming is equivalent to restricting F to a visible or *active* subdomain $\hat{\Omega}_a \subset \hat{\Omega}$. Correspondingly, the complementary portion $F(\hat{\Omega} \setminus \hat{\Omega}_a)$ is invisible or trimmed away. We note that the only change in a trimming operation is the change in the definition domain: from $\hat{\Omega}$ to $\hat{\Omega}_a$. The trimming operation is illustrated in Fig. 1. We can observe its resemblance to the immersed idea, where neither of their background meshes aligns with the domain boundary.

The active subdomain $\hat{\Omega}_a$ is identified by a set of closed loops $\{\hat{\Gamma}_i\}$. Each trimming loop $\hat{\Gamma}_i$ is generally a collection of connected B-spline curves. Trimming loops are oriented such that all points in $\hat{\Omega}_a$ locate on the left side (in convention) of any $\hat{\Gamma}_i$. In practice, trimming curves are often given in the physical domain and may not even lie on the surface $F(\hat{\Omega})$. To obtain the parametric counterpart $\hat{\Gamma}_i$, points are sampled along Γ_i and then pulled back to $\hat{\Omega}$ through the inversion algorithm [34]. $\hat{\Gamma}_i$ is constructed based on these sampled points.

From the analysis point of view, numerical integration is performed on the active subdomain $\hat{\Omega}_a$ rather than the entire domain $\hat{\Omega}$. $\hat{\Omega}_a$ consists of non-trimmed elements and

Fig. 1 Illustration of trimming, where Γ_i $(i = 2, 3)$ are trimming loops in the physical domain
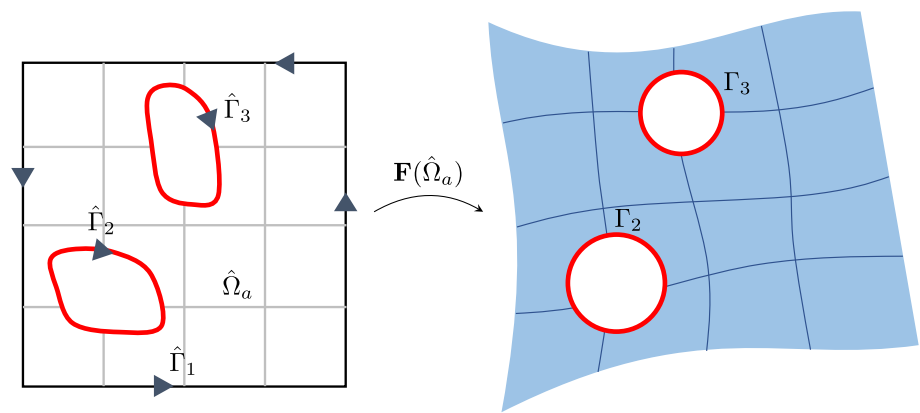
trimmed elements. Given an element $K \in \hat{\mathcal{M}}$, K is cut if $0 < |K \cap \hat{\Omega}_a| < |K|$, where $|\cdot|$ denotes area. Its active portion, $K \cap \hat{\Omega}_a$, is referred to as a trimmed element. Clearly, non-trimmed elements are those with $|K \cap \hat{\Omega}_a| = |K|$, whereas elements with no active area (i.e., $|K \cap \hat{\Omega}_a| = 0$) are inactive in the sense that they are not used in geometric representation or in analysis.

While it is straightforward to integrate non-trimmed elements, the challenge lies in the integration for a trimmed element because it is topologically a polygon with certain edges corresponding to part of a (high-order) trimming curve, where no quadrature rules are immediately available. One solution is to reparameterize trimmed elements such that the standard Gauss quadrature rule can be applied. Reparameterization of a trimmed element is a *local* operation that has no influence on other elements. It consists of an approximation step and a decomposition step. The former approximates each involved trimming curve with a Bézier curve, whereas the latter decomposes trimmed element into quadrilateral/triangular cells. We explain related details in the following. Interested readers may refer to [35] for a comprehensive review on the treatment of trimming in IGA.

Approximation step An element $K \in \hat{\mathcal{M}}$ may be cut by multiple trimming loops. For each involved trimming loop $\hat{\Gamma}_i$, we focus on the portion restricted to K, i.e., $\hat{\Gamma}_i \cap K$. In general, $\hat{\Gamma}_i \cap K$ may be composed of several B-spline curves that are piecewise polynomials, but Gauss quadrature can only be applied to polynomials.

Therefore, we approximate $\hat{\Gamma}_i \cap K$ with a Bézier curve that has a polynomial representation. The degree of the target Bézier curve is chosen to be that of spline discretization even if the trimming curve has a higher degree. It has been proven that this choice guarantees optimal error estimates [16]. The remaining procedure of least-squares fitting the Bézier curve to the sampled points of $\hat{\Gamma}_i \cap K$ is straightforward.

Note that the above discussion is based on the assumption that the trimming curve $\hat{\Gamma}_i \cap K$ is sufficiently smooth. However, generally it involves points of reduced continuity or even C^0 points. In practice, we observe that to retain inte-

gration accuracy, it suffices to only respect those C^0 points, for which we split $\hat{\Gamma}_i \cap K$ at each of them and fit a Bézier curve separately to each resulting piece.

Decomposition step We develop a divide-and-conquer approach to decompose trimmed elements into reparameterization cells. The method applies to 2D only. We first investigate how an element (in the parametric domain) is cut by one or two trimming curves. The key is to find a pair of curves and create one or multiple ruled surfaces between them. In each pair of curves, one has to be the trimming curve, whereas the other can be a straight segment (A, H), a series of concatenated segments (E, F), a point (K), or another trimming curve (B, C, D); see Fig. 2. As a result, 8 base cases are identified. Ruled surfaces are then created between a pair of such curves. Note that cases G and H present a single red curve and lead to degenerated quadrilateral reparameterization cells: For the case G, the ruled surface is created using the red curve and the opposite red vertex; whereas in the case H, the cell is created by splitting the red curve at its two vertices, and generating a ruled surface between them. As
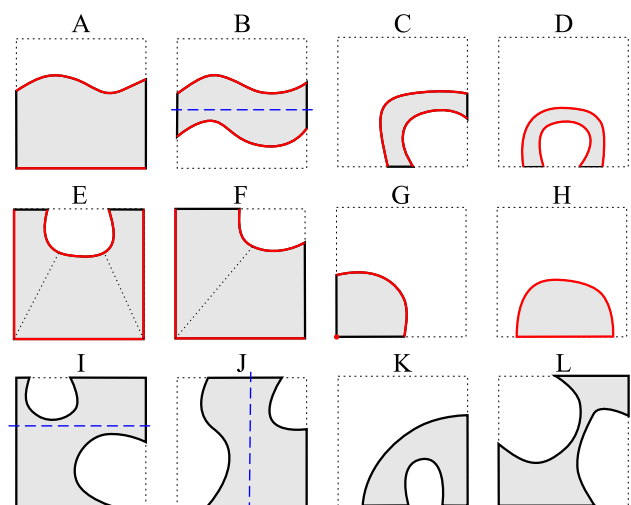

Fig. 2 Reparameterization of trimmed elements using quadrilateral cells

already mentioned above, the curved segments of the trimming loop may present C^0 features. Using this strategy, these features will be explicitly represented in the generated reparameterization. On the other hand, applying the same idea to finding base cases in 3D becomes intractable due to the proliferation of possible cases. One viable solution is to explore the unique base cases of marching cubes [36] and design a corresponding decomposition for each of them. Alternatively, each cut element may be divided into a set of tetrahedra, which, however, leads to a proliferation of integration cells [37,38].

If the algorithm above fails (the generated ruled surface presents a change in its Jacobian's sign), or the trimmed element does not fall into one of the base categories A-H, a divide-and-conquer strategy is applied: The trimming loop is split with a line parallel to one of the two coordinate axes, and the algorithm is applied again for the resulting pieces. This is also the case in which the element presents nested trimming loops. Such a recursive method works up to a certain geometric tolerance, beyond which the splitting algorithm terminates and unresolved trimming features will be ignored. Adoption of geometric tolerances is a common practice in CAD when implementing trimming-related operations.

For selecting the splitting line we consider the bounding boxes of all the curved segments on the trimming loop, cherry pick the one with the smallest volume, and consider a line that passes through that bounding box's mid point and is perpendicular to its longest coordinate direction. Nevertheless, whenever possible, the splitting line is defined such that different curved segments are not intersected by the line and they are separated into different reparameterization cells (see, e.g., cases B, I, and J in Fig. 2). This is not always possible; see for instance cases C, D, K, and L.

2.3 Union

Following [1,33], we adopt a hierarchy of overlapping patches to perform the union operation. We consider a two-patch union for simplicity, which is actually sufficient to help explain our proposed method in Sect. 3. Interested readers may refer to [1] for more general constructions. Given a pair of domains, we put one on top of another. The top and bottom domains have the geometric mappings $\boldsymbol{F}_t : \hat{\Omega}^t \to \Omega^t$ and $\boldsymbol{F}_b : \hat{\Omega}^b \to \Omega^b$, respectively. Their union is created by first trimming Ω^b with Ω^t, leading to a trimmed (bottom) domain $\Omega_a^b = \Omega^b \backslash \Omega^t$. Next, Ω_a^b is combined with Ω^t to constitute the computational domain $\Omega = \Omega_a^b \cup \Omega^t$; see Fig. 3. In such an overlapping construction, the bottom patch is always trimmed whereas the top patch is intact. Note that we can choose either of the two patches to be on the top. It has been shown that different arrangements do not influence solution accuracy or matrix conditioning in linear elliptic problems [1].
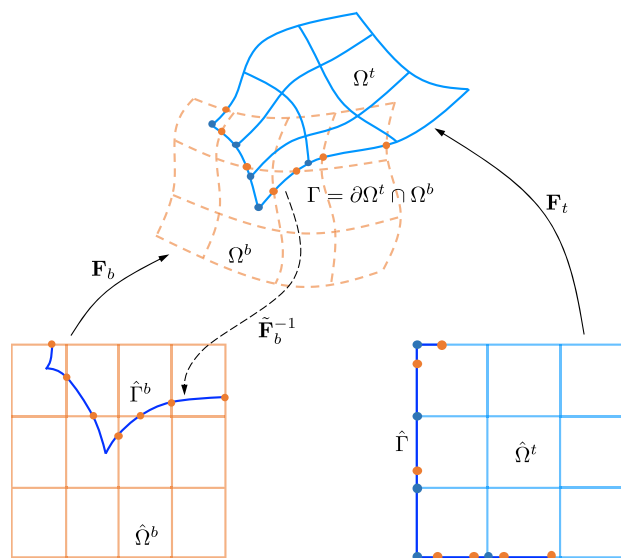

Fig. 3 Union of two overlapping patches and generation of their interface quadrature mesh

With trimming handled according to Sect. 2.2, analysis-aware treatment of the union operation centers on weakly coupling patches through their interfaces, where we choose Nitsche's method for its consistent and symmetric formulation [39]. We then need to address the following challenging issues: (1) generation of interface quadrature meshes to accurately compute involved interface integrals, and (2) stabilization of flux terms in the interface integral to guarantee the well-posedness of the problem regardless of how elements in the bottom patch are cut.

The key to generating an interface quadrature mesh is to find a mesh intersection on the interface. This ensures that on each quadrature cell, all the involved basis functions from both top and bottom patches are polynomials rather than piecewise polynomials. In 2D, this is to find curve–curve intersections between the interface and the knot-line curves of both patches. Note that in the overlapping construction of union, the interface Γ (in the physical domain) is part of the trimming loop in the bottom patch Ω^b. Moreover, Γ itself is part of the top patch boundary, i.e., $\Gamma = \partial\Omega^t \cap \Omega^b \subset \partial\Omega^t$. In other words, Γ already has the mesh information of Ω^t; see, for example, the blue dots in Fig. 3. It is left to find its intersections with the knot-line curves of Ω^b. We proceed in the parametric domain of the bottom patch $\hat{\Omega}^b$. An approximate preimage of Γ is first found in $\hat{\Omega}^b$ through the inversion algorithm and spline fitting, i.e., $\hat{\Gamma}^b = \tilde{\boldsymbol{F}}_b^{-1} \circ \Gamma$, where $\tilde{\boldsymbol{F}}_b^{-1}$ indicates an approximate inversion map. Next, we can easily find the intersections of $\hat{\Gamma}^b$ with the axis-aligned knot lines in $\hat{\Omega}^b$; see the orange dots in $\hat{\Omega}^b$ in Fig. 3. For each resulting intersection point $\boldsymbol{\xi}$, we find its image $\boldsymbol{F}_b(\boldsymbol{\xi})$ in the physical domain and further bring it onto $\partial\hat{\Omega}^t$, leading to a desired intersection $\tilde{\boldsymbol{F}}_t^{-1} \circ \boldsymbol{F}_b(\boldsymbol{\xi})$. Once all the intersections

are found, a 1D mesh can be readily constructed as the interface quadrature mesh, for example, based on the blue and orange dots in $\hat{\Omega}^t$ in Fig. 3.

2.4 Stabilized formulations

We choose Nitsche's method [39] to couple independent domains for its consistent and symmetric formulation. However, the flux terms (those involving normal derivatives) in Nitsche's formulation may give rise to an instability issue when elements adjacent to an interface are badly cut, that is, only an extremely small portion of an element is left after trimming [21,40]. The lack of stability indicates the potential violation of the coercivity condition in the bilinear form, leading to that the well-posedness of the problem is not guaranteed. Therefore, a proper stabilization method is needed and it often poses as one of the most challenging problems in employing Boolean operations in IGA.

In this regard, we adopt the *minimal stabilization* method that was developed in our previous work on the overlapping construction of union [1]. It was originally introduced to address the stability issue in Nitsche's formulation on trimmed boundaries [21]. It has shown optimal convergence behaviors as well as trimming-independent conditioning in various tests. As stabilization methods are problem-dependent, let us take two linear elliptic problems, Poisson's problem and linear elasticity, as the model problems to explain the formulation details.

Poisson's problem In the two-domain setting, the strong form of the Poisson's problem is stated as follows. Given $f : \Omega \to \mathbb{R}$, $g_D : \Gamma_D \to \mathbb{R}$, and $g_N : \Gamma_N \to \mathbb{R}$, find $u : \Omega \to \mathbb{R}$ such that

$$
\begin{cases}
-\Delta u = f & \text{in } \Omega = \Omega^t \cup \Omega_a^b, \\
u^t - u^b = 0 & \text{on } \Gamma = \partial\Omega^t \cap \Omega^b, \\
\nabla u^t \cdot \boldsymbol{n}^t + \nabla u^b \cdot \boldsymbol{n}^b = 0 & \text{on } \Gamma = \partial\Omega^t \cap \Omega^b, \\
u = g_D & \text{on } \Gamma_D, \\
\nabla u \cdot \boldsymbol{n} = g_N & \text{on } \Gamma_N,
\end{cases}
\tag{8}
$$

where $u^t = u|_{\Omega^t}$, $u^b = u|_{\Omega_a^b}$ are restrictions of u to respective domains; $\boldsymbol{n}^t$, $\boldsymbol{n}^b$, and $\boldsymbol{n}$ are outwards unit normals of $\partial\Omega^t$, $\partial\Omega_a^b$ and $\partial\Omega$ respectively; and Γ_D and Γ_N ($\Gamma_D \cap \Gamma_N = \emptyset$, $\overline{\Gamma_D \cup \Gamma_N} = \partial\Omega$) are Dirichlet and Neumann boundaries, respectively. The second and third equations are transmission conditions across the interface Γ. We have also assumed that the Dirichlet boundary Γ_D is not trimmed to simplify formulations. One may refer to [21] for the case of trimmed Γ_D.

Before presenting the weak formulation of Problem (8), we first introduce the following generic approximation space,

$$
\begin{aligned}
\mathcal{V}_h^\alpha = &\{v_h^t \in \mathcal{B}^t : v_h^t|_{\Gamma_D \cap \partial\Omega^t} = \alpha\} \\
&\oplus \{v_h^b \in \mathcal{B}^b|_{\Omega_a^b} : v_h^b|_{\Gamma_D \cap \partial\Omega^b} = \alpha\},
\end{aligned}
\tag{9}
$$

where $\alpha : \Gamma_D \to \mathbb{R}$ is a generic scalar function with suitable regularity, $\mathcal{B}^t$ and $\mathcal{B}^b$ are B-spline (or NURBS) spaces on Ω^t and Ω^b, respectively, and $\mathcal{B}^b|_{\Omega_a^b}$ is the restriction of $\mathcal{B}^b$ to the active subdomain Ω_a^b. We have the approximation spaces of trial functions $\mathcal{V}_h^{g_D}$ and test functions $\mathcal{V}_h^0$ when $\alpha = g_D$ and $\alpha = 0$, respectively.

The discrete weak form of Problem (8) is stated as follows: Find $u_h \in \mathcal{V}_h^{g_D}$ such that

$$
a_h(u_h, v_h) = l(v_h), \quad \forall v_h \in \mathcal{V}_h^0,
\tag{10}
$$

where

$$
\begin{aligned}
a_h(u_h, v_h) = &\int_\Omega \nabla u_h \cdot \nabla v_h \\
&- \int_\Gamma \langle \nabla u_h \cdot \boldsymbol{n}^t \rangle [v_h] - \int_\Gamma \langle \nabla v_h \cdot \boldsymbol{n}^t \rangle [u_h] \\
&+ \beta(h_t^{-1} + h_b^{-1}) \int_\Gamma [u_h][v_h],
\end{aligned}
\tag{11}
$$

and

$$
l(v_h) = \int_\Omega f\, v_h + \int_{\Gamma_N} g_N\, v_h.
\tag{12}
$$

The transmission conditions are weakly enforced by Nitsche's method. In Eq. (11), $[u_h] := u_h^t|_\Gamma - u_h^b|_\Gamma$ is the jump term across the interface Γ, whereas $\langle \nabla u_h \cdot \boldsymbol{n}^t \rangle$ represents the flux through Γ and will be discussed in detail soon. Assuming quasi-uniform meshes in Ω^t and Ω^b, h_t and h_b represent the maximum element sizes of corresponding meshes. β is a penalty parameter. We take $\beta = 6p_{\max}^2$ following [33], where $p_{\max}$ is the maximum degree in the neighboring patches of Γ.

We discuss two types of fluxes here, the symmetric average flux,

$$
\langle \nabla u_h \cdot \boldsymbol{n}^t \rangle = \frac{1}{2}(\nabla u_h^t + \nabla u_h^b) \cdot \boldsymbol{n}^t,
\tag{13}
$$

and the one-sided flux from Ω^t,

$$
\langle \nabla u_h \cdot \boldsymbol{n}^t \rangle = \nabla u_h^t \cdot \boldsymbol{n}^t.
\tag{14}
$$

It has been shown that it is the flux from a certain *trimmed* patch that causes the instability issue [1,21]. Therefore, if the symmetric average flux is adopted in Eq. (11), $\nabla u_h^b \cdot \boldsymbol{n}^t$ is a flux from the trimmed patch Ω_a^b, and thus stabilization

is needed for $\nabla u_h^b \cdot \boldsymbol{n}^t$. In contrast, no further treatment is needed to ensure stability if the one-sided flux from the non-trimmed Ω^t is used.[1] It has been shown that the two types of fluxes yield very similar results in terms of solution accuracy and matrix conditioning for linear elliptic problems [1]. Therefore, we choose the one-sided flux in this work to obtain a formulation without further treatment regarding stabilization.

Remark 1 The stabilization through Eq. (14) leverages the adjacency of trimmed and untrimmed meshes. Essentially, Eq. (14) can be viewed as imposing the third equation in Eq. (8) as a Neumann boundary condition on the trimmed domain. Since trimming at Neumann boundaries does not lead to instability, the entire need for additional stabilization vanishes.

Linear elasticity The second model problem is linear elasticity under assumption of homogeneous and isotropic material, small strains, and small displacements. The strong form is stated as follows. Given $f_i : \Omega \to \mathbb{R}$, $g_{D_i} : \Gamma_{D_i} \to \mathbb{R}$, and $g_{N_i} : \Gamma_{N_i} \to \mathbb{R}$, find the displacement $u_i : \Omega \to \mathbb{R}$ such that

$$\begin{cases} \sigma_{ij,j} + f_i = 0 & \text{in} \quad \Omega = \Omega^t \cup \Omega_a^b, \\ u_i^t - u_i^b = 0 & \text{on} \quad \Gamma = \partial\Omega^t \cap \Omega^b, \\ \sigma_{ij}^t n_j^t + \sigma_{ij}^b n_j^b = 0 & \text{on} \quad \Gamma = \partial\Omega^t \cap \Omega^b, \\ u_i = g_{D_i} & \text{on} \quad \Gamma_{D_i}, \\ \sigma_{ij} n_j = g_{N_i} & \text{on} \quad \Gamma_{N_i}, \end{cases} \quad (15)$$

where

$$\begin{cases} \sigma_{ij} = \lambda \delta_{ij} \delta_{kl} \epsilon_{kl} + \mu(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{jk})\epsilon_{kl}, \\ \epsilon_{kl} = \dfrac{1}{2}(u_{k,l} + u_{l,k}). \end{cases} \quad (16)$$

In Eqs. (15,16), indices i, j, k, and l take on values $1, \ldots, d$, where $d \in \{2, 3\}$ is the number of the spatial dimensions. The displacement field $\boldsymbol{u}$ is vector-valued with u_i being the i-th component; the same notation applies to f_i, g_i, h_i, and n_j (the j-th component of the normal $\boldsymbol{n}$). σ_{ij} and ϵ_{kl} are Cartesian components of the Cauchy stress tensor and the infinitesimal strain tensor, respectively. The comma in $\sigma_{ij,j}$ and $u_{i,j}$ denotes differentiation with respect to the spatial coordinate, e.g., $u_{i,j} = \partial u_i / \partial x_j$. Moreover, the summation convention is applied to repeated indices, for instances, $\sigma_{ij,j} = \sigma_{i1,1} + \sigma_{i2,2}$ and $\sigma_{ij} n_j = \sigma_{i1} n_1 + \sigma_{i2} n_2$ in 2D. The superscripts "t" and "b" again denote restrictions to the top and bottom patches,

respectively. The Dirichlet and Neumann boundary conditions are applied independently in each direction and thus $\Gamma_{D_i} \cap \Gamma_{N_i} = \emptyset$ and $\overline{\Gamma_{D_i} \cup \Gamma_{N_i}} = \partial\Omega$ for $i = 1, \ldots, d$. δ_{ij} is the Kronecker delta, i.e.,

$$\delta_{ij} = \begin{cases} 1 & i = j, \\ 0 & \text{otherwise.} \end{cases} \quad (17)$$

In Eq. (16), constants λ and μ are material parameters called Lamé parameters, which are often expressed in terms of the Young's modulus E and and Poisson's ratio ν,

$$\begin{aligned} \lambda &= \frac{\nu E}{(1 + \nu)(1 - 2\nu)}, \\ \mu &= \frac{E}{2(1 + \nu)}. \end{aligned} \quad (18)$$

In 2D, Eq. (18) falls into the plane strain assumption. Also note that different domains may be occupied by different materials, and thus the values of λ and μ can vary from domain to domain.

The corresponding discrete weak formulation with Nitsche's method to deal with the interface is stated as follows: Find $\boldsymbol{u}_h = (u_{1,h}, u_{2,h}) \in \mathcal{V}_h^{g_{D_1}} \times \mathcal{V}_h^{g_{D_2}}$ (see Eq. (9)) such that

$$a_h(\boldsymbol{u}_h, \boldsymbol{v}_h) = l(\boldsymbol{v}_h), \quad \forall \boldsymbol{v}_h = (v_{1,h}, v_{2,h}) \in \mathcal{V}_h^0 \times \mathcal{V}_h^0, \quad (19)$$

where

$$\begin{aligned} a_h(\boldsymbol{u}_h, \boldsymbol{v}_h) = & \int_\Omega \sigma_{ij}(\boldsymbol{u}_h)\, \epsilon_{ij}(\boldsymbol{v}_h) \\ & - \int_\Gamma \langle \sigma_{ij}(\boldsymbol{u}_h) n_j^t \rangle [v_{i,h}] - \int_\Gamma \langle \sigma_{ij}(\boldsymbol{v}_h) n_j^t \rangle [u_{i,h}] \\ & + \beta \left(h_t^{-1} + h_b^{-1} \right) \int_\Gamma [u_{i,h}][v_{i,h}], \end{aligned} \quad (20)$$

and

$$l(\boldsymbol{v}_h) = \int_\Omega f_i\, v_{i,h} + \sum_{j=1}^d \left(\int_{\Gamma_{N_j}} g_{N_i}\, v_{i,h} \right). \quad (21)$$

In Eq. (20), $[u_{i,h}] := u_{i,h}^t - u_{i,h}^b$ and $\langle \sigma_{ij}(\boldsymbol{u}_h) n_j^t \rangle$ denote the displacement jump and the stress flux across Γ, respectively. We again adopt the one-sided flux, i.e.,

$$\langle \sigma_{ij}(\boldsymbol{u}_h)\, n_j^t \rangle = \sigma_{ij}^t(\boldsymbol{u}_h^t)\, n_j^t, \quad (22)$$

which does not need further treatment for stabilization. The penalty parameter β depends on both spline degrees and
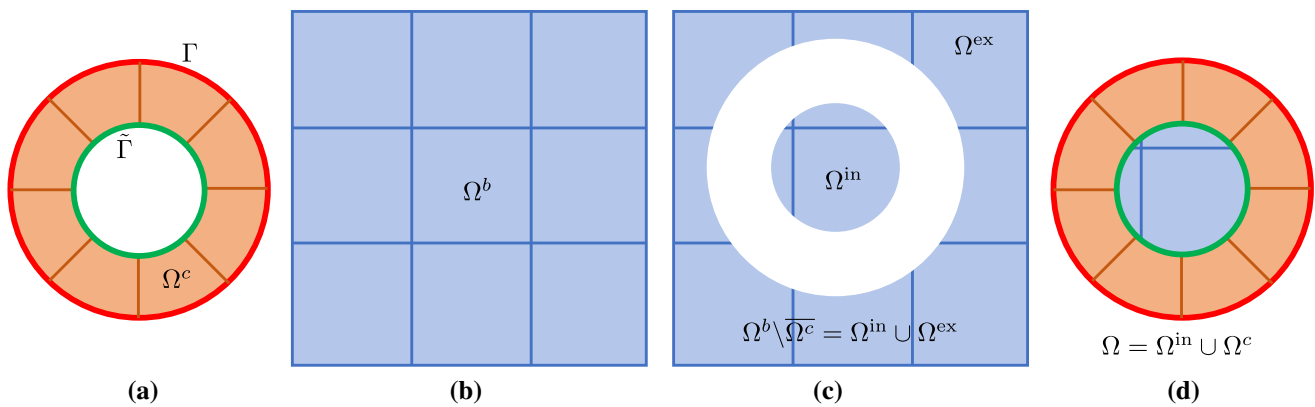
---

[1] This is true only in the two-patch union where Ω^t is already at the top of the overlapping hierarchy. In the multi-patch union, Ω^t may be cut by certain patches that are on top of Ω^t, and thus $\nabla u_h^t \cdot \boldsymbol{n}^t$ may also need to be stabilized according to, for instance, the minimal stabilization [1].

Fig. 4 Geometric construction of IBCM for the boundary type. **a** The input boundary curve Γ (the red circle) and the resulting conformal layer Ω^c by extruding Γ inwards to the target curve $\tilde{\Gamma}$, **b** the back-ground patch Ω^b, **c** two disconnected regions (Ω^{ex} and Ω^{in}) by cutting Ω^b with Ω^c, and **d** the computational domain Ω composed of Ω^{in} and Ω^c

material properties,

$$\beta = 6 p_{\max}^2 \times 8(3\lambda_{\max} + 2\mu_{\max}), \tag{23}$$

where $\lambda_{\max}$ and $\mu_{\max}$ are the maximum Lamé constants of neighboring patches. The choice is inspired by [33,41].

3 Immersed boundary-conformal isogeometric method

In this section, we introduce the proposed method, namely the *Immersed Boundary-Conformal Method* (IBCM). We start with two types of geometric constructions using IBCM. We then proceed to discuss the key technologies that guarantee IBCM to work properly, including parameterization-consistent extrusion and analysis-aware treatment of Boolean operations in IBCM.

3.1 Geometric construction

The geometric construction of IBCM is conceptually simple and can be divided into two types depending on the features of interest: the boundary type and the interface type. The *boundary type* aims to capture boundary features, such as boundary shapes and local solution features near boundaries. The *interface type*, on the other hand, deals with interfaces between different materials such as inclusions and fiber-reinforced materials, where stresses exhibit discontinuity across material interfaces. The major steps in constructing an IBCM representation are the same in both types, including extrusion, trimming and union.

Boundary-type construction We first explain the boundary type. In the extrusion step, we start with the representation of a boundary Γ, which is a closed loop formed by a set of oriented and connected B-spline/NURBS curves. We extrude the loop inwards to yield a ring-like layer Ω^c that is obviously conformal to the given boundary, so we call it a *conformal layer*. Ω^c is generally represented by multiple B-spline/NURBS patches. Details of constructing Ω^c will be discussed in Sect. 3.2.

We explain the remaining steps with the reference to Fig. 4. In the trimming step, we embed Ω^c into a sufficiently large background domain Ω^b such that $\Omega^c \subset \Omega^b$. Ω^b is usually represented by a B-spline mesh defined on a Cartesian grid. As a result, Ω^b is cut by Ω^c into two disconnected regions: an exterior region Ω^{ex} and an interior region Ω^{in}. In other words, we have $\Omega^b \backslash \overline{\Omega^c} = \Omega^{\text{ex}} \cup \Omega^{\text{in}}$.

Finally in the union step, Ω^{in} is coupled with Ω^c to constitute the computational domain Ω, i.e., $\Omega = \Omega^{\text{in}} \cup \Omega^c$. This way, we obtain the IBCM representation of Ω.

Interface-type construction We next discuss the interface-type construction, which can be obtained by repeatedly applying the boundary-type construction. Let Γ denote an interface of two domains Ω^1 and Ω^2, each of which has its own material properties. We explain details in the following with the help of Fig. 5.

We first create an IBCM representation for Ω^1, which in fact follows the same procedure as the boundary-type construction; see Fig. 5b. The IBCM construction for Ω^2 is almost the same. Now, Γ is extruded towards the interior of Ω^2 to obtain a conformal layer $\Omega^{2,c}$. Clearly, the extrusion direction is opposite to that for Ω^1. After the corresponding background mesh is cut, the exterior region is coupled with $\Omega^{2,c}$; see Fig. 5c. Finally, the two IBCM representations are combined together on Γ in a conformal manner, leading to an entire IBCM representation of $\Omega^1 \cup \Omega^2$, as shown in Fig. 5d.

In general, when more than one boundary/interface feature appears in geometric modeling, we need to construct
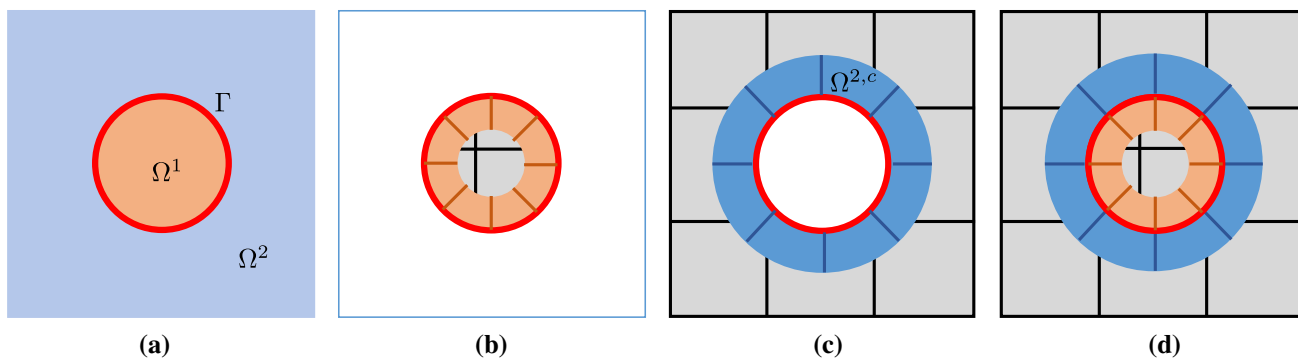
Fig. 5 Geometric construction of IBCM for the interface type. **a** Two domains with different material properties and the interface Γ, **b**, **c** the IBCM representation of Ω^1 and Ω^2, respectively, and **d** the entire IBCM representation of $\Omega^1 \cup \Omega^2$

an IBCM representation for each of them. Combining all the resulting IBCM representations together is straightforward through conformal interfaces. We will present such an example in Sect. 4.

Remark 2 IBCM aims to leverage the geometric flexibility of immersed methods with the advantages of boundary-fitted methods. While the large portion of a geometry is represented following the immersed manner, its boundary (or interface), as the key geometric feature, has a boundary-fitted representation through the conformal layer. In other words, the conformal mesh is placed where it is needed most. On the other hand, meshing is still needed in IBCM to obtain a desired conformal layer through extrusion. Although it remains a challenge in general cases, extrusion is much easier to manage than finding a boundary-conformal parameterization for the entire domain. From this perspective, IBCM helps alleviate the meshing difficulties encountered in IGA while retaining geometry-aligned discretization around key geometric features.

Remark 3 Several benefits of conformal discretization are immediately available in IBCM thanks to the conformal layer. First, it is possible to strongly impose Dirichlet boundary conditions, which is preferable when point-wise satisfaction is desired (e.g., the clamped boundary in solid mechanics). In contrast, imposing Dirichlet boundary conditions often poses as one of the biggest challenges in immersed methods. Second, IBCM is also suitable to model material interfaces when perfect bonding is the case, as the kinematic constraints are naturally met due to the conformal representation of the interface.

Remark 4 In immersed boundary methods, the geometry boundary always serves as the trimming loop on the background mesh. In contrast, IBCM moves trimming curves away from the boundary/interface. As such geometric features are often critical to solution accuracy and trimming is the origin of various issues in analysis, we expect that
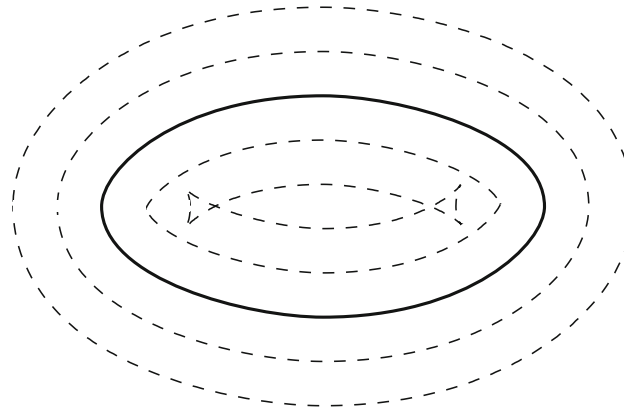

Fig. 6 A progenitor curve (thick) and various offset curves (dashed) with different distance d

by separating the two, IBCM can benefit solution accuracy. Moreover, as conformal layers align with geometric features, it is much more intuitive and convenient for IBCM to control mesh resolutions there than by using immersed methods.

3.2 Parameterization-consistent extrusion

We aim for a conformal layer Ω^c that (1) represents the boundary or interface Γ exactly and (2) is consistent with the parameterization of Γ. First, we define a target curve $\tilde{\Gamma}$, which specifies the interface of Ω^c to the interior region of the background domain Ω^{in}; see Fig. 4a.

From a geometric point of view, the most natural choice for $\tilde{\Gamma}$ is an offset to Γ. Offset curves are defined as the locus of points that are at a constant distance d along the normal vector from the so-called progenitor curve [42,43], and there are several techniques in the literature to generate them; see e.g. [44,45]. These schemes are well-suited to preserve geometric features such as kinks and cusps. In general, the resulting offset curves are more complex than the progenitor curve, and they may have additional cusps and may self-intersect locally or globally.

These self-intersections are linked to the distance d of the offset curve (see Fig. 6). In particular, local self-intersections occur in concave regions when the absolute value of d exceeds the minimum radius of curvature, and global self-intersections occur when the distance between two distinct points on the progenitor curve is smaller than $2d$ [43]. Wallner et al. [46] presented an approach to determine the maximal offset distance to avoid self-intersections.

To sum up, using offset curves for the construction of the conformal layer Ω^c may imply that the parameterization from the target curve $\tilde{\Gamma}$ does not match with the one from the interface Γ, especially when d is large. In this case, Ω^c may be constructed as a loft surface between Γ and $\tilde{\Gamma}$, where a superset of the knot vectors of Γ and $\tilde{\Gamma}$ define the resulting parameterization. A loft surface is created by fitting a series of given curves, with the control over the tangents of the surface. The related function is available in many CAD systems such as Rhinoceros [47].

This approach has some disadvantages: First, the final parameterization of Ω^c is partly determined by the offset curve scheme, and second, C^0-continuities may be introduced when the offset curve has additional cusps or self-intersections that have been trimmed away.

Fortunately, we are very flexible in the definition of the target curve $\tilde{\Gamma}$ in terms of the distance d and its shape, as will be shown in Sect. 4. Hence, we suggest the following procedure for the construction of Ω^c:

1. Define $\tilde{\Gamma}$ either by an offset curve or any other curve that does not intersect Γ and is at least as smooth as Γ.
2. Project the Greville points of Γ onto $\tilde{\Gamma}$.
3. Use the projected Greville points to construct an approximation of $\tilde{\Gamma}$ that has the same knots and degree as Γ.

Greville points are defined as follows. We consider a degree-p NURBS curve whose knot vector is $\{\xi_1, \xi_2, \ldots, \xi_{n+p+1}\}$, where n is the number of control points. The Greville abscissae are defined as an average of certain knots: $g_i = (\xi_{i+1} + \cdots + \xi_{i+p})/p$, $i = 1, \ldots, n$. Letting $C(\xi)$ be the geometric mapping of the NURBS curve, we call $C(g_i)$ its Greville points.

This Greville point projection allows a straightforward realization of the conformal layer Ω^c as a parameterization-consistent extrusion of Γ. Note that the knot vector of the boundary or interface Γ determines the continuity of the final approximation of $\tilde{\Gamma}$, which may be less smooth than the target curve $\tilde{\Gamma}$. In general, this does not lead to any complications in the construction. However, for sharp features of Γ such as cusps, it is beneficial to incorporate the present C^0 continuity already in the target curve. To do so, an offset with a relatively small distance d can be employed to capture the corresponding portion of $\tilde{\Gamma}$.

Remark 5 When the target curve $\tilde{\Gamma}$ is given by an offset curve, we first check if it has additional cusps or regions with high curvature. If so, we replace them with rounded fillets to avoid that projected Greville points coincide.

Remark 6 The proposed construction of the conformal layer Ω^c does not guarantee that the resulting parameterization is bijective. Thus, we check if the Jacobian determinant is greater than zero at the integration points. If this is not the case, Ω^c may be reconstructed either by simply decreasing the offset distance d or using more advanced techniques such as elliptic grid generation [48], the introduction of internal guiding curves [49] or integer-grid maps [50]. It is pointed out that all examples in this paper do not need any correction of the conformal layer because the parameterizations obtained are bijective; see Sect. 4.

3.3 Boolean operations in IBCM

Recall that the geometric construction of IBCM involves trimming and union. Trimming applies to the background B-spline mesh, where we follow Sect. 2.2 to reparameterize cut elements for numerical integration. Note that the geometric mapping of the background patch is usually an identity map, which eliminates the need to find an approximate trimming loop in the parametric domain through the inversion algorithm.

In the union operation, part of the trimmed background mesh is coupled with the conformal layer using Nitsche's method. From the overlapping perspective, the background patch lies on the bottom whereas the conformal layer is on top. We follow Sect. 2.3 to generate an interface quadrature mesh. Moreover, as discussed in Sect. 2.4, with the one-sided flux from the (non-trimmed) conformal layer, Nitsche's formulation is stable and needs no further treatment. To this end, Boolean operations are properly handled in IBCM such that it can be readily applied to linear elliptic problems.

Remark 7 As weak coupling depends on the problem of interest, extending the method to other problems, such as shells, nonlinear elasticity, and Stokes or Navier-Stokes problems, requires further investigation particularly in the analysis side. A different coupling method may also be needed when it is too cumbersome to use Nitsche's formulation. There are indeed many challenging problems for IBCM to accommodate a larger class of problems. Among them, our priority lies in extending the method to 3D, which needs robust implementation to support trimming, union and extrusion.

4 Numerical examples

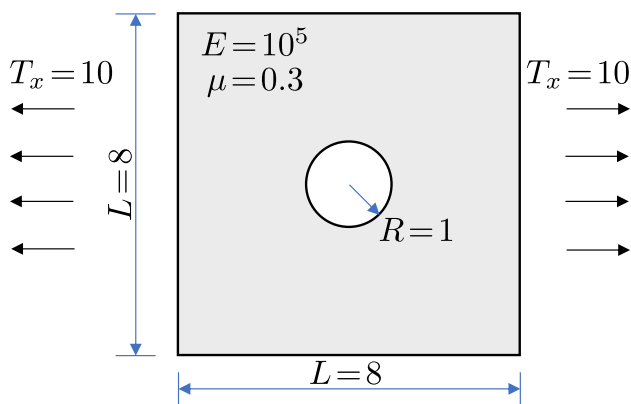In this section, we first study several 2D benchmark problems to evaluate the accuracy and convergence of the

Fig. 7 Problem settings of the plate-with-a-hole example. T_x is the applied traction at infinity in the x direction
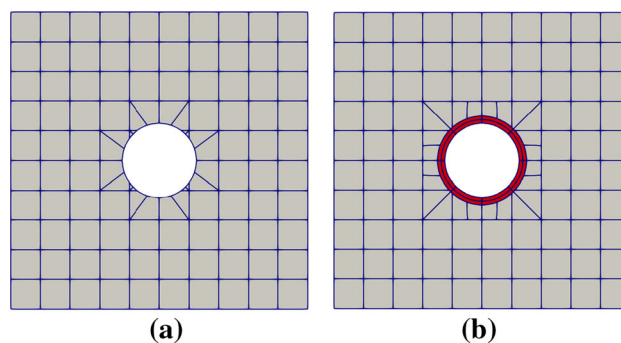

Fig. 8 Initial Bézier meshes of the plate-with-a-hole problem via trimming (**a**) and IBCM (**b**). In **b**, an extra annulus is added around the hole as the conformal layer. In the background mesh, integration cells of trimmed elements are used for visualization
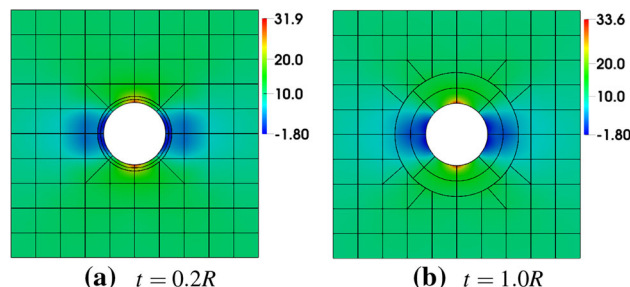

Fig. 9 Different thicknesses of the conformal layer. The color map represents numerical solution of σ_{xx} on the initial mesh. The exact maximum σ_{xx} is 30. In the background mesh, integration cells of trimmed elements are used for visualization

proposed method. They include problems concerning boundary features as well as modeling interfaces between different materials. The former includes examples of a plate with a hole, an L-shaped domain, and a flower geometry, whereas the latter studies a bimaterial disk. Moreover, we present two examples to show the capability of IBCM in representing complex geometric features, which includes a spanner model and a fiber-reinforced composite. In the end, we push a step forward to apply IBCM to an advection–diffusion problem, where we will observe that IBCM can help efficiently resolve the boundary-layer phenomenon in an intuitive manner. In all the tests, biquadratic spline bases are used. In addition, Dirichlet boundary conditions are always strongly imposed except for the advection–diffusion problem, but as splines are not interpolatory, non-homogeneous Dirichlet data needs to be projected to the involved spline spaces.

4.1 Plate with a hole

We start with the plate-with-a-hole test. It is a linear elasticity problem where an infinite plate with a circular hole is under constant in-plane tension. A finite portion with the hole being at the center is taken for the numerical test; see Fig. 7 for the problem settings.

Clearly, the hole is a boundary geometric feature of interest. We study two kinds of geometric constructions for comparison, trimming versus IBCM; see Fig. 8. In the trimming construction, the hole is represented by a NURBS curve, which is also the trimming loop of the background mesh. In this case, we only need to reparameterize cut elements for numerical integration. Alternatively in an IBCM construction, we observe that a conformal layer (marked in red), represented by a NURBS patch, is an annulus sitting on top of the background mesh. The geometric feature is represented by a conformal discretization in the computation domain. The conformal layer is then coupled with the cut background mesh through the union operation. Moreover,

the trimming loop of the background mesh becomes the outer circle of the annulus, which, as opposed to the trimming construction, is no longer the geometric feature itself.

We also study the influence of the thickness t of the conformal layer. We consider $t = 0.2R$, $t = 0.6R$ and $t = 1.0R$, with R being the radius of the hole. In each case, the conformal layer (i.e., the annulus) is represented by a 8×2 mesh; see Fig. 9.

As a reference, we further take a conformal discretization for the whole geometry, which in this case is easy to obtain via a single biquadratic NURBS patch; see the input Bézier mesh in Fig. 10. The element size around the hole is comparable to that of IBCM with $t = 0.6R$.

We summarize the convergence plots in Fig. 11, with respect to both the element size indicator h and the square root of degrees of freedom $DOF^{1/2}$. We observe that compared to the trimming construction, all IBCM constructions yields much more accurate results. As expected, the case of $t = 0.2R$ has the smallest error because its conformal layer has the smallest element size around the hole. The other two cases ($t = 0.6R$ and $t = 1.0R$), on the other hand, also achieve comparable accuracy and still improve a lot compared to the trimming construction. This implies that the solution may not be sensitive to the thickness of the
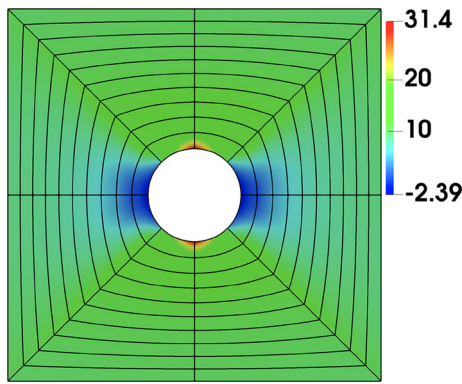
Fig. 10 Conformal discretization for the whole plate-with-a-hole geometry. The color map represents numerical solution of σ_{xx} on the initial mesh. The exact maximum σ_{xx} is 30. (Color figure online)

conformal layer, which, however, needs further study to conclude. We also notice in Fig. 11b that the result of the full conformal discretization is very close to that of IBCM with $t = 0.6R$, which is expected because they have a comparable element size around the hole. In other words, IBCM can achieve the same level of accuracy as a full conformal discretization when critical geometric features are resolved similarly.

4.2 L-shaped domain

This example is aimed to show how IBCM improves solution accuracy when the solution has a local feature, such as a large gradient or even singularity. Such local phenomena are often closely related to geometric features. For example, the solution gradient may exhibit singularity at certain corners. The key idea is to add a layer of mesh to where local features of the solution are expected. This indeed needs "rough" a priori knowledge about the solution field.

As an example, we solve the Laplace equation on a L-shaped domain, where the solution at the reentrant corner exhibits a singularity in its first order derivative. As a result, the convergence rates are governed by the solution regularity rather than the degree of the basis, which are expected to be $\frac{4}{3}$ and $\frac{2}{3}$ for L^2- and H^1-norm errors, respectively. As a reference test shown in Fig. 12a, the domain is represented by a single B-spline patch that is globally C^0-continuous due to sharp corners. Alternatively, we put an extra layer of mesh (three quarters of a disk) on top of the B-spline patch; see Fig. 12b. The extra layer is represented by a degenerated NURBS patch. Note that such a construction is merely a union of overlapping patches rather than an IBCM construction, but the idea coincides with IBCM in the spirit of better capturing local geometry/solution features by adding extra layers. Therefore, we treat it as a special case of IBCM. We compare and summarize the convergence plots of the two geometric
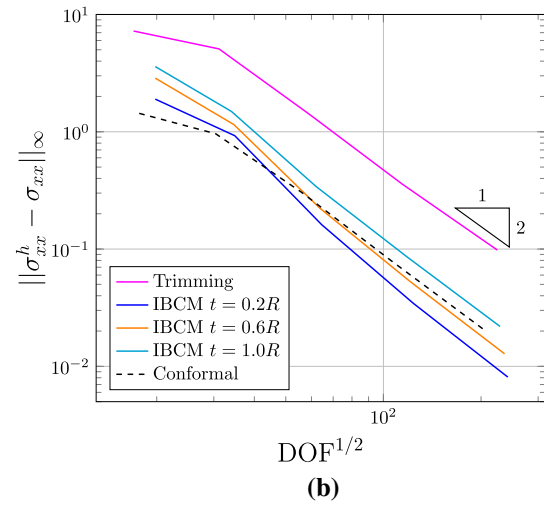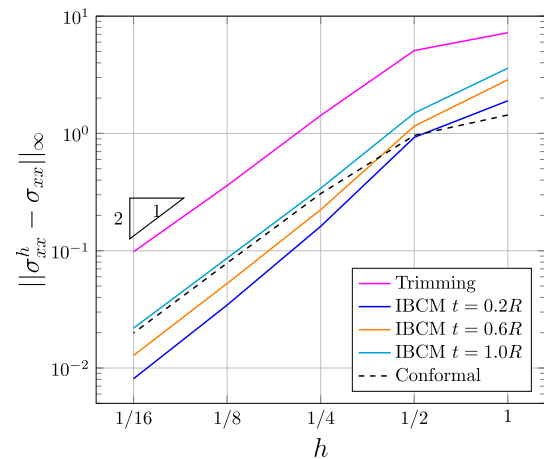

Fig. 11 Convergence plots with respect to h (**a**) and $DOF^{1/2}$ (**b**) in the plate-with-a-hole problem
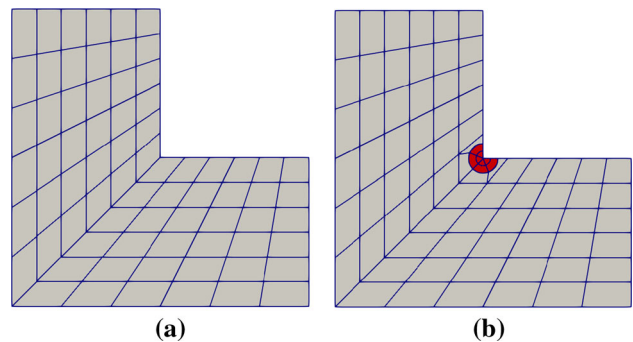

Fig. 12 Initial Bézier meshes of the L-shaped domain by a single B-spline patch (**a**) and via IBCM (**b**). In **b**, a conformal layer (shaded red) is added on top of the interior material to yield a conformal discretization around the material interface

constructions in Fig. 13. We observe that both constructions eventually yield the same convergence rates but IBCM, as expected, greatly improves accuracy with the same DOF (or mesh size).
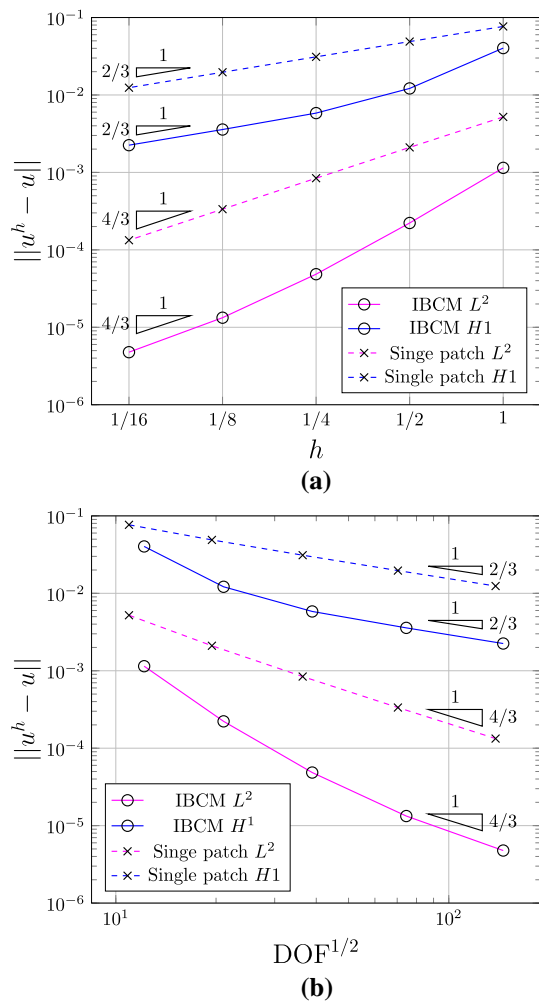
(a)

(b)

Fig. 13 Convergence plots with respect to h (**a**) and DOF$^{1/2}$ (**b**) in the L-shaped domain problem. In the background mesh of (**b**), integration cells of trimmed elements are used for visualization. (Color figure online)

Fig. 14 Conformal layers obtained through an offset curve (**a**) and a circular target curve (**b**)

(a) **(b)**
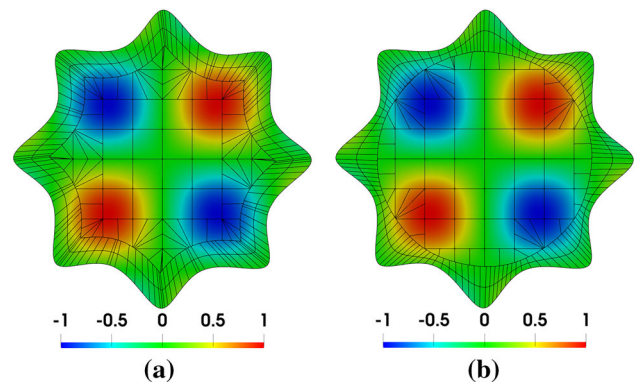
Fig. 15 Solutions on the initial Bézier mesh that has an offset-based conformal layer (**a**), and on the mesh that has a circle-based conformal layer (**b**). In the background mesh, integration cells of trimmed elements are used for visualization

On the other hand, it is worth mentioning that adaptive mesh refinement is usually the method of choice to recover optimal convergence for problems with irregular solutions. In IGA, T-splines [51–54], hierarchical B-splines [55–59], and locally-refinable B-splines [60,61] are typical examples in this family of methods.

4.3 Flower

Next, we solve Poisson's equation on a flower-shaped domain to test how the shape of a conformal layer influences the numerical solution. The input is a B-spline curve Γ representing the flower boundary. We study two different constructions for the target curve $\tilde{\Gamma}$: (1) $\tilde{\Gamma}$ is constructed as an offset curve; and (2) $\tilde{\Gamma}$ is simply a circle; see Fig. 14. In Case 1, the offset curve has a similar shape to Γ, but it generally has a different knot vector from Γ. Therefore, a loft surface is constructed

as the conformal layer Ω^c. As a result, its knot vector is a superset of those of both Γ and $\tilde{\Gamma}$, leading to a dense mesh in Ω^c. Moreover, C^0 continuities may be introduced to Ω^c due to the presence of cusps and removal of self-intersections in $\tilde{\Gamma}$.

On the other hand, due to the flexibility of Case 2, $\tilde{\Gamma}$ can maintain the same knot vector as Γ through Greville point projection. Thus, Ω^c is constructed simply as ruled surface between Γ and $\tilde{\Gamma}$. This way, the mesh resolution of Ω^c is controlled by the input boundary curve.

With the conformal layers, we construct their corresponding IBCM representations and perform a convergence study using the following manufactured solution,

$$u(x, y) = \sin\left(\frac{\pi x}{R}\right) \sin\left(\frac{\pi y}{R}\right), \tag{24}$$

where R is a constant related to the "radius" of the flower. The Dirichlet boundary condition is strongly imposed on the entire boundary. The solutions on the initial Bézier meshes are shown in Fig. 15, where we have refined the conformal layers to make them have a similar mesh resolution. Moreover, we summarize the convergence plots in Fig. 16. We
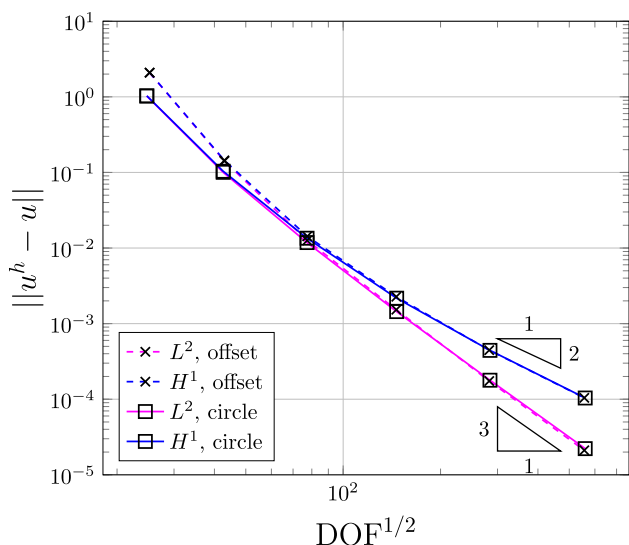
Fig. 16 Convergence plots with respect to $\text{DOF}^{1/2}$ in the flower example. Note that "offset" indicates the results using the offset target curve, whereas "circle" corresponds to results using the circular target curve
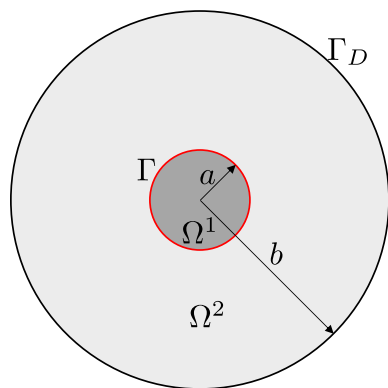

Fig. 17 A disk composed of two domains with different materials Ω^1 and Ω^2, where Γ (the red circle) is the material interface. (Color figure online)

observe that the results are almost identical in both geometric representations. In other words, numerical results are almost not influenced by the shape of the target curve, although further study is needed to come to a conclusion. Nonetheless, it indicates that we are not restricted to a specific choice for the target curve, which in turn provides further flexibility to the IBCM construction.

4.4 Bimaterial disk

This test is motivated by [62] and is aimed to test how IBCM resolves interface features when modeling multiple materials. We solve the linear elasticity problem on a disk composed of two different materials, the interior material Ω^1 and the exterior material Ω^2; see Fig. 17. Their corresponding material properties are given as $E_1 = 1$, $\nu_1 = 0.25$,

and $E_2 = 10$, $\nu_2 = 0.3$, respectively. The Lamé constants (λ_1, μ_1 and λ_2, μ_2) are obtained according to Eq. (18). The exact solutions are given in polar coordinates (r, θ). The displacement field is written as

$$u_r(r) = \begin{cases} \left[\left(1 - \frac{b^2}{a^2}\right)\alpha + \frac{b^2}{a^2}\right] r, \ 0 \leq r \leq a, \\ \left(r - \frac{b^2}{r}\right)\alpha + \frac{b^2}{r}, \quad a < r \leq b, \end{cases} \quad (25)$$

$$u_\theta = 0,$$

where a and b are the radius of Ω^1 and the outer radius of Ω^2, respectively, and

$$\alpha = \frac{(\lambda_1 + \mu_1 + \mu_2)b^2}{(\lambda_2 + \mu_2)a^2 + (\lambda_1 + \mu_1)(b^2 - a^2) + \mu_2 b^2}. \quad (26)$$

We set $a = 0.4$ and $b = 2.0$ in the test. The radial (ϵ_{rr}) and hoop ($\epsilon_{\theta\theta}$) strains are given by

$$\epsilon_{rr}(r) = \begin{cases} \left(1 - \frac{b^2}{a^2}\right)\alpha + \frac{b^2}{a^2}, \ 0 \leq r \leq a, \\ \left(1 + \frac{b^2}{r^2}\right)\alpha - \frac{b^2}{r^2}, \ a < r \leq b, \end{cases} \quad (27)$$

and

$$\epsilon_{\theta\theta}(r) = \begin{cases} \left(1 - \frac{b^2}{a^2}\right)\alpha + \frac{b^2}{a^2}, \ 0 \leq r \leq a, \\ \left(1 - \frac{b^2}{r^2}\right)\alpha + \frac{b^2}{r^2}, \ a < r \leq b. \end{cases} \quad (28)$$

The radial (σ_{rr}) and hoop ($\sigma_{\theta\theta}$) stresses are

$$\begin{aligned} \sigma_{rr}(r) &= \lambda(\epsilon_{rr} + \epsilon_{\theta\theta}) + 2\mu\epsilon_{rr}, \\ \sigma_{\theta\theta}(r) &= \lambda(\epsilon_{rr} + \epsilon_{\theta\theta}) + 2\mu\epsilon_{\theta\theta}, \end{aligned} \quad (29)$$

where $(\lambda, \mu) \in \{(\lambda_1, \mu_1), (\lambda_2, \mu_2)\}$. The shear stress is zero everywhere. All solutions are axisymmetric in the sense that they are independent of θ. We observe that in Eqs. (25, 27, 28), u_r and $\epsilon_{\theta\theta}$ are continuous across the material interface, but ϵ_{rr} experiences a discontinuity. Moreover, both stresses (σ_{rr} and $\sigma_{\theta\theta}$) are discontinuous due to different material parameters.

Two geometric constructions are studied: a union of two overlapping patches (as in [1]) and an IBCM construction; see Fig. 18. In the union construction, the grey annulus sits on top of the yellow background patch, which represent the material domains Ω^2 and Ω^1, respectively. On the other hand, the interior material Ω^1 in an IBCM representation is represented by two patches, the green conformal layer and the yellow background mesh. The mesh of Ω^2 remains the same. According to Eq. (25), the exact displacement, $u_r(b) = b$, is applied on the outer boundary Γ_D; see Fig. 17. We summarize the convergence plots in Fig. 19. Again, we observe the same expected convergence rates but improved accuracy per DOF in IBCM.
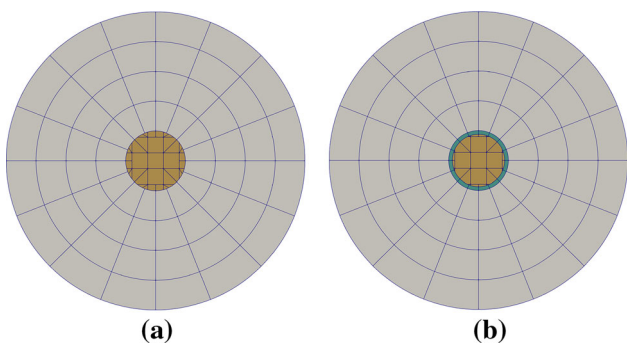
Fig. 18 Initial Bézier meshes of the bimaterial disk via union (**a**) and IBCM (**b**), where each patch is represented by a different color. Particularly in **b**, the interior material Ω^1 is represented by two patches: the green annulus and the yellow background mesh. In the background mesh, integration cells of trimmed elements are used for visualization. (Color figure online)
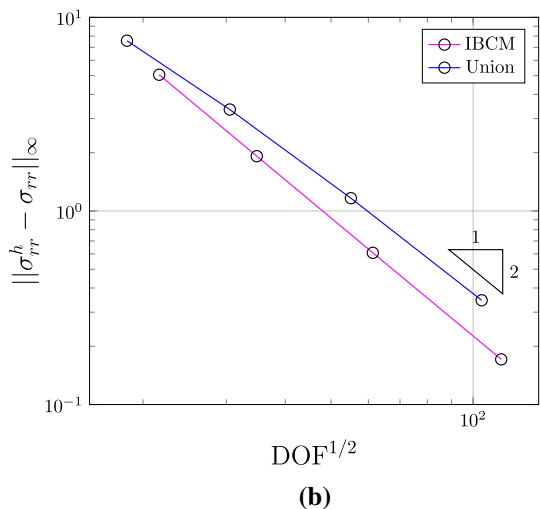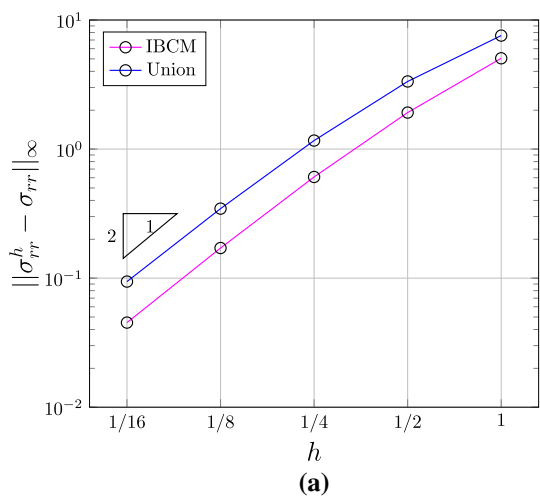

Fig. 19 Convergence plots with respect to h (**a**) and $\text{DOF}^{1/2}$ (**b**) in the bimaterial disk test
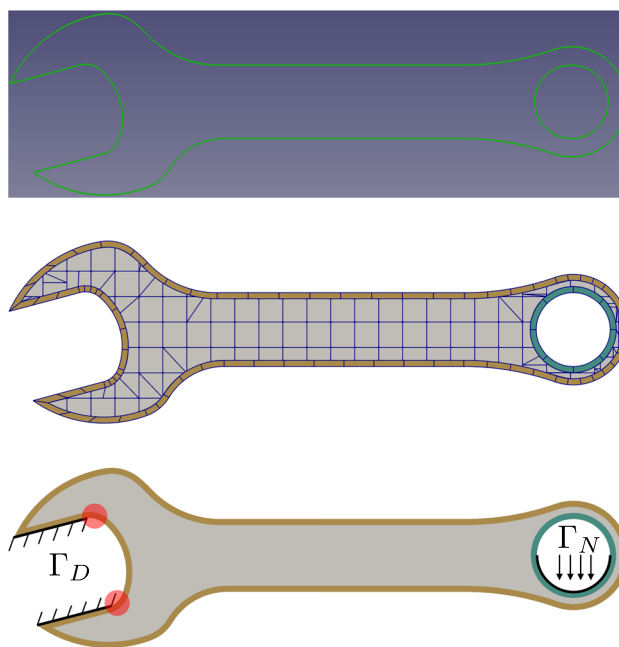
Fig. 20 The input B-rep of spanner (**a**), the corresponding IBCM representation (**b**), and boundary conditions (**c**). In the background mesh of **b**, integration cells of trimmed elements are used for visualization. In **c**, stress singularities are expected in the red-spotted regions due to the sudden change from the clamped displacement condition to the traction-free condition

4.5 Spanner

We next present a spanner model to demonstrate how to use IBCM to represent complex geometries. We will also show that it is flexible for IBCM to deal with Dirichlet boundary conditions as well as to control the mesh resolution around areas of interest. As shown in Fig. 20a, the input B-rep of the spanner is composed of two loops, each represented by a set of NURBS curves. A detailed description of the geometry data is given in [63]. We first generate an offset curve for each loop, which, in this particular case, has a nearly identical parameterization as the input loop. The conformal layer is then immediately available as a loft surface; see Fig. 20b. The remaining procedure of IBCM follows the boundary-type construction.

We proceed to solve the linear elasticity problem on the IBCM representation, with material constants $E = 10^5$ and $\nu = 0.3$. As shown in Fig. 20c, we clamp part of the jaw ($\boldsymbol{u} = \boldsymbol{0}$ on Γ_D), and we apply traction downwards on the lower half of the handle with a magnitude of 1.0. All the remaining part of the boundary is traction free, i.e., homogeneous Neumann boundary condition. This way, singularities (in stresses) are expected in the transition regions from the clamped displacement condition to the traction-free condition; see the red-spotted regions in Fig. 20c. We further "locally" refine the mesh to capture such features. The von
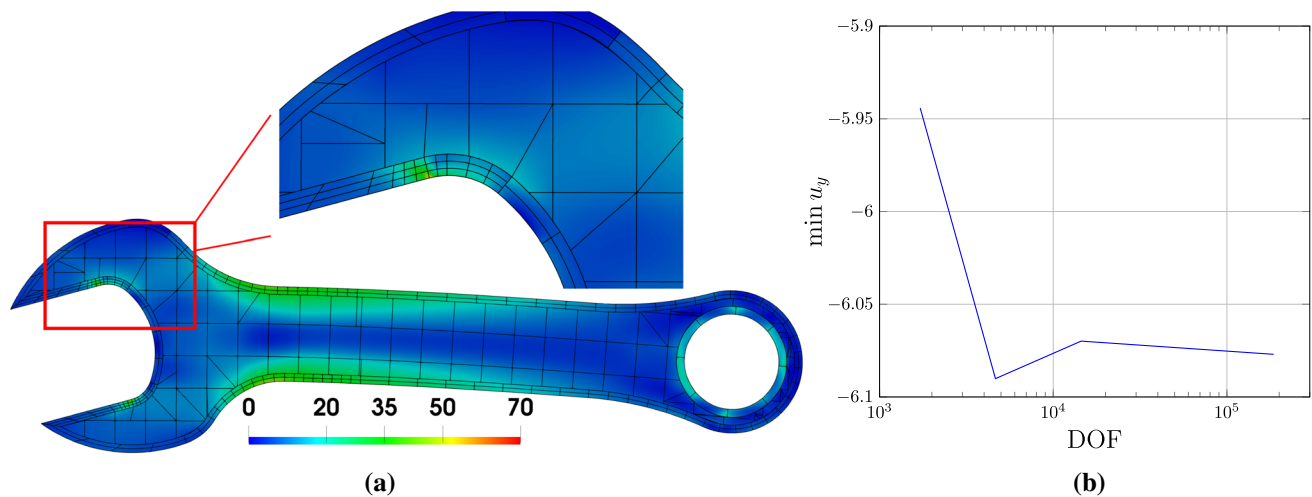
Fig. 21 Results of the spanner model. **a** The von-Mises stress on the IBCM representation of the spanner, and **b** the convergence plot of the maximum downwards displacement ($-y$ direction). In **a**, the mesh resolution is (manually) adapted to large stresses. In the zoomed-in figure of **a**, some part of the mesh in the conformal layer is not conformal. In the background mesh of **a**, integration cells of trimmed elements are used for visualization

Mises stress is shown in Fig. 21a. We observe that with the conformal layer, we can easily adapt the mesh resolution to the solution features. Moreover, when a conformal layer has a multi-patch representation, meshes in different patches do not need to be conformal across patch interfaces; see the zoomed picture in Fig. 21a. Non-conformal patch interfaces are coupled with Nitsche's method and are treated as a special case of the general overlapping construction. This way, mesh refinement is localized to the patch level.

Moreover, we check the convergence of the maximum downwards displacement (the $-y$ direction), or min u_y, with a series of globally refined meshes. The convergence plot is shown in Fig. 21b with respect to DOF, where we observe a convergent result. Here we study the convergence in displacements rather than stresses because in this example, stresses exhibit singularity due to the boundary conditions, which we do on purpose to show how IBCM can be used to flexibly capture such a solution feature.

4.6 Fiber-reinforced composite

Inspired by [64], we perform stress analysis on a fiber-reinforced composite. The test is mainly aimed to show the capability of IBCM to represent complex material interfaces with conformal discretizations. A typical cross section is studied under the plane strain assumption; see Fig. 22. The matrix material is modeled as a square that occupies the domain $[-5, 5] \times [-5, 5]$. Then multiple circular fibers of unit radii are randomly positioned in the matrix, provided that none of them overlaps with another. For each circular material interface, a pair of conformal annuli is obtained following the interface-type construction, where one has the

material property of fibers and the other has the property of the matrix. Every fiber has an independent background mesh, whereas the background mesh of the matrix is cut by multiple annuli. Combining all the annuli and the active parts of background meshes yields the IBCM representation for the fiber-reinforced composite.

We solve the linear elasticity problem on the resulting geometric model. Uniform tension $T_x = 100$ is imposed on the right boundary of the matrix, whereas $u_x = 0$ and $u_y = 0$ are imposed on the left and bottom boundaries, respectively. Homogeneous Neumann boundary conditions are applied elsewhere. Material constants are given as follows: $E_{\text{fiber}} = 100$, $\nu_{\text{fiber}} = 0.33$, and $E_{\text{matrix}} = 1$, $\nu_{\text{matrix}} = 0.3$. As a result of discontinuous material data, stress discontinuities are expected across material interfaces.

Starting from the input mesh in Fig. 22, we obtain a series of globally refined meshes for a convergence study. The distribution of the stress σ_{xx} is shown in Fig. 23a, which is computed with the mesh after one refinement. As expected, large stresses occur in fibers as well as in the narrow bands between fibers. The convergence plot is shown in Fig. 23b. In each mesh, we compute the maximum σ_{xx} (the $x - x$ stress) among all the sampled points, and we check the convergence of this quantity. A relative error is evaluated using an overkill solution, which is obtained with the mesh after five times of refinement. We observe that the error converges roughly at an expected rate of two.

4.7 Advection–diffusion problem

As the last example, we push a step forward to study IBCM beyond the elliptic problems. It is motivated by the singu-
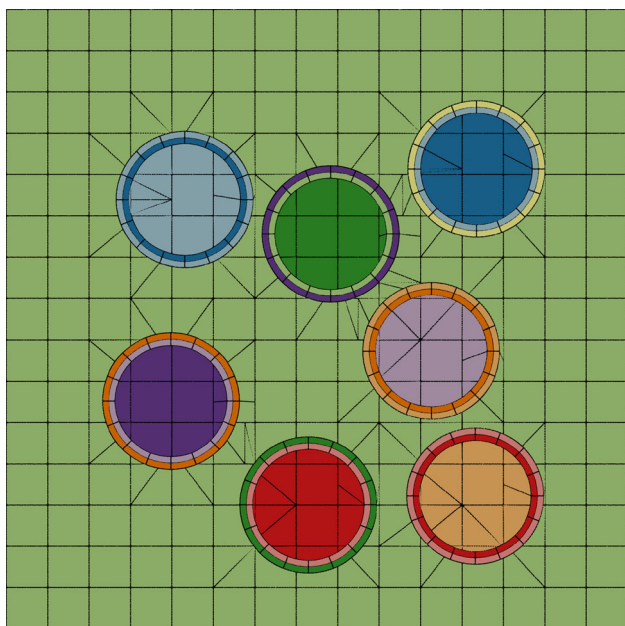
Fig. 22 The initial IBCM representation of the fiber-reinforced composite, where the circular fibers are randomly positioned and different patches are represented by different colors. In the background mesh, integration cells of trimmed elements are used for visualization

larly perturbed model problem [65], but here we will adopt a simplified boundary condition. The underlying partial differential equation (PDE) is an advection–diffusion equation, which is fundamentally different from those in the previous examples because of the advection term. It states as follows. Given $f : \Omega \to \mathbb{R}$ and $g_D : \Gamma_D \to \mathbb{R}$, find $u : \Omega \to \mathbb{R}$ such that

$$
\begin{cases}
-\epsilon \Delta u + \boldsymbol{a} \cdot \nabla u = f & \text{in} \quad \Omega = \Omega^t \cup \Omega_a^b, \\
u^t - u^b = 0 & \text{on} \quad \Gamma = \partial \Omega^t \cap \Omega^b, \\
\nabla u^t \cdot \boldsymbol{n}^t + \nabla u^b \cdot \boldsymbol{n}^b = 0 & \text{on} \quad \Gamma = \partial \Omega^t \cap \Omega^b, \\
u = g_D & \text{on} \quad \Gamma_D \equiv \partial \Omega,
\end{cases}
\tag{30}
$$

where $\epsilon \in \mathbb{R}$ and $\boldsymbol{a} \in \mathbb{R}^2$ are a diffusivity constant and a velocity constant, respectively, and the other notations (including those in the following) are the same as those in Eq. (8). Note that we only consider the Dirichlet boundary condition in this problem.

We follow [66] to obtain the corresponding weak formulation. $\partial \Omega$ is divided into the outflow boundary $\partial \Omega^+ = \{ \boldsymbol{x} \in \partial \Omega : \boldsymbol{a} \cdot \boldsymbol{n} \geq 0 \}$ and the inflow boundary $\partial \Omega^- = \{ \boldsymbol{x} \in \partial \Omega : \boldsymbol{a} \cdot \boldsymbol{n} < 0 \}$, where $\boldsymbol{n}$ is the outward normal of $\partial \Omega$. The discrete weak formulation states as follows: Find $u_h \in \mathcal{V}_h^{g_D}$ such that

$$
a_h(u_h, v_h) = l(v_h), \quad \forall v_h \in \mathcal{V}_h^0,
\tag{31}
$$
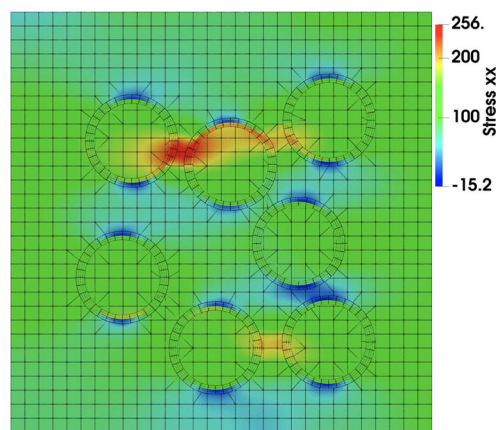
(a)
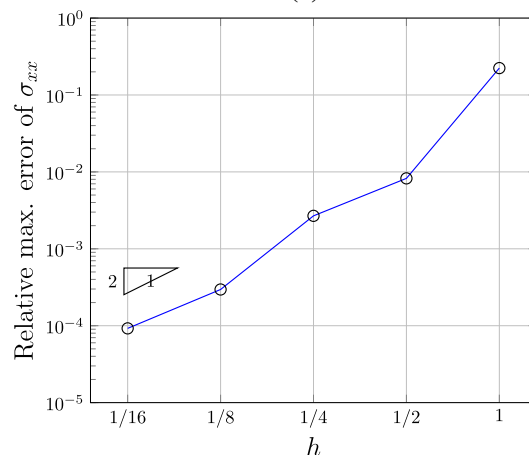
(b)

Fig. 23 Results of the fiber-reinforced composite. **a** The stress σ_{xx} computed on the mesh after one refinement, and **b** the convergence plot of σ_{xx}. In the background mesh of **a**, integration cells of trimmed elements are used for visualization

where

$$
\begin{aligned}
a_h(u_h, v_h) = {} & \epsilon \int_\Omega \nabla u_h \cdot \nabla v_h - \int_\Omega u_h \, \boldsymbol{a} \cdot \nabla v_h \\
& - \epsilon \int_\Gamma \langle \nabla u_h \cdot \boldsymbol{n}^t \rangle [v_h] - \epsilon \int_\Gamma \langle \nabla v_h \cdot \boldsymbol{n}^t \rangle [u_h] \\
& + \epsilon \beta (h_t^{-1} + h_b^{-1}) \int_\Gamma [u_h][v_h], \\
& + \int_\Gamma \mathrm{up}(u_h) \, [v_h] \, \boldsymbol{a} \cdot \boldsymbol{n}^t \\
& + \int_{\partial \Omega^+} u_h v_h \, \boldsymbol{a} \cdot \boldsymbol{n} + \epsilon \beta h^{-1} \int_{\partial \Omega} u_h v_h
\end{aligned}
\tag{32}
$$

and

$$
l(v_h) = \int_\Omega f \, v_h + \epsilon \beta h^{-1} \int_{\partial \Omega} g_D \, v_h - \int_{\partial \Omega^-} g_D \, v_h \, \boldsymbol{a} \cdot \boldsymbol{n},
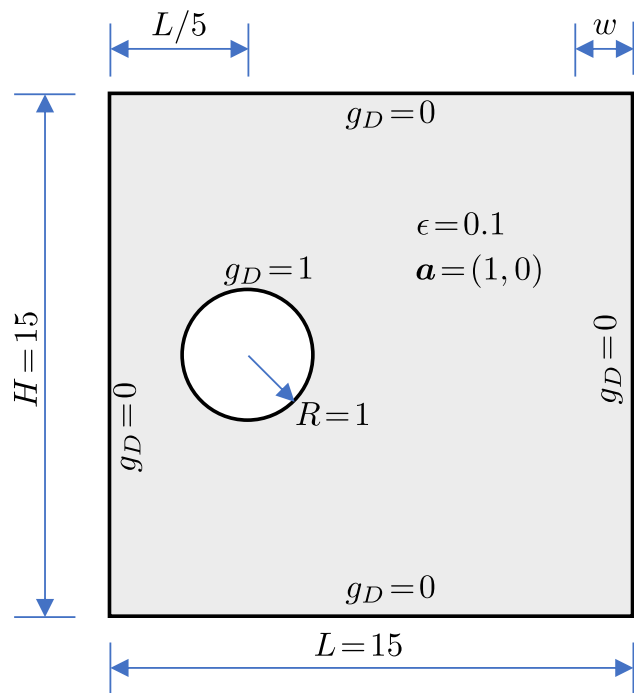\tag{33}
$$

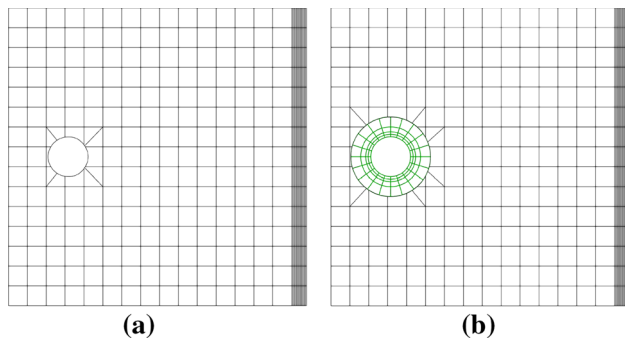**Fig. 24** The geometry and problem setting for the advection–diffusion problem

Fig. 25 The initial Bézier meshes of the plate with a hole via trimming (**a**) and IBCM (**b**). In the background mesh, integration cells of trimmed elements are used for visualization

where

$$h(\boldsymbol{x}) = \begin{cases} h_t & \text{if } \boldsymbol{x} \in \partial\Omega \cap \partial\Omega^t, \\ h_b & \text{if } \boldsymbol{x} \in \partial\Omega \cap \partial\Omega^b, \end{cases} \tag{34}$$

and

$$\text{up}(u_h(\boldsymbol{x})) = \begin{cases} u^t(\boldsymbol{x}) & \text{if } \boldsymbol{a} \cdot \boldsymbol{n}^t(\boldsymbol{x}) \geq 0, \\ u^b(\boldsymbol{x}) & \text{otherwise.} \end{cases} \tag{35}$$

Eq. (35) represents an *upwind* scheme that takes certain quantity of interest from the upwind side. Note that the normal of the interface Γ is $\boldsymbol{n}^t$, i.e., the outward normal of the top domain Ω^t.

(a) Mesh 0 (DOF: 544) **(b)** Mesh 1 (DOF: 1984)

(c) Mesh 2 (DOF: 7548) **(d)** Mesh 3 (DOF: 29402)

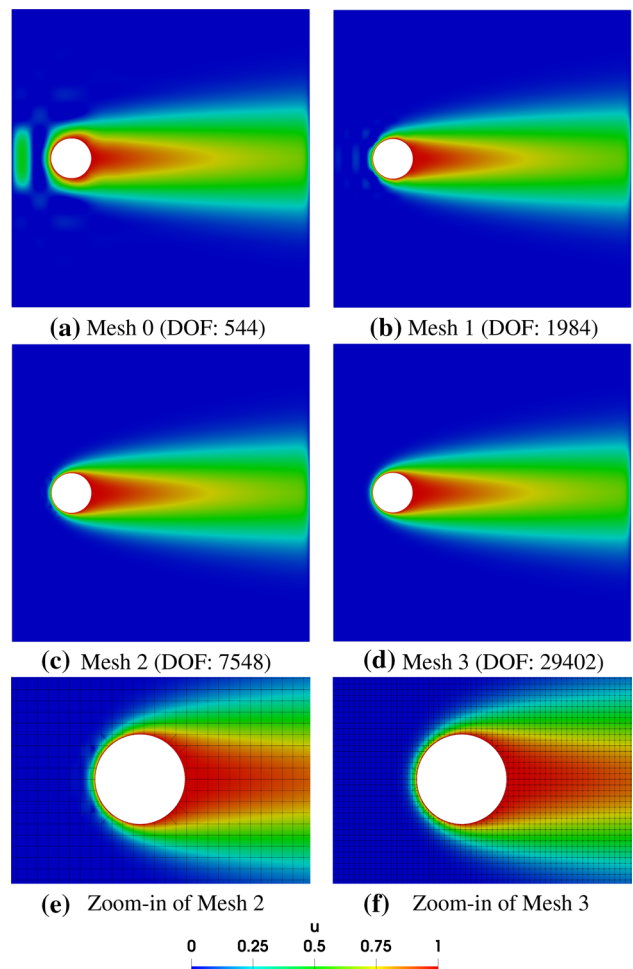**(e)** Zoom-in of Mesh 2 **(f)** Zoom-in of Mesh 3

Fig. 26 Simulation results of the advection–diffusion problem using trimming. In the background meshes of (e, f), integration cells of trimmed elements are used for visualization

As shown in Fig. 24, the geometry of interest is a plate with a hole. The plate has the dimension $L \times H$ where $L = H = 15$. The hole is centered at $(L/5, H/2)$ with a radius of $R = 1$. The diffusivity is set to be $\epsilon = 0.1$ to make the problem advection-dominated, and a constant rightward velocity of $\boldsymbol{a} = (1, 0)$ is adopted. According to Eq. (32), we *weakly* impose the Dirichlet boundary condition on all the boundaries. More specifically, we impose $g_D = 1$ on the hole and $g_D = 0$ on the other boundaries. The boundary-layer phenomenon is expected around the hole and the right boundary, where dense meshes are needed to resolve it. It is straightforward to "locally" refine the mesh near the right boundary by adding more knot lines. Therefore, we focus on the treatment of the hole, where we compare the performance of trimming and IBCM.

We first study the geometric representation obtained via trimming. The initial Bézier mesh is shown in Fig. 25a, where a dense mesh of width $w = 0.05L$ (see Fig. 24) is adopted near the right boundary. We solve the advection–diffusion
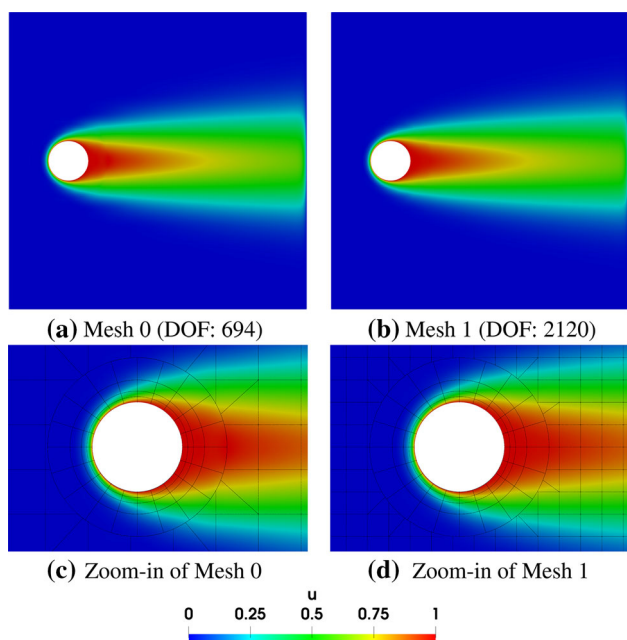
(a) Mesh 0 (DOF: 694) **(b)** Mesh 1 (DOF: 2120)

(c) Zoom-in of Mesh 0 **(d)** Zoom-in of Mesh 1

Fig. 27 Simulation results of the advection–diffusion problem using IBCM. In the background meshes of (c, d), integration cells of trimmed elements are used for visualization
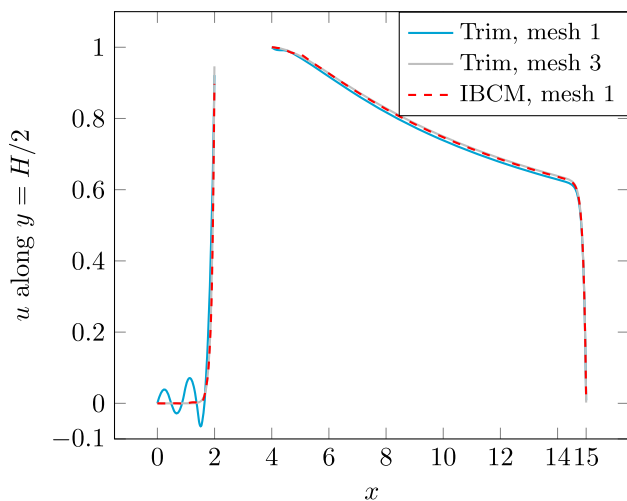

Fig. 28 The solution u over the horizontal line $y = H/2$. The region [2, 4] represents the hole

problem on a series of globally refined meshes. The results are shown in Fig. 26. We observe that the high-quality solution is obtained only when the mesh size is sufficiently small with respect to the boundary layer; see Fig. 26f.

On other other hand, we can add a conformal layer around the hole to enhance the solution. The initial Bézier mesh is shown in Fig. 25b, where the thickness of the conformal layer is 1. As the conformal layer aligns with the hole, we can easily control the mesh resolution around the hole. Even though the background mesh is the same as that in Fig. 25a, the solution is significantly improved; see Fig. 27a. Moreover,

with the background mesh globally refined just once and the conformal layer unchanged, the solution using IBCM is already comparable to the best result using trimming; see Figs. 27b and 26d. In other words, with IBCM, we can use much fewer DOF ($< 1/10$) to resolve the boundary-layer phenomenon.

Moreover, we plot the solution field along a horizontal line $y = H/2$; see Fig. 28. Note that the region [2, 4] represents the hole. Fig. 28 shows three results: trimming with Mesh 1 (Fig. 26b), trimming with Mesh 3 (Fig. 26d), and IBCM with Mesh 1 (Fig. 27b). We thus quantitatively confirm that IBCM with Mesh 1 achieves a very similar result to trimming with Mesh 3, whereas trimming with Mesh 1 exhibits oscillation near the hole as the mesh resolution is not fine enough.

5 Conclusions and future work

In this paper, we have presented an immersed boundary-conformal method (IBCM) to leverage advantages of both conformal discretization and immersed methods. This is enabled by analysis-aware treatment of trimming and union operations as well as sophisticated construction of conformal layers. To efficiently and robustly deal with cut elements, we present an enhanced decomposition method to reduce the number of quadrature cells needed for a cut element. The union operation weakly couples independent domains through Nitsche's method. With the one-sided flux from the non-trimmed conformal layer, no further treatment is needed regarding stabilization.

On the other hand, the key to constructing a conformal layer lies largely in creating a target curve for an input boundary curve. The target curve can be constructed either as an offset curve or as a curve that has a very different shape from the input. Offset curves seem to be a natural choice, but very often they end up with cusps and self-intersections, and moreover, the parameterization is generally different from the input. Therefore, it requires further repairing and lofting to be able to construct a desired conformal layer. Fortunately, it is flexible for IBCM to choose the shape of the target curve, and there is numerical evidence that the shape does not play a significant role in the solution accuracy. With this flexibility, a target curve that has the same parameterization as the input can be constructed, and thus the conformal layer is readily available.

Two types of geometric constructions with IBCM are presented: a boundary type and an interface type. Both types can be used to represent complex geometric and/or solution features. In a boundary-type IBCM representation, as it features a conformal discretization near the boundary, many benefits are carried over from boundary-fitted methods, such as the intuitive control of mesh resolution and the ability to strongly impose Dirichlet boundary conditions.

In an interface-type representation, kinematic constraints on the interface are automatically satisfied. Various benchmark problems are present to show the improved accuracy of IBCM compared to other means of geometric representations, such as solely trimming or union. There also exists evidence that the results are not sensitive to the thickness and shape of a conformal layer, but further study is needed to conclude. Moreover, two examples with complex geometric features are presented to show the flexibility of IBCM in representing complex geometries, strongly imposing Dirichlet boundary conditions, as well as easy adaptation of mesh resolution to solution features. In the end, we push a step forward to study an advection–diffusion problem with IBCM, where we have shown that IBCM can efficiently resolve the boundary-layer phenomenon near general geometric features in an intuitive manner.

In the future, on top of all potential directions is the extension to 3D. This is indeed challenging as it requires a robust analysis-aware treatment for trimming and union. Constructing conformal layers for surfaces also becomes much more involving. On the other hand, extending IBCM to other problems is also promising, such as the Stokes problem, nonlinear problems, and shell/plate problems, where coming up with stabilized formulations poses as the most challenging problem.

References

1. Antolin P, Buffa A, Puppi R, Wei X (2021) Overlapping multipatch isogeometric method with minimal stabilization. SIAM J Sci Comput 43(1):A330–A354

2. Hughes TJR, Cottrell JA, Bazilevs Y (2005) Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Comput Methods Appl Mech Eng 194(39):4135–4195

3. Cottrell JA, Hughes TJR, Bazilevs Y (2009) Isogeometric analysis: toward integration of CAD and FEA. Wiley, Hoboken

4. Martin T, Cohen E, Kirby RM (2009) Volumetric parameterization and trivariate B-spline fitting using harmonic functions. Comput Aided Geom Des 26(6):648–664

5. Zhang Y, Wang W, Hughes TJR (2013) Conformal solid T-spline construction from boundary T-spline representations. Comput Mech 51(6):1051–1059

6. Rank E, Ruess M, Kollmannsberger S, Schillinger D, Düster A (2012) Geometric modeling, isogeometric analysis and the finite cell method. Comput Methods Appl Mech Eng 249–252:104–115

7. Rüberg T, Cirak F (2012) Subdivision-stabilised immersed B-spline finite elements for moving boundary flows. Comput Methods Appl Mech Eng 209–212:266–283

8. Schillinger D, Dedè L, Scott MA, Evans JA, Borden MJ, Rank E, Hughes TJR (2012) An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces. Comput Methods Appl Mech Eng 249–252:116–150

9. Kamensky D, Hsu M-C, Schillinger D, Evans JA, Aggarwal A, Bazilevs Y, Sacks MS, Hughes TJR (2015) An immersogeometric variational framework for fluid-structure interaction: application to bioprosthetic heart valves. Comput Methods Appl Mech Eng 284:1005–1053

10. Hoang T, Verhoosel CV, Qin C-Z, Auricchio F, Reali A, van Brummelen EH (2019) Skeleton-stabilized immersogeometric analysis for incompressible viscous flow problems. Comput Methods Appl Mech Eng 344:421–450

11. Casquero H, Bona-Casas C, Toshniwal D, Hughes TJR, Gomez H, Zhang YJ (2020) The divergence-conforming immersed boundary method: application to vesicle and capsule dynamics. arXiv:2001.08244

12. Li Z, Ito K (2006) The immersed interface method: numerical solutions of PDEs involving interfaces and irregular domains. SIAM, Philadelphia

13. Kim H, Seo Y, Youn S (2009) Isogeometric analysis for trimmed cad surfaces. Comput Methods Appl Mech Eng 198(37):2982–2995

14. Nagy AP, Benson DJ (2015) On the numerical integration of trimmed isogeometric elements. Comput Methods Appl Mech Eng 284:165–185

15. Kudela L, Zander N, Bog T, Kollmannsberger S, Rank E (2015) Efficient and accurate numerical quadrature for immersed boundary methods. Adv Model Simul Eng Sci 2:1–22

16. Antolin P, Buffa A, Martinelli M (2019) Isogeometric analysis on V-reps: first results. Comput Methods Appl Mech Eng 355:976–1002

17. Massarwi F, Antolin P, Elber G (2019) Volumetric untrimming: precise decomposition of trimmed trivariates into tensor products. Comput Aided Geom Des 71:1–15

18. Marussig B, Zechner J, Beer G, Fries TP (2017) Stable isogeometric analysis of trimmed geometries. Comput Methods Appl Mech Eng 316:497–521

19. Marussig B, Hiemstra R, Hughes TJR (2018) Improved conditioning of isogeometric analysis matrices for trimmed geometries. Comput Methods Appl Mech Eng 334:79–110

20. Guo Y, Heller J, Hughes TJR, Ruess M, Schillinger D (2018) Variationally consistent isogeometric analysis of trimmed thin shells at finite deformations, based on the STEP exchange format. Comput Methods Appl Mech Eng 336:39–79

21. Buffa A, Puppi R, Vázquez R (2020) A minimal stabilization procedure for isogeometric methods on trimmed geometries. SIAM J Numer Anal 58(5):2711–2735

22. Bazilevs Y, Hughes TJR (2007) Weak imposition of Dirichlet boundary conditions in fluid mechanics. Comput Fluids 36(1):12–26

23. Embar A, Dolbow J, Harari I (2010) Imposing Dirichlet boundary conditions with Nitsche's method and spline-based finite elements. Int J Numer Methods Eng 83(7):877–898

24. Benek JA, Steger JL, Dougherty FC (1983) A flexible grid embedding technique with application to the Euler equations. AIAA J 373–382

25. Liou M-S, Kao K-H (1994) Progress in grid generation: from Chimera to DRAGON grids. NASA Technical Memorandum 106709

26. Henshaw WD (1994) A fourth-order accurate method for the incompressible Navier–Stokes equations on overlapping grids. J Comput Phys 113(1):13–25

27. Tang HS, Jones SC, Sotiropoulos F (2003) An overset-grid method for 3D unsteady incompressible flows. J Comput Phys 191(2):567–600

28. Kannan R, Wang ZJ (2007) Overset adaptive Cartesian/prism grid method for stationary and moving-boundary flow problems. AIAA J 45(7):1774–1779

29. Duan Z, Wang ZJ (2020) High-order overset flux reconstruction method for dynamic moving grids. AIAA J 58(10):4534–4547

30. Bouclier R, Passieux J, Salaün M (2016) Local enrichment of NURBS patches using a non-intrusive coupling strategy: geometric details, local refinement, inclusion, fracture. Comput Methods Appl Mech Eng 300:1–26

31. Dokken JS, Funke SW, Johansson A, Schmidt S (2019) Shape optimization using the finite element method on multiple meshes with Nitsche coupling. SIAM J Sci Comput 41(3):A1923–A1948

32. Becker R, Burman E, Hansbo P (2011) A hierarchical NXFEM for fictitious domain simulations. Int J Numer Methods Eng 86(4–5):549–559

33. Johansson A, Kehlet B, Larson MG, Logg A (2019) Multimesh finite element methods: solving PDEs on multiple intersecting meshes. Comput Methods Appl Mech Eng 343:672–689

34. Piegl L, Tiller W (1997) The NURBS book, 2nd edn. Springer, Berlin

35. Marussig B, Hughes TJR (2018) A review of trimming in isogeometric analysis: challenges, data exchange and simulation aspects. Arch Comput Methods Eng 25:1059–1127

36. Lorensen WE, Cline HE (1987) Marching cubes: a high resolution 3D surface construction algorithm. ACM SIGGRAPH Comput Graph 21(4):163–169

37. Fries TP, Omerović S, Schöllhammer D, Steidl J (2017) Higher-order meshing of implicit geometries—part I: Integration and interpolation in cut elements. Comput Methods Appl Mech Eng 313:759–784

38. Fries TP, Schöllhammer D (2017) Higher-order meshing of implicit geometries, part II: approximations on manifolds. Comput Methods Appl Mech Eng 326:270–297

39. Nitsche J (1971) Über ein variationsprinzip zur lösung von dirichlet-problemen bei verwendung von teilräumen, die keinen randbedingungen unterworfen sind. Abh Math Semin Univ Hambg 36:9–15

40. de Prenter F, Verhoosel CV, van Zwieten GJ, van Brummelen EH (2017) Condition number analysis and preconditioning of the finite cell method. Comput Methods Appl Mech Eng 316:297–327

41. Hansbo P (2005) Nitsche's method for interface problems in computational mechanics. GAMM-Mitteilungen 28:183–206

42. Hoschek J, Lasser D (1992) Grundlagen der geometrischen Datenverarbeitung. Vieweg+Teubner

43. Patrikalakis NM, Maekawa T (2009) Shape interrogation for computer aided design and manufacturing. Springer, Berlin

44. Maekawa T (1999) An overview of offset curves and surfaces. Comput Aided Des 31(3):165–173

45. Pham B (1992) Offset curves and surfaces: a brief survey. Comput Aided Des 24(4):223–229

46. Wallner J, Sakkalis T, Maekawa T, Pottmann H, Yu G (2001) Self-intersections of offset curves and surfaces. Int J Shape Model 07(01):1–21

47. Rhino - Rhinoceros 3D (2021). https://www.rhino3d.com

48. Hinz J, Müller M, Vuik C (2018) Elliptic grid generation techniques in the framework of isogeometric analysis applications. Comput Aided Geom Des 65:48–75

49. Randrianarivony M (2006) Geometric processing of CAD data and meshes as input of integral equation solvers. PhD thesis, Computer Science Faculty Technische Universität Chemnitz

50. Bommes D, Campen M, Ebke H-C, Alliez P, Kobbelt L (2013) Integer-grid maps for reliable quad meshing. ACM Trans Graph 32(4):98:1–98:12

51. Sederberg TW, Zheng J, Bakenov A, Nasri A (2003) T-splines and T-NURCCs. ACM Trans Graph 22(3):477–484

52. Beirão da Veiga L, Buffa A, Cho D, Sangalli G (2012) Analysis-suitable T-splines are dual-compatible. Comput Methods Appl Mech Eng 249–252:42–51

53. Zhang J, Li X (2018) Local refinement for analysis-suitable++ T-splines. Comput Methods Appl Mech Eng 342:32–45

54. Casquero H, Wei X, Toshniwal D, Li A, Hughes TJR, Kiendl J, Zhang YJ (2020) Seamless integration of design and Kirchhoff–Love shell analysis using analysis-suitable unstructured T-splines. Comput Methods Appl Mech Eng 360:112765

55. Vuong A-V, Giannelli C, Jüttler B, Simeon B (2011) A hierarchical approach to adaptive local refinement in isogeometric analysis. Comput Methods Appl Mech Eng 200(49):3554–3567

56. Giannelli C, Jüttler B, Speleers H (2012) THB-splines: the truncated basis for hierarchical splines. Comput Aided Geom Des 29(7):485–498

57. Wei X, Zhang Y, Hughes TJR, Scott MA (2015) Truncated hierarchical Catmull–Clark subdivision with local refinement. Comput Methods Appl Mech Eng 291:1–20

58. Buffa A, Giannelli C (2016) Adaptive isogeometric methods with hierarchical splines: error estimator and convergence. Math Models Methods Appl Sci 26(1):1–25

59. Bracco C, Buffa A, Giannelli C, Vázquez R (2019) Adaptive isogeometric methods with hierarchical splines: an overview. Discrete Contin Dyn Syst A 39(1):241–261

60. Johannessen KA, Kvamsdal T, Dokken T (2014) Isogeometric analysis using lr b-splines. Comput Methods Appl Mech Eng 269:471–514

61. Patrizi F, Manni C, Pelosi F, Speleers H (2020) Adaptive refinement with locally linearly independent lr b-splines: theory and applications. Comput Methods Appl Mech Eng 369:113230

62. Sukumar N, Chopp DL, Moës N, Belytschko T (2001) Modeling holes and inclusions by level sets in the extended finite-element method. Comput Methods Appl Mech Eng 190(46):6183–6200

63. Stanford JW, Fries TP (2019) A higher-order conformal decomposition finite element method for plane B-rep geometries. Comput Struct 214:15–27

64. Herráez M, González C, Lopes CS, Guzmán de Villoria R, LLorca J, Varela T, Sánchez J (2016) Computational micromechanics evaluation of the effect of fibre shape on the transverse strength of unidirectional composites: an approach to virtual materials design. Compos Part A Appl Sci Manuf 91:484–492

65. Hemker PW (1996) A singularly perturbed model problem for numerical computation. J Comput Appl Math 76(1):277–285

66. Buffa A, Hughes TJR, Sangalli G (2006) Analysis of a multiscale discontinuous Galerkin method for convection–diffusion problems. SIAM J Numer Anal 44(4):1420–1440