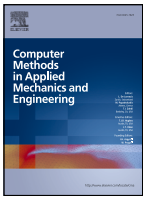
Computer Methods in Applied Mechanics and Engineering 446 (2025) 118269

Contents lists available at ScienceDirect

Comput. Methods Appl. Mech. Engrg.

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



A high-order immersed finite-difference discretization for solving linear and nonlinear elasticity problems

James Gabbard, Wim M. van Rees *

Department of Mechanical Engineering, Massachusetts Institute of Technology, 77 Masschusetts Ave., Cambridge, 02139, MA, United States

ARTICLE INFO

Keywords:
Immersed method
High-order methods
Elasticity
Large strains
Mesh-free

ABSTRACT

This work presents a high order immersed finite difference method to discretize 2D linear and nonlinear elasticity problems on regular Cartesian grids. Our demonstrations show up to sixth order convergence for the displacement field and boundary traction distribution in 2D discretizations of linear and nonlinear elasticity, which incorporate displacement or traction boundary conditions, material interfaces, and spatially-variable and discontinuous material properties. To demonstrate geometric flexibility the convergence results are obtained with non-convex test geometries, and both the linear and nonlinear elasticity discretizations are applied successfully to a complex lattice structure. Lastly, we demonstrate the ability of our method to match, locally, the accuracy of a finite element simulation on intricate single- and multi-material elasticity problems. The ability to generate high-fidelity results without manual mesh generation opens opportunities in optimization and data-driven machine learning, as well as physical applications such as elastodynamics and fluid-structure interactions.

1. Introduction

The predominant approach to solving linear and nonlinear elastic problems in complex domains is to use body fitted meshes. Body fitted meshes provide a simple treatment of domain boundary and material interface conditions, since these conditions can be applied directly to element edges and discretization nodes. Though well established, the process of meshing complex geometries can still provide challenges that require manual intervention. This is especially relevant for high order techniques applied to domains with curved boundaries, and also applies to multiply connected complex domains and multiscale structures. Furthermore, with advances in optimization and machine learning, simulation pipelines are increasingly required that robustly solve problems within parametrically defined domains in a fully automated manner.

To avoid the need for generating body-fitted meshes, a wide range of immersed geometry methods have been proposed. In solid mechanics, the vast majority of these fictitious or extended domain methods are based on the finite element techniques. Prominent examples include the finite cell method (FCM), which introduces a fictitious stiffness in the non-physical part of the domain [1,2]; the Cut Finite Element Method (CutFEM), which employs a stabilized finite element formulation on elements intersected by an immersed boundary, typically using ghost penalty terms to ensure stability and consistency [3,4]; and the extended and generalized finite element methods (XFEM/GFEM), which enrich the approximation space to capture discontinuities or singularities without conforming meshes [5–8]. Many other techniques exist, and we refer to reviews provided in [2,9] for a more exhaustive list. As noted in [9], the main challenges in these methods are the numerical integration of discretized fields over cut/enriched elements; the

* Corresponding author.
E-mail address: wvanrees@mit.edu (W.M. van Rees).

https://doi.org/10.1016/j.cma.2025.118269
Received 30 March 2025; Received in revised form 20 June 2025; Accepted 24 July 2025

deterioration in conditioning of the linear system associated with small cells; and the enforcement of essential boundary conditions on immersed boundaries and interfaces. Though the focus of this work is on a finite difference based method, we briefly discuss below these main challenges in the finite element community to motivate the proposition of an alternative approach.

For numerical integration, subdivision of cut-cells is typically employed to achieve effective but low-order quadrature in the vicinity of discontinuities [2,4]. Local quad- or octree-based refinement approaches can then be used to reduce the integration error through spatial refinement [1,2,10], whose efficiency can be improved through boundary-conforming subdivision [11]. In the FCM, such refinement approaches to reducing quadrature errors is combined with a high order FEM discretization to yield high order convergence of the scheme overall. However, as highlighted in [12], refinement techniques can result in a significant increase in the number of sub-cells, especially in three dimensions. Alternatively, the immersed geometry can be integrated using subtriangulation [4] or other interface reconstruction approaches, typically relying on a level-set representation. High-order versions of such quadrature schemes have been proposed by [13,14] and [15,16], which account for the curvature of the boundary but require involved geometric reconstruction efforts that can sacrifice robustness. Overall, constructing robust, high-order quadrature schemes remains a persisting challenge in immersed finite element methods, motivating the development of alternative approaches [17].

A separate issue with immersed finite element methods is that the presence of small cells typically increases the condition number of the system of equations associated with the discretized problem. This is no different from the cut-cell finite volume community, where the 'small-cell' problem has received extensive attention in the past [18]. For the finite cell method, the fictitious stiffness parameter can be scaled to improve stability; nevertheless, [2] notes that poor conditioning still prevents the use of iterative solvers. In CutFEM, stabilization is typically achieved using a ghost penalty technique, which adds a user-defined penalty parameter that scales a stabilization term acting on neighboring cut elements [4,19]. An extensive and recent review on further conditioning approaches is provided in [9].

To impose Dirichlet (essential) boundary conditions, weak enforcement approaches are typically used. The original FCM relied on a simple but robust penalization approach [1], approximating the Dirichlet condition using a penalization term that balanced stability and accuracy. Nitsche's method provides a more consistent manner to enforce Dirichlet boundary conditions [9,20], but introduces a per-cell parameter that can, theoretically, become arbitrarily large on cut-cells [21]. To mitigate this, stabilization as in the ghost penalty or non-symmetric approaches [22,23] can be used. The review in [9] provides a state-of-the-art review of stabilization and conditioning issues in immersed finite element methods.

Overall, immersed finite element methods have seen tremendous development to maturation, and are now broadly applied in scientific research across domains. However, issues pertaining to extension to high order, stability, and efficiency persist, motivating continuing development and improvement. In particular, the development of formally high order approaches that combine robustness and simplicity of implementation is still an area of active research.

Immersed finite difference/volume based methods offer an alternative to the finite element approach, and face their own set of challenges. These methods discretize the strong form of the equations, allowing explicit treatment of boundary and jump conditions on embedded geometries in the discrete differential operators. Notably, [24] demonstrated a finite difference/volume method to tackle single material linear elasticity problems with Dirichlet (displacement) or Neumann (stress) boundary conditions imposed on immersed boundaries. The method exhibits second order accuracy and treats bodies with homogeneous, isotropic material properties. Results were demonstrated in both 2D and 3D domains, and the utility of the approach was proven using a shape optimization method. In [25,26] the authors propose a second order accurate Immersed Interface Method to handle multimaterial elasticity problem with imposed jump conditions on immersed interfaces. The approach was developed in 2D for isotropic materials with piecewise-constant material parameters. In [27], a second order finite difference method was presented with largely the same characteristics as [25], but extended to tackle inhomogeneous material properties – the method was extended to 3D in [28]. In [29], a second order generalized finite difference method was developed for 2D linear elasticity problems with piecewise-constant material properties, separated by an immersed interface on which displacement and traction jump conditions are enforced.

Compared to immersed finite element methods, finite difference approaches avoid the challenges related to numerical quadrature and the imposition of essential boundary conditions, while conditioning challenges are typically easily avoided as well. However, the above finite difference/volume works share some common limitations compared to the capabilities of prevalent immersed finite element methods. First, these existing methods impose either domain boundary conditions (Dirichlet, Neumann [24]) or material interface conditions [25–29] on the immersed geometry, but no method has demonstrated the ability to handle both. Second, the elastic models considered have been constrained to isotropic materials, with only [27] treating nonhomogeneous material properties. Third, all discretizations are second order accurate. Finally, all approaches are restricted to linear elastic problems, and hence small deformations. In this work we address all four restrictions through a high order immersed interface discretization of linear and nonlinear elasticity problems. Our method is based on a discretization of the Laplacian that relies on previous work on parabolic and elliptic partial differential equations [30–32], but is extended here to cross-derivative terms with variable coefficients. Our approach has no limitations on the constitutive law and thus allows for anistropic materials. We demonstrate up to sixth order convergence of the solution and boundary tractions in the infinity norm, while representing our geometry fully locally and only through grid intersections and associated normal vectors. Further, we extend our approach to finite deformation elasticity with hyperelastic materials while retaining high order convergence. We show the ability of our solver to simulate challenging lattice structure elasticity problems, and compare the accuracy of our approach with a commercial finite element solver.

The rest of this manuscript is structured as follows. In Section 2 we recall the constant coefficient discretization of the scalar Poisson equation presented in [32]. The discretization is extended to variable coefficients in Section 3. These stencils form the basis of our linear elasticity discretization as explained in Section 4. The extension to nonlinear elasticity is discussed in Section 5. Though each of the sections above contain individual convergence and verification results, we show further applications of our methodology

in Section 6, including a comparison with a finite element solver. We discuss the relative merits of our method in Section 7, and provide conclusions in Section 8.

2. Discretization of the constant-coefficient scalar Poisson equation

In this section, we briefly summarize the results of [32] presenting our high-order immersed discretization of the scalar Poisson equation. We first consider the treatment of the constant coefficient Poisson equation with immersed domain boundaries on which Dirichlet and/or Neumann boundary conditions are imposed. Then, we discuss the piecewise-constant coefficient Poisson equation with immersed interfaces on which jump conditions are imposed.

2.1. Discretization of immersed boundaries

For immersed *boundary* problems, we consider the Poisson equation posed in domain Ω^+ with Dirichlet or Neumann conditions prescribed on the boundary, so that the PDE and boundary conditions are

$$\begin{aligned} \nabla \cdot (\beta \nabla u) &= f \text{ in } \Omega^+, \\ u &= \bar{u} \text{ on } \Gamma_D, \\ \beta \partial_n u &= \bar{q} \text{ on } \Gamma_N, \end{aligned} \tag{1}$$

where $\Gamma = \Gamma_D \cup \Gamma_N$ is the boundary of domain Ω^+. Here the prescribed boundary conditions and the source term $f(\mathbf{x})$ are assumed to be smooth functions, and the coefficient β is assumed constant. The discretization of the Laplace operator follows the methodology outlined in [30,31], which is briefly summarized here.

For interior points, the Laplacian operator $\nabla^2 = \sum_{i=1}^{d} \partial_i^2$ is discretized at each grid point using dimension splitting, with the standard central centered finite difference stencil used to discretize the second derivative ∂_i^2 along each coordinate axis. Near immersed boundaries an interpolation procedure incorporating boundary information is used to extend the solution, providing ghost values for the finite difference scheme.

The boundary treatments used in this work are based on an evolution of the original immersed interface method [33,34], as presented in [30,35] and outlined below. Following a convention from the immersed interface literature, we refer to the intersection between a 1D finite difference stencil and the surface Γ as a *control point*, denoted $\mathbf{x}_c$. The evaluation point for this stencil is referred to as an *affected point*, since the discretization is affected by the presence of the surface.

In our approach, each control point $\mathbf{x}_c$ on the immersed domain boundary is associated with a set of interpolation points $\mathcal{X}_c^+ \subset \Omega^+$, and with a polynomial $p_c(\mathbf{x})$ of degree k that approximately interpolates the domain values $\{u(\mathbf{x}_i) \mid \mathbf{x}_i \in \mathcal{X}_c^+\}$ in a least squares sense. Each 1D finite difference stencil that intersects the boundary at $\mathbf{x}_c$ is applied not directly to the function $u(\mathbf{x})$, but to the extended function

$$u_c(\mathbf{x}) = \begin{cases} u(\mathbf{x}), & \mathbf{x} \in \Omega^+ \\ p_c(\mathbf{x}), & \mathbf{x} \notin \Omega^+ \end{cases} \tag{2}$$

As defined in [31], the set of interpolation points in the least squares domain $\mathcal{X}_c^+$ includes the control point, excludes the closest grid point, and includes all other grid points that are (1) part of the domain Ω^+, and (2) fall within a half-elliptical region centered on the boundary whose semi-major axis is aligned with the local normal vector to the surface (Fig. 1(a)). These interpolants are suitable for all boundary conditions and any smooth geometry satisfying a well-defined curvature constraint [31]. In this work, for any given polynomial order we choose the major and minor axes of the region so that we can guarantee the existence of $p_c(\mathbf{x})$ on a grid with spacing Δx as long as the immersed surface satisfies

$$|\kappa \Delta x| < 1/4, \tag{3}$$

where κ is the maximum scalar curvature of the surface.

Each of the 1D finite difference stencils which intersect the boundary at $\mathbf{x}_c$ can require up to w ghost values at grid points $\{\mathbf{x}_g\}$ that fall outside of the domain. These ghost values are obtained by evaluating $p_c(\mathbf{x})$ through w separate stencil operations with coefficients $\{s_c^g\} \cup \{s_\alpha^g\}_{\alpha=1}^n$ [31], so that

$$p_c(\mathbf{x}_g) = s_c^g u(\mathbf{x}_c) + \sum_{\alpha=1}^{n} s_\alpha^g u(\mathbf{x}_\alpha) \tag{4}$$

at each point $\mathbf{x}_g$ requiring a ghost value. For points with Neumann boundary conditions $u(\mathbf{x}_c)$ is not directly available, but it can be approximated based on the boundary condition $\bar{q}(\mathbf{x}_c)$ and nearby solution values. To clarify, let $\{s_c\} \cup \{s_i\}_{i=1}^n$ be a set of stencil coefficients that approximate the normal derivative of p_c at $\mathbf{x}_c$, so that

$$\partial_n u(\mathbf{x}_c) = s_c u(\mathbf{x}_c) + \sum_{i=1}^{n} s_i u(\mathbf{x}_i) + \mathcal{O}(\Delta x^{k-1}). \tag{5}$$

When a Neumann condition is prescribed at a control point, Eq. (5) can be inverted to give

$$u(\mathbf{x}_c) = \frac{1}{s_c}\left(\frac{\bar{q}(\mathbf{x}_c)}{\beta} - \sum_{i=1}^{N-1} s_i u(\mathbf{x}_i) \right) + \mathcal{O}(\Delta x^k). \tag{6}$$

This requires one additional set of stencil coefficients which evaluate the normal derivative $\partial_n p(\mathbf{x}_c)$ on the boundary.

2.2. Discretization of immersed interfaces

For problems with immersed interfaces, we consider the Poisson equation defined over two subdomains Ω^+ and Ω^- with piecewise constant coefficients β^+ and β^-:

$$\beta(\mathbf{x}) = \begin{cases} \beta^+, & \mathbf{x} \in \Omega^+ \\ \beta^-, & \mathbf{x} \in \Omega^- \end{cases}. \tag{7}$$

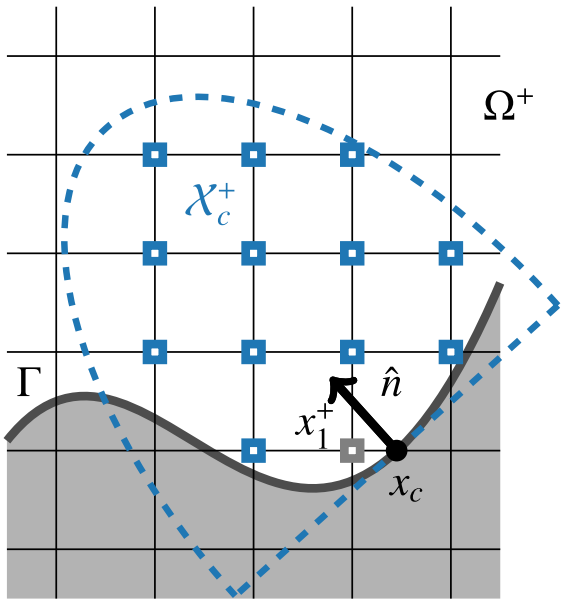
Jump conditions on the solution and its flux are prescribed on the interface between the subdomains, Γ_M, so that the full problem becomes

$$\begin{aligned} \nabla \cdot (\beta \nabla u) &= f \text{ in } \Omega, \\ [u] &= j_0(\mathbf{s}) \text{ on } \Gamma_M, \\ [\beta \partial_n u] &= j_1(\mathbf{s}) \text{ on } \Gamma_M. \end{aligned} \tag{8}$$

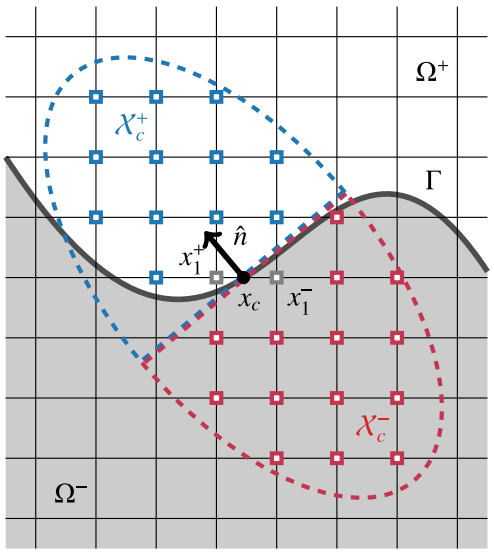
Here $[g] = g^+ - g^-$ denotes the jump of function g across the interface, defined as the difference between g^+, the limit of $g(\mathbf{x})$ approaching the interface from the positive side, and g^-, the limit of $g(\mathbf{x})$ approaching the interface from the negative side. Positive and negative sides are here associated with the sign of the level-set field so that surface normals point into the positive side. In classical elasticity, these jumps are typically zero to enforce continuity of displacement and tractions; examples of non-zero jumps can arise when considering electromechanics, slip, or active interfaces. The source term $f(\mathbf{x})$ is also allowed to be discontinuous across the material interface, though this has minimal effect on the solution procedure.

For piecewise constant $\beta(\mathbf{x})$, the operator $\nabla \cdot (\beta \nabla u)$ reduces to $\beta \nabla^2 u$ away from Γ_M, so that the standard dimension-split discretization of interior points remains valid. To discretize the jump boundary conditions, the boundary values $u^-(\mathbf{x}_c)$ and $u^+(\mathbf{x}_c)$ from either side of the interface are computed with the aid of two sets of stencils coefficients [30,31], associated with half-elliptical regions as shown in Fig. 1(b). The first set $\{s_c^+, s_i^+\}$ maps the boundary value $u^+(\mathbf{x}_c)$ and solution values from Ω^+ to the normal derivative $\partial_n u^+(\mathbf{x}_c)$, while the second set $\{s_c^-, s_i^-\}$ is designed analogously to map solution values from Ω^- to the normal derivative $\partial_n u^-(\mathbf{x}_c)$. The boundary values $u^\pm(\mathbf{x}_c)$ can then be determined from the discretized jump conditions $u^+(\mathbf{x}_c) - u^-(\mathbf{x}_c) = j_0(\mathbf{x}_c)$ and

$$\beta^+ \left(s_c^+ u^+(\mathbf{x}_c) + \sum_{i=1}^{n^+} s_i^+ u(\mathbf{x}_i^+) \right) - \beta^- \left(s_c^- u^-(\mathbf{x}_c) + \sum_{i=1}^{n^-} s_i^- u(\mathbf{x}_i^-) \right) = j_1(\mathbf{x}_c). \tag{9}$$



(a) Multidimensional interpolant for immersed boundaries.



(b) Multidimensional interpolants for immersed interfaces.

Fig. 1. Each crossing between a grid line and the boundary (x_c) is used to construct ghosts points for an affected grid point (light grey) using a multidimensional interpolant constructed from a half-elliptical region of grid points. For immersed boundaries (a), the interpolant is constructed by incorporating imposed boundary conditions (Dirichlet, Neumann). For immersed interfaces (b), interpolants on both sides are constructed by incorporating imposed jump conditions.

For $j_0(\mathbf{x}_c) = j_1(\mathbf{x}_c) = 0$, the closed-form solution to Eq. (9) is

$$u^-(\mathbf{x}_c) = u^+(\mathbf{x}_c) = \bar{u} \text{ with } \bar{u} = -\frac{\beta^+ \sum_{i=1}^{n^+} s_i^+ u(\mathbf{x}_i^+) - \beta^- \sum_{i=1}^{n^-} s_i^- u(\mathbf{x}_i^-)}{\beta^+ s_c^+ - \beta^- s_c^-}. \tag{10}$$

When jumps are present, the boundary values instead are given separately by

$$u^+(\mathbf{x}_c) = \bar{u} + \frac{j_1 - \beta^- s_c^- j_0(\mathbf{x})}{\beta^+ s_c^+ - \beta^- s_c^-}, \quad u^-(\mathbf{x}_c) = \bar{u} + \frac{j_1 - \beta^+ s_c^+ j_0(\mathbf{x})}{\beta^+ s_c^+ - \beta^- s_c^-}. \tag{11}$$

Once determined, the boundary values $u^{\pm}(\mathbf{x}_c)$ can be used in stencil operations on either side of the interface. We note that Eq. (9) treats both sides of the interface on an even footing. Further, as the ratio β^-/β^+ tends to zero, Eq. (11) approaches to the well-behaved Neumann boundary treatment given the previous section. This indicates that the boundary treatment is robust to large jumps in coefficients and does not exhibit singular behavior when β^-/β^+ tends to zero or infinity.

3. Discretization of variable coefficient scalar Poisson equation

In our previous work and as discussed above, the coefficient $\beta(\mathbf{x})$ in the Poisson equation was restricted to be either constant or piecewise constant. When $\beta(\mathbf{x})$ is spatially variable, both the interior and boundary discretizations must be altered to maintain high order accuracy. This section introduces a novel variable-coefficient immersed interface discretization which reduces to the constant coefficient discretization when $\nabla\beta = \mathbf{0}$. We restrict our focus on the scalar Poisson equation, with extensions to a vector equation discussed in the next section.

For generality, the discretization assumes that $\beta(\mathbf{x})$ is defined only by its value at each grid point and at the control points. This allows the method to generalize easily to systems of nonlinear elliptic PDEs, in which β may depend on the solution and several other auxiliary fields. Collocating $\beta(\mathbf{x})$ with the solution $u(\mathbf{x})$ at each grid point also reduces the amount of geometry processing needed compared a staggered arrangement, and allows any boundary stencils to be applied to both solution values and coefficient values. This will be essential for the discretization introduced below.

3.1. Bilinear stencils for the variable coefficient operator

For interior points, the operator $\nabla \cdot (\beta \nabla u) = \sum_{i=1}^{d} \partial_i(\beta \partial_i u)$ is discretized via dimension splitting. It is possible to discretize the 1D operators $\partial_i(\beta \partial_i u)$ by replacing each first derivative with a centered difference stencil of width w. However, the resulting discrete operator has an unnecessarily large stencil width of $2w$, and does not reduce to the standard centered discretization of $\beta\nabla^2 u$ for constant β. Here we derive an alternative that maintains a total stencil width of w and reduces to the standard discretization for constant β. The derivation is similar to the variable-coefficient finite-volume discretization developed in [36] and applied to ice sheet modeling in [37].

Consider a 1D Cartesian grid with grid points $x_i = i\Delta x$, and let u_i and β_i indicate the value of the solution and coefficient at the i-th grid point. The operator $\partial_x(\beta \partial_x u)$ is invariant under the transformation $x \to -x$ and linear in both $u(x)$ and $\beta(x)$. Thus it is natural to require that a finite difference discretization of this operator is symmetric and bilinear in the solution $\{u_i\}$ and coefficient $\{\beta_i\}$. Fixing a stencil width w, we seek a discretization of the form

$$\partial_x(\beta \partial_x u)_i = \sum_{j,k=-w}^{w} G_{jk} \beta(x_{i+j}) u(x_{i+k}) + \mathcal{O}(\Delta x^{2w}), \tag{12}$$

where the bilinear stencil coefficients G_{ij} obey the symmetry condition $G_{i,j} = G_{-i,-j}$, and the order $2w$ error term is chosen to match accuracy of the centered second derivative stencil of the same width.

To determine the accuracy of the bilinear stencil, the solution and coefficient can be replaced by their Taylor expansions $\beta(x) = \sum_{p=0}^{\infty} \beta^{(p)} \frac{x^p}{p!}$ and $u(x) = \sum_{q=0}^{\infty} u^{(q)} \frac{x^q}{q!}$. Choosing $x_0 = 0$ as the evaluation point, the resulting expressions for both the continuous and discrete operators are

$$\partial_x(\beta \partial_x u) = \sum_{p,q=0}^{\infty} \frac{\beta^{(p)} u^{(q)}}{p!q!} q(p+q-1) x^{p+q-2}, \tag{13}$$

$$\partial_x(\beta \partial_x u) = \sum_{p,q=0}^{\infty} \frac{\beta^{(p)} u^{(q)}}{p!q!} \sum_{j,k=-w}^{w} G_{jk} x_j^p x_k^q. \tag{14}$$

To achieve the maximal order of accuracy $n = 2w$, Eqs. (13) and (14) must agree for all terms with $p + q - 2 < n$. After rescaling the $\{x_i\}$ to eliminate factors of Δx, this leads to the exactness conditions

$$\sum_{j,k=-w}^{w} G_{jk} j^p k^q = \begin{cases} q(p+q-1), & p+q=2, \\ 0, & \text{otherwise}, \end{cases} \tag{15}$$

which must hold for $p + q - 2 < n$ and $p, q \geq 0$. The symmetry condition implies that $\sum G_{jk} j^p k^q = (-1)^{p+q} \sum G_{jk} j^p k^q$, so that Eq. (15) is automatically satisfied when $p + q$ is odd. The remaining exactness conditions occur for $p + q$ even and form a system of $(w + 1)^2$ independent linear constraints on the coefficients G_{jk}. A quick accounting shows that there are $(2w + 1)(w + 1)$ unique coefficients in

the symmetric bilinear stencil, leading to a $w(w+1)$ dimensional space of stencils with width w and order $2w$. This is in contrast to the standard difference centered stencil for ∂_x^2 of width w and order $2w$, which is uniquely determined.

Before constructing any stencils G_{jk} which satisfy the order constraints, there are two points worth noting. First, when $\beta(x) = 1$, any symmetric bilinear stencil of order $2w$ reduces to the standard centered difference stencil of order $2w$ for the second derivative. To show this, define the sums $g_k = \sum_{j=-w}^{w} G_{jk}$, so that the application of the bilinear stencil with $\beta = 1$ reduces to $\partial_x^2 u_i = \sum_{k=-w}^{w} g_k u_k$. The exactness conditions from Eq. (15) with $p = 0$ reduce to constraints on g_k,

$$\sum_{k=-w}^{w} g_k k^q = \begin{cases} q(q-1), & q = 2, \\ 0, & \text{otherwise}, \end{cases} \tag{16}$$

which are only satisfied when g_k is the unique second derivative stencil of width w and order $2w$. This immediately implies the second point, which is that no bilinear stencil of width w can have an order of accuracy greater than $2w$, despite the $w(w+1)$ free parameters available. If one were to exist, it would reduce for $\beta = 1$ to a finite difference stencil of width w and accuracy greater than $2w$, which does not exist.

3.2. Two constructions for high order bilinear stencils

For immersed interface discretizations, we make use of two constructions for symmetric bilinear stencils of arbitrary even order $2w$. Both constructions are chosen because they generalize immediately to nonlinear operators of the form $\partial_x f(x, u, \partial_x u)$, which will be useful in later discretizations of nonlinear elasticity presented in Section 5.

For the first construction, let $\{D_{ij}\}_{j,k=-w}^{w}$ be the unique differentiation matrix that satisfies

$$\partial_x u_j = \sum_{k=-w}^{w} D_{jk} u_k + \mathcal{O}(\Delta x^{2w}) \text{ for } -w \le j \le w. \tag{17}$$

The j-th row of D_{jk} is a first derivative stencil of order $2w$ with an evaluation point offset by j grid points from the stencil center. To approximate $\partial_x(\beta \partial_x u)$, the solution values $\{u_k\}_{k=-w}^{w}$ are multiplied by D_{jk} to approximate the derivatives $\{\partial_x u_j\}_{j=-w}^{w}$ at each stencil point. These derivatives are then multiplied point-wise by the coefficient values $\{\beta_j\}_{j=-w}^{w}$, and the centered difference stencil $\{D_{0j}\}_{j=-w}^{w}$ is applied to the result. Choosing $x_0 = 0$ as the evalution point, the full approximation reads

$$\partial_x(\beta \partial_x u) = \sum_{j,k=-w}^{w} D_{0j} \beta_j (D_{jk} u_k) \quad \Rightarrow \quad G_{jk} = D_{0j} D_{jk}. \tag{18}$$

One can verify that this construction leads to a symmetric stencil G_{jk} satisfying the conditions for accuracy of order $2w$. For the nonlinear extension, pointwise multiplication by β_j is replaced by pointwise evaluation of the nonlinear function $f(x_j, u_j, \partial_x u_j)$. The resulting discretization is

$$\partial_x f(x, u, \partial_x u) = \sum_{j=-w}^{w} D_{0j} f\left(x_j, u_j, \sum_{k=-w}^{w} D_{jk} u_k \right). \tag{19}$$

A Taylor series analysis confirms that the nonlinear discretization also has accuracy of order $2w$ when q is smooth in all of its arguments.

The second construction is slightly more complex, but leads to a conservative finite difference scheme. Instead of discretizing $\partial_x(\beta \partial_x u)$ directly, the discrete operator is written as a difference of fluxes defined at the half grid points, so that

$$\partial_x(\beta \partial_x u)_i = \frac{(\beta \partial_x u)_{i+1/2} - (\beta \partial_x u)_{i-1/2}}{\Delta x} + \mathcal{O}(\Delta x^{2w}). \tag{20}$$

To achieve accuracy of order $2w$, each flux $(\beta \partial_x u)_{i-1/2}$ is constructed as a bilinear combination of the solution and coefficient values at the nearest $2w$ grid points. Let $\{Q_{jk}\}_{j,k=-w}^{w-1}$ be the unique differentiation matrix that satisfies

$$\partial_x u_j = \sum_{k=-w}^{w-1} Q_{jk} u_k + \mathcal{O}(\Delta x^{2w-1}), \quad -w \le j \le w-1, \tag{21}$$

so that multiplication by Q_{jk} transforms a vector of solution values $\{u_{i+j}\}_{j=-w}^{w-1}$ into a vector of approximate derivatives $\{\partial_x u_{i+j}\}_{j=-w}^{w-1}$. After obtaining the derivatives and multiplying pointwise by the coefficient β, the values $\{(\beta \partial_x u)_{i+j}\}_{j=-w}^{w-1}$ are interpolated to construct the flux value $(\beta \partial_x u)_{i-1/2}$. For a conservative finite difference scheme, the standard interpolation coefficients $\{I_j\}_{j=-w}^{w-1}$ are constructed to satisfy

$$u_{i-1/2} = \sum_{j=-w}^{w-1} I_j \bar{u}_{i+j} + \mathcal{O}(\Delta x^{2w}), \text{ with } \bar{u}_i = \int_{x_{i-1/2}}^{x_{i+1/2}} u(x) \mathrm{d}x. \tag{22}$$

This construction is introduced and justified in [38], with further analysis provided in [39]. The full bilinear flux for the conservative discretization can then be written

$$(\beta \partial_x u)_{i-1/2} = \sum_{j,k=-w}^{w-1} I_j Q_{jk} \beta_{i+j} u_{i+k}. \tag{23}$$

Although the derivative matrix Q_{jk} yields derivatives with order $2w-1$ accuracy at each interpolation point, the symmetry of the full bilinear flux stencil leads to order $2w$ accuracy for the operator $\partial_x(\beta\partial_x u)$. For a nonlinear operator the corresponding discretization is

$$\partial_x f(x,u,\partial_x u)_i = \frac{f_{i+1/2}-f_{i-1/2}}{\Delta x}, \text{ with } f_{i-1/2} = \sum_{j=-w}^{w-1} I_j f\left(x_j, u_j, \sum_{k=-w}^{w-1} Q_{jk} u_{i+k}\right). \tag{24}$$

As above, this discretization has accuracy of order $2w$ for smooth f. The conservative scheme is used to generate all results in this work, unless otherwise specified.

3.3. Immersed interface discretization with variable coefficients

Compared to the interior discretization, the modifications to the immersed interface boundary treatment with variable coefficients are minor. Each bilinear stencil which crosses a domain boundary or material interface requires ghost values for both $u(\mathbf{x})$ and $\beta(\mathbf{x})$. The ghost solution values are approximated via the usual boundary stencil operation. Because $\beta(\mathbf{x})$ is also defined at the grid points and the control points, the ghost coefficient values can be obtained using the same interpolation stencils. The bilinear stencil of order n has the same stencil width and order of accuracy as the standard centered difference stencil of order n, so that the variable coefficient discretization requires the same set of boundary stencil coefficients as its constant coefficient counterpart. Dirichlet boundary conditions are handled without modification, by incorporating known boundary data $\bar{u}(\mathbf{x}_c)$ into the stencil operation at each control point $\mathbf{x}_c$. Neumann conditions and jump conditions on the material interface are also handled in the usual way, using the local coefficient values $\beta(\mathbf{x}_c)$ or $\beta^{\pm}(\mathbf{x}_c)$ in Eqs. (6) and (11).

3.4. Verification for the variable-coefficient Poisson equation

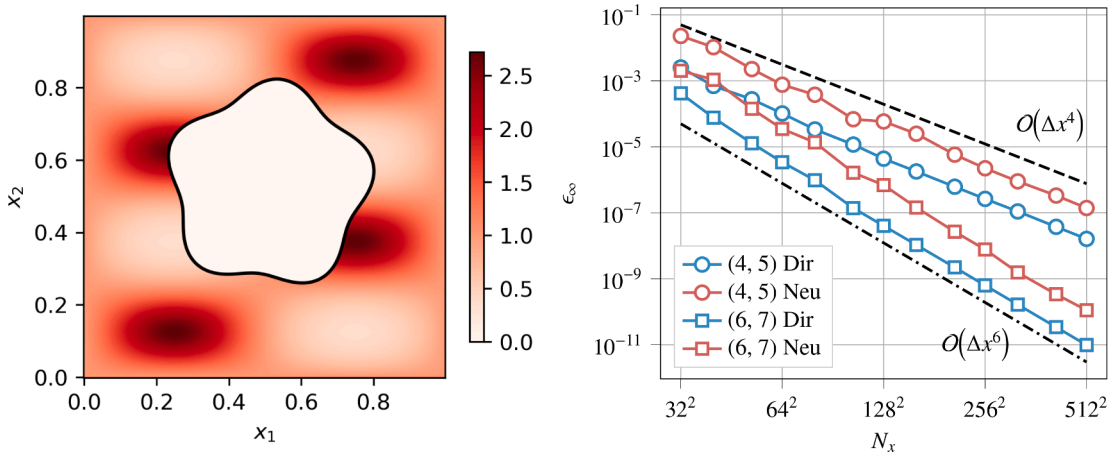
To verify the convergence of the variable-coefficient Poisson test case, we use the five-lobed star-shaped geometry used previously in [32]. We employ the following functional form of the manufactured solution and variable coefficient

$$u(\mathbf{x}) = \sin(4\pi x_1)\sin(2\pi x_2), \tag{25}$$

$$\beta(\mathbf{x}) = \exp(\sin(2\pi x_1)\sin(4\pi x_2)). \tag{26}$$

The geometry and the field $\beta(\mathbf{x})$ are illustrated in Fig. 2(a).

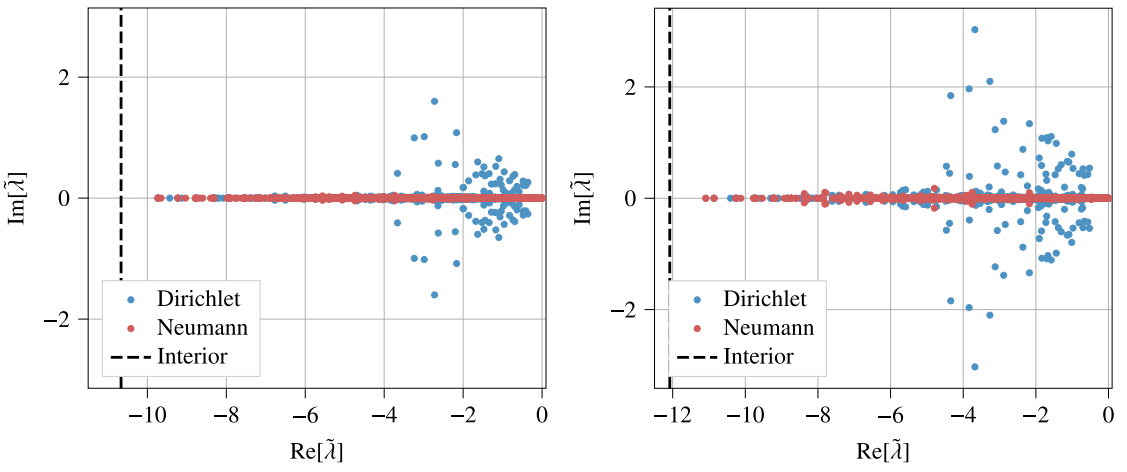
For brevity, we refer to a discretization with an n-th order interior stencil and k-th order boundary interpolants as an (n,k) discretization, and focus on high order interior discretizations with $n=4$ or $n=6$. Throughout this work we set $k=n+1$ rather than $k=n+2$, motivated by elliptical regularity effects. In [40] it was shown using potential theory that reduced truncation errors near cut cells with imposed boundary conditions do not deteriorate the overall solution error. For our method, we confirmed that the choice $k=n+1$ is indeed sufficient to achieve kth order convergence in the L_∞ norm for all boundary conditions considered here [30–32], both for parabolic and elliptic equations with constant coefficients. Fig. 2(b) plots the L_∞ error in the discrete solution for both (4, 5) and (6, 7) discretizations with Dirichlet or Neumann boundary conditions and a variable coefficient The results confirm that also with variable coefficients, the proposed discretizations converge with the expected order of accuracy, regardless of the type of boundary condition imposed.



(a) Variable coefficient test case.

(b) Solution error.

Fig. 2. Results for the scalar Poisson equation with a variable coefficient. (a) the geometry and variable coefficient for this test case. (b) L_∞ error in the discrete solution for (4, 5) and (6, 7) discretizations with Dirichlet or Neumann boundary conditions. The discretizations converge at the expected fourth or sixth order for either boundary condition.



(a) (Spectrum of the (4, 5) operator.

(b) Spectrum of the (6, 7) operator.

Fig. 3. Spectra of the (4, 5) and (6, 7) variable-coefficient discretizations at $N_x = 64$. Each plot shows the scaled eigenvalues $\tilde{\lambda} = \Delta x^2 \lambda / \beta_{\max}$. For both discretizations the real part of the scaled eigenvalues remains within the range predicted by the constant-coefficient interior discretization of the same order.

Since all our immersed geometry corrections are linear in the field values, the discretized form of the variable-coefficient Poisson equation together with the boundary conditions can be written as a linear system of equations for the unknown solution u at the grid points. Fig. 3 plots the spectra of the matrix associated with this linear system for each of the considered boundary conditions and discretization orders at resolution $N_x = 64$, with the eigenvalues scaled by a factor $\Delta x^2/\beta_{\max}$ where $\beta_{\max} = \max_{\mathbf{x}\in\Omega} \beta(\mathbf{x})$. Also shown is a dashed line indicating the most negative eigenvalue of interior discretization with a constant coefficient $\beta_{\max}$, determined using a standard von Neumann analysis. The eigenvalues of both the $(4, 5)$ and $(6, 7)$ discretizations have real parts of smaller magnitude, indicating that the boundary treatment does not suffer from a “small-cell" issue. Overall, the results indicate that the variable coefficient discretization achieves the same high order accuracy and well-behaved spectrum as the constant-coefficient case.

4. Linear elasticity

This section applies the variable coefficient Poisson discretization from Section 3 to the equations of linear elasticity, which involve a vector-valued unknown, cross-derivative terms, and more complex boundary and interface conditions.

4.1. Continuous formulation

For convenience, we adopt the Einstein summation convention for the remainder of this work, unless explicitly stated otherwise. Consider a domain Ω occupied by an elastic body, and let $u_i(\mathbf{x})$ represent the two components of the displacement field. Throughout the domain, a prescribed body force $\bar{f}_i$ acts on the elastic material. The linear elastic properties of the material are characterized by the spatially-variable rank-four stiffness tensor $C_{ijkl}(\mathbf{x})$, which may be discontinuous across a material interface Γ_M. Let $\sigma_{ij} = C_{ijkl}\partial_l u_k$ represent the Cauchy stress tensor in the material, and let $t_i = \sigma_{ij} n_j$ represent the traction vector on either the domain boundary or a material interface. To specify boundary conditions on the material, the domain boundary $\partial\Omega$ is partitioned into two sets Γ^D and Γ^N. On Γ^D a fixed displacement field $\bar{u}_i(\mathbf{s})$ is prescribed, while on Γ^N the traction $\bar{t}_i(\mathbf{s})$ is prescribed. Finally, on the material interfaces Γ^M both the jump in displacement $[\bar{u}_i]$ and jump in traction $[\bar{t}_i]$ are prescribed. Taken all together, the equilibrium equation for the body and the prescribed boundary conditions form the elliptic PDE system

$$\begin{aligned} \partial_j(C_{ijkl}\partial_l u_k) &= \bar{f}_i \text{ on } \Omega, \\ u_i &= \bar{u}_i \text{ on } \Gamma_D \\ C_{ijkl} n_j \partial_k u_l &= \bar{t}_i \text{ on } \Gamma_N \\ \left[u_i\right] &= \left[\bar{u}_i\right] \text{ on } \Gamma_M \\ \left[C_{ijkl} n_j \partial_l u_k\right] &= [\bar{t}_i] \text{ on } \Gamma_M. \end{aligned} \tag{27}$$

The stiffness tensor $C_{ijkl}(\mathbf{x})$ is assumed to obey the major symmetry $C_{ijkl} = C_{klij}$, the minor symmetry $C_{ijkl} = C_{jikl}$, and the positive definiteness property $C_{ijkl}\epsilon_{ij}\epsilon_{kl} > 0$ for all nonzero symmetric tensors ϵ_{ij}. The discretizations presented below are valid with no further assumptions on the stiffness tensor, to allow for anisotropic materials. However, for convergence tests we assume an isotropic but spatially variable stiffness tensor of the form

$$C_{ijkl} = \lambda(\mathbf{x})\delta_{ij}\delta_{kl} + \mu(\mathbf{x})\left(\delta_{ik}\delta_{jl} + \delta_{il}\delta_{kj}\right), \tag{28}$$

where $\lambda(\mathbf{x})$ and $\mu(\mathbf{x})$ are spatially-variable Lamé parameters.

4.2. Immersed interface discretization

On the whole, the discretization of the linear elastic system is a straightforward extension of the variable-coefficient Poisson discretization with a vector unknown $u_i(\mathbf{x})$ and tensor coefficient $C_{ijkl}(\mathbf{x})$. For fixed i and k, terms of the form $\partial_j(C_{ijkl}\partial_l u_k)$ with $j = l$ involve derivatives taken along the same axis, and are treated using the dimension-split variable coefficient discretization outlined in Section 3. The main numerical novelties are the treatment of cross derivative terms $\partial_j(C_{ijkl}\partial_l u_k)$ with $j \neq l$, the elimination of traction boundary conditions, and the elimination of material interface conditions, each of which is discussed separately below.

4.2.1. Cross derivative terms with variable coefficients

For notational simplicity, this section treats a scalar cross derivative term notated $\partial_j(\beta\partial_i u)$ with $i \neq j$. For linear elasticity, the resulting procedure is applied separately to discretize each term $\partial_j(C_{ijkl}\partial_l u_k)$ with $j \neq l$.

On the interior of the domain, cross derivative terms are discretized in a dimension-split manner. The inner derivative $\partial_i u$ is approximated using a centered difference stencil of width w and order $n = 2w$ applied along the x_i axis, and the result is multiplied by the coefficient $\beta(\mathbf{x})$ to form the field $\beta\partial_i u$. Finally, a centered difference stencil is applied to this field along the x_j axis to complete the discretization of $\partial_j(\beta\partial u_i)$. The result has order n accuracy, and requires an $n+1$ by $n+1$ block of solution values centered on the grid point along with $n+1$ coefficient values offset from the grid point along the x_j axis.

Near domain boundaries, the centered difference stencils for $\partial_i u$ intersect the domain boundary at control points located on x_i-direction grid lines. At these control points, ghost values of the solution $u(\mathbf{x})$ are constructed using values from the interpolation points in $\vec{\mathcal{X}}_c$, including the boundary value $u(\mathbf{x}_c)$. Incorporating ghost values into each centered difference results in a high order accurate gradient $\partial_i u$ at the affected points, which can be multiplied point-wise by the coefficient to yield $\beta\partial_i u$ at each affected point. To differentiate $\beta\partial_i u$ at the affected points, a second centered difference stencil is applied along the x_j axis. These stencils intersect the domain boundary at control points aligned with the x_j axis, and require ghost values for the field $\beta\partial_i u$. At each control point, ghost values for the gradient $\partial_i u$ are obtained evaluating by the derivative of the interpolating polynomial $p_c(\mathbf{x})$, using w sets of coefficients $\{s_c^{i,g}\} \cup \{s_\alpha^{i,g}\}_{\alpha=1}^n$ satisfying

$$\partial_i p_c(\mathbf{x}_g) = s_c^{i,g} u(\mathbf{x}_c) + \sum_{\alpha=1}^{n} s_\alpha^{i,g} u(\mathbf{x}_\alpha) \tag{29}$$

for each of the w required ghost points $\mathbf{x}_g$. Ghost values of the coefficient are obtained with the usual stencil operation, using the boundary values $\beta(\mathbf{x}_c)$ as well and interior values $\{\beta(\mathbf{x}_\alpha)\}_{\alpha=1}^n$. The results are multiplied point-wise to yield ghost values for the field $\beta\partial u_i$.

Eliminating traction boundary conditions

While similar in form to a Neumann boundary condition, the traction boundary condition $C_{ijkl}n_j\partial_l u_k = \bar{t}_i$ involves a vector unknown u_i. Consequently, obtaining the wall displacement value $u(\mathbf{x}_c)$ from the traction boundary condition (analogous to Eq. (6) for the Poisson equation) involves forming and inverting a small linear system at each control point. For $i = 1, 2$ let $\{s_c^i\} \cup \{s_\alpha^i\}_{\alpha=1}^n$ be a set of stencil coefficients defined so that $\partial_i u(\mathbf{x}_c) = s_c^i u(\mathbf{x}_c) + \sum_{\alpha=1}^N s_\alpha^i u(\mathbf{x}_\alpha) + \mathcal{O}(\Delta x^{k-1})$. In terms of these stencil coefficients, the traction boundary condition at each control point can be discretized as

$$C_{ijkl}n_j\left(s_c^l u_k(\mathbf{x}_c) + \sum_{\alpha=1}^{n} s_\alpha^l u_k(\mathbf{x}_\alpha)\right) = \bar{t}_i. \tag{30}$$

To simplify, define a pair of matrices $\mathbf{M}_l$ so that $(M_l)_{ik} = C_{ijkl}n_j$ for $l = 1, 2$. Rearranging Eq. (30) leads to the two by two linear system

$$(s_c^l\mathbf{M}_l)\mathbf{u}(\mathbf{x}_c) = \mathbf{t} - \sum_{\alpha=1}^{N}(s_\alpha^l\mathbf{M}_l)\mathbf{u}(\mathbf{x}_\alpha). \tag{31}$$

To express the boundary displacement explicitly as a function of the prescribed traction and interior displacement values, let $\mathbf{N}_c = (s_c^l\mathbf{M}_l)^{-1}$ and for $l = 1, 2$ let $\mathbf{N}_l = -\mathbf{N}_c\mathbf{M}_l$. Inverting Eq. (31) then gives

$$\mathbf{u}(\mathbf{x}_c) = \mathbf{N}_c\mathbf{t} + \sum_{\alpha=1}^{N}(s_\alpha^l\mathbf{N}_l)\mathbf{u}(\mathbf{x}_\alpha). \tag{32}$$

For the case of linear isotropic elasticity, the matrices $\mathbf{M}_l$ can be written explicitly as

$$\mathbf{M}_l = \lambda\mathbf{n}\otimes\mathbf{e}_l + \mu\big(\mathbf{e}_l\otimes\mathbf{n} + n_l\mathbf{I}\big), \tag{33}$$

where λ and μ are the Lamé parameters evaluated at the control point.

Eliminating material interface conditions

The techniques of the previous section can also be used to obtain wall displacement values $u_i^+(\mathbf{x}_c)$ and $u_i^-(\mathbf{x}_c)$ from the material interface conditions $[u_i] = [\bar{u}_i]$ and $[C_{ijkl}n_j u_{k,l}] = [\bar{t}_i]$, analogous to Eq. (11) for the Poisson equation. The displacement jump is written

explicitly as $u_i^+(\mathbf{x}_c) - u_i^-(\mathbf{x}_c) = [\bar{u}_i]$. The traction jump is discretized as

$$C^+_{ijkl} n_j \left(s_c^{l,+} u_k^+(\mathbf{x}_c) + \sum_{\alpha=1}^{n^+} s_\alpha^{l,+} u_k(\mathbf{x}_\alpha^+) \right) - C^-_{ijkl} n_j \left(s_c^{l,-} u_k^-(\mathbf{x}_c) + \sum_{\alpha=1}^{n^-} s_\alpha^{l,-} u_k(\mathbf{x}_\alpha^-) \right) = [\bar{t}_i]. \tag{34}$$

Define the matrices $\mathbf{M}_l^\pm$ as above, now a superscript to indicate the appropriate side of the interface. The jump conditions form a four by four system

$$\begin{bmatrix} \mathbf{I} & -\mathbf{I} \\ (s_c^{l,+}\mathbf{M}_l^+) & -(s_c^{l,-}\mathbf{M}_l^-) \end{bmatrix} \begin{bmatrix} \mathbf{u}^+(\mathbf{x}_c) \\ \mathbf{u}^-(\mathbf{s}_c) \end{bmatrix} = \begin{bmatrix} [\bar{\mathbf{u}}] \\ \tilde{\mathbf{t}} \end{bmatrix}, \tag{35}$$

where the right hand side vector $\tilde{\mathbf{t}}$ is defined by

$$\tilde{\mathbf{t}} = [\bar{\mathbf{t}}] - \left[\sum_{\alpha=1}^{n^+} (s_\alpha^{l,+}\mathbf{M}_l^+)\mathbf{u}(\mathbf{x}_\alpha^+) - \sum_{\alpha=1}^{n^-} (s_\alpha^{l,-}\mathbf{M}_l^-)\mathbf{u}(\mathbf{x}_\alpha^-) \right]. \tag{36}$$

To proceed, let $\mathbf{N}_c = (s_c^{l,+}\mathbf{M}_l^+ - s_c^{l,-}\mathbf{M}_l^-)^{-1}$ and for $l = 1, 2$ let $\mathbf{N}_l^\pm = -\mathbf{N}_c\mathbf{M}_l^\pm$. Using the matrix identity

$$\begin{bmatrix} \mathbf{I} & -\mathbf{I} \\ \mathbf{M}_1 & -\mathbf{M}_2 \end{bmatrix} \begin{bmatrix} (\mathbf{M}_1 - \mathbf{M}_2)^{-1} & \\ & (\mathbf{M}_1 - \mathbf{M}_2)^{-1} \end{bmatrix} \begin{bmatrix} -\mathbf{M}_1 & \mathbf{I} \\ -\mathbf{M}_2 & \mathbf{I} \end{bmatrix} = \begin{bmatrix} \mathbf{I} & \\ & \mathbf{I} \end{bmatrix}, \tag{37}$$

the above system can be inverted to yield

$$\mathbf{u}^+(\mathbf{x}_c) = \bar{\mathbf{u}} + \mathbf{N}_c[\bar{\mathbf{t}}] + (s_c^{l,-}\mathbf{N}_l^-)[\bar{\mathbf{u}}] \text{ and } \mathbf{u}^-(\mathbf{x}_c) = \bar{\mathbf{u}} + \mathbf{N}_c[\bar{\mathbf{t}}] + (s_c^{l,+}\mathbf{N}_l^+)[\bar{\mathbf{u}}], \tag{38}$$

where $\bar{\mathbf{u}}$ is a linear combination of the interior solution values given by

$$\bar{\mathbf{u}} = \sum_{\alpha=1}^{n^+} (s_\alpha^{l,+}\mathbf{N}_l^+)\mathbf{u}(\mathbf{x}_\alpha^+) - \sum_{\alpha=1}^{n^-} (s_\alpha^{l,-}\mathbf{N}_l^-)\mathbf{u}(\mathbf{x}_\alpha^-). \tag{39}$$

When the jumps $[\bar{\mathbf{u}}]$ and $[\bar{\mathbf{t}}]$ vanish, the solution reduces to $\mathbf{u}^+(\mathbf{x}_c) = \mathbf{u}^-(\mathbf{x}_c) = \bar{\mathbf{u}}$.

4.3. Verification for linear elasticity

As a test case for the linear elastic discretization, an elastic body is subjected to a prescribed deformation of the form

$$u_1(\mathbf{x}) = 0.04 \sin(4\pi x_1)\cos(2\pi x_2), \quad u_2(\mathbf{x}) = 0.04 \sin(4\pi x_1)\cos(6\pi x_2). \tag{40}$$

The body occupies the region between two immersed boundaries, both of which are star-shaped and defined by the same star-shaped level set as above. Both are centered on the point $\mathbf{x}_0 = [0.501, 0.502]$; the inner boundary has average radius $r_0 = 0.379$ and perturbation radius $\tilde{r} = 0.015$, while for the outer boundary $r_0 = 0.151$ and $\tilde{r} = 0.035$. The elastic body is isotropic with spatially varying Lamé parameters

$$\lambda(\mathbf{x}) = 1.5 + 0.5\cos^2(2\pi x_1)\cos^2(2\pi x_2), \quad \mu(\mathbf{x}) = 0.8 + 0.3\sin^2(2\pi x_1)\sin^2(2\pi x_2). \tag{41}$$

The body is immersed in a unit square domain of uniform resolution $N_x \times N_x$. The geometry and prescribed deformation for this test case are illustrated in Fig. 4.

Fig. 5(a) plots the L_∞ norm of the truncation error in the linear elastic operator $\partial_j(C_{ijkl}\partial_l u_k)$ as a function of the grid spacing Δx for both a (4, 5) and (6, 7) discretization. Results are shown for two separate test cases, one with displacement boundary conditions and one with traction boundary conditions. As for the Poisson test cases, these discretizations exhibit third and fifth order truncation error respectively for either set of boundary conditions. Fig. 5(b) plots the L_∞ error in the displacement field obtained by solving the discrete system, this time with a displacement boundary condition on the outer boundary and a traction boundary condition on the inner boundary. The plot also includes the L_∞ norm of the traction on the outer boundary, which is not prescribed but calculated from the discrete solution. The (4, 5) and (6, 7) discretizations achieve fourth and sixth order convergence respectively for both the displacement field and the boundary traction.

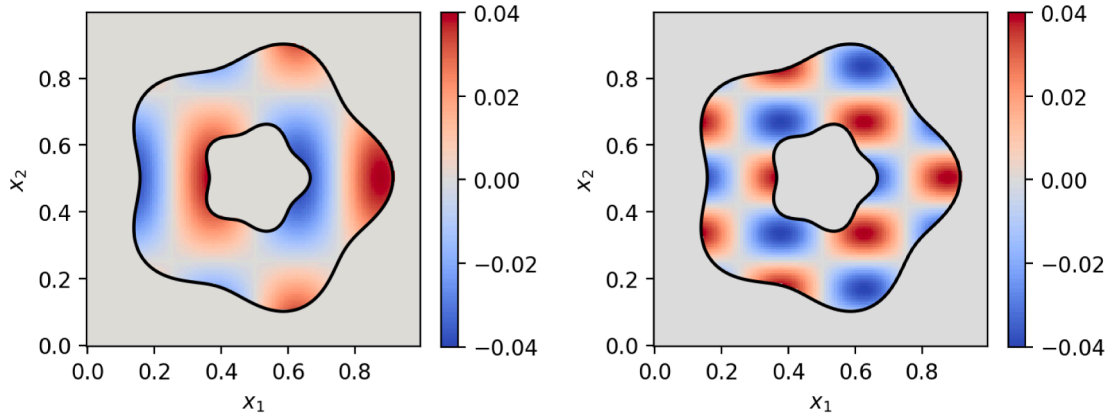
To test the discretization with material interfaces, a second region Ω^- is added inside the inner star-shaped boundary, and the average radii are adjusted to $r_0 = 0.221$ on the inner boundary and $r_0 = 0.419$ on the outer boundary. On the newly added inner region, the manufactured displacement field is

$$u_1^-(\mathbf{x}) = 0.02 \sin(2\pi x_1)\sin(2\pi x_2), \quad u_2^-(\mathbf{x}) = 0.028 \sin(2\pi x_1)\cos(4\pi x_2), \tag{42}$$

and the elastic body has an isotropic stiffness tensor with Lamé parameters

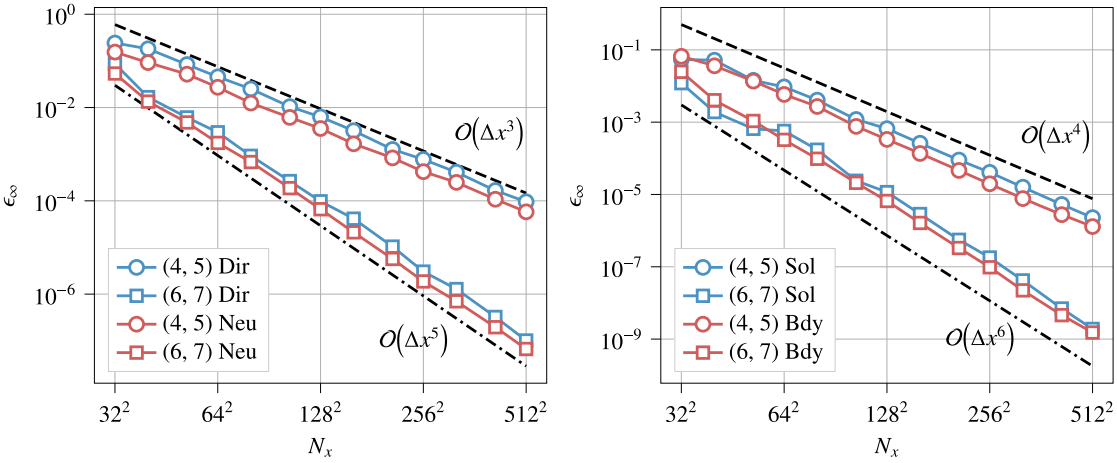
$$\lambda^-(\mathbf{x}) = 1.1 + 0.3\cos^2(4\pi x_1)\cos^2(2\pi x_2), \quad \mu^-(\mathbf{x}) = 0.6 + 0.7\sin^2(2\pi x_1)\cos^2(4\pi x_2). \tag{43}$$

For all tests a displacement boundary condition is prescribed on the outer boundary, while jumps in the displacement and traction are prescribed on the material interface. The geometry and prescribed displacement field for this test case are illustrated in Fig. 6. Fig. 7(a) plots the truncation error in the linear elastic operator with material interfaces, which is third order and fifth order for the (4, 5) and (6, 7) discretizations respectively. Fig. 7(b) plots the error in the displacement field obtained by solving the discrete system, as well as the traction distribution on the inner surface of the material interface. Both quantities converge with the expected fourth or sixth order accuracy.



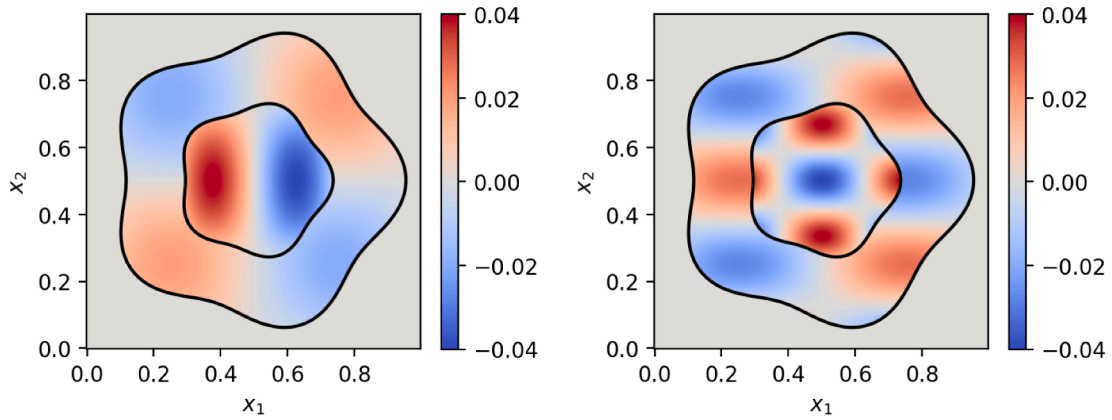
(a) $u_1(\mathbf{x})$. (b) $u_2(\mathbf{x})$.

Fig. 4. Manufactured solution used to test the linear elastic discretization.



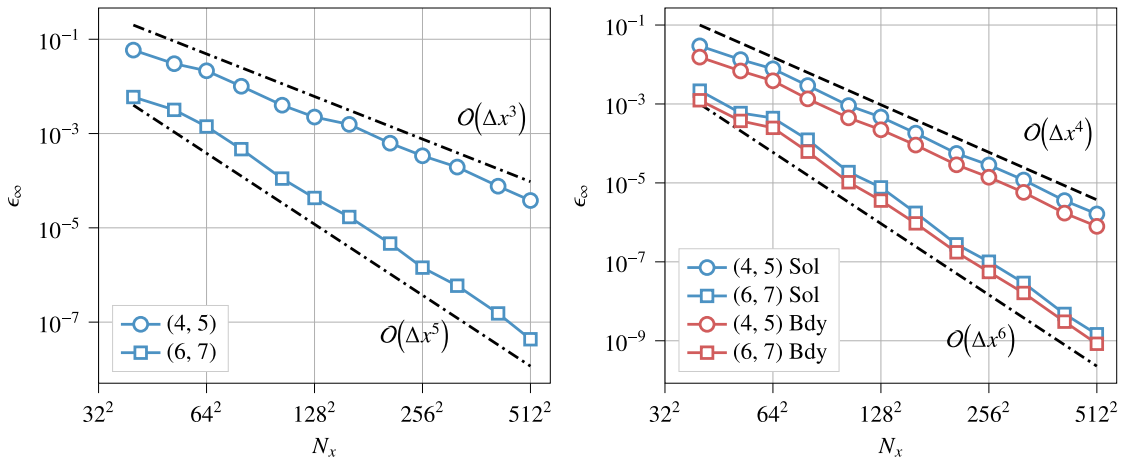
(a) Truncation error. (b) Solution error.

Fig. 5. Convergence of (a) the truncation error and (b) the solution error for a linear elastic test case. The (4, 5) and (6, 7) discretizations achieve fourth and sixth order convergence in the L_∞ norm for both the displacement field and boundary traction. The truncation error for each discretization converges at third and fifth order, respectively.



(a) $u_1(\mathbf{x})$. (b) $u_2(\mathbf{x})$.

Fig. 6. Manufactured solution used to test the linear elastic discretization with a material interface.



(a) Truncation error. (b) Solution error.

Fig. 7. Convergence of (a) the truncation error and (b) the solution error for a linear elastic test case with material interfaces. The addition of the material interface does not affect the convergence rates, which match those observed in Fig. 5.

5. Nonlinear elasticity

This final section presents a high order method for finite-deformation elasticity with hyperelastic materials. The discretization requires a nonlinear extension of the variable-coefficient and cross-derivative techniques presented in Sections 3 and 4, and an alternative approach to discretizing traction boundary conditions that is suitable for nonlinear problems.

5.1. Continuous formulation

For 2D nonlinear elastic problems, we consider a Lagrangian formulation. Let $y_i(\mathbf{x})$ be the location in the deformed configuration of the material point located at $\mathbf{x}$ in the reference configuration. The constitutive relation for a nonlinear hyperelastic material is defined by a nonlinear strain energy density function $W(\mathbf{F})$, where $F_{ij}(\mathbf{x}) = \partial_j y_i(\mathbf{x})$ is the deformation gradient. The stress in the interior of the reference configuration is defined by the first Piola-Kirchhoff stress tensor

$$S_{ij} = \frac{\partial W}{\partial F_{ij}}. \tag{44}$$

In terms of these quantities, the nonlinear system of PDEs governing the deformation field $y_i(\mathbf{x})$ is

$$\begin{aligned} \partial_j S_{ij}(\mathbf{F}) &= \bar{f}_i \text{ on } \Omega, \\ y_i &= \bar{y}_i \text{ on } \Gamma_D, \\ S_{ij}(\mathbf{F}) n_j &= \bar{t}_i \text{ on } \Gamma_N. \end{aligned} \tag{45}$$

At material interfaces, the displacement is assumed to be continuous, so that there is no tearing or cracking of the material. However, a traction jump is permitted, and the corresponding interface condition is

$$\left[S_{ij}(\mathbf{F}) n_j\right] = [\bar{t}_i] \text{ on } \Gamma_M. \tag{46}$$

This section considers only dead loading conditions, in which the prescribed body force $\bar{f}_i$ and the first Piola-Kirchhoff traction vector $\bar{t}_i$ prescribed on the boundary do not depend on the solution u_i.

As a nonlinear system of PDEs, Eq. (45) must be solved iteratively for the unknown displacements field $\mathbf{y}$ defined on the domain Ω and on the non-displacement boundaries $\Gamma_N \cup \Gamma_M$. To measure the convergence of the solution, define the domain residual field $\mathbf{r}^f$, boundary residual field $\mathbf{r}^t$, and material interface residual field $\mathbf{r}^{[t]}$ by

$$\begin{aligned} r_i^f &= \bar{f}_i - \partial_j S_{ij}(\mathbf{F}) \text{ on } \Omega, \\ r_i^t &= \bar{t}_i - S_{ij}(\mathbf{F}) n_j \text{ on } \Gamma_N, \\ r_i^{[t]} &= [\bar{t}_i] - \left[S_{ij}(\mathbf{F}) n_j\right] \text{ on } \Gamma_M. \end{aligned} \tag{47}$$

At each iteration of the nonlinear solve, Eq. (47) is linearized about the current solution, leading to terms that involve the elasticity tensor

$$A_{ijkl}(\mathbf{x}) \equiv \left.\frac{\partial S_{ij}(\mathbf{F})}{\partial F_{kl}}\right|_{\mathbf{F}(\mathbf{x})} = \left.\frac{\partial^2 W(\mathbf{F})}{\partial F_{ij}} F_{kl}\right|_{\mathbf{F}(\mathbf{x})}. \tag{48}$$

This tensor has the major symmetry $A_{ijkl} = A_{klij}$, but in general there is no minor symmetry, so that A_{ijkl} has ten independent components in 2D.

5.2. Interior discretization

Let $\mathbf{S}(\partial_1\mathbf{y}, \partial_2\mathbf{y})$ indicate the Piola-Kirchhoff stress calculated as a function of the two displacement derivatives $\partial_1\mathbf{y}$ and $\partial_2\mathbf{y}$. The fields $\partial_1\mathbf{y}$ and $\partial_2\mathbf{y}$ are calculated at the grid points using centered difference stencils of width w applied along the x_1 and x_2 axes respectively. To discretize $\partial_1 S_{11}(\partial_1\mathbf{y}, \partial_2\mathbf{y})$, the derivative $\partial_2\mathbf{y}$ is treated as a known quantity, while $\partial_1 S_{11}(\partial_1\mathbf{y}, \cdot)$ is treated as a nonlinear second derivative operator applied along the x_1 axis. Making use of the nonlinear variable-coefficient discretization proposed in Section 3.2 leads to the discretization

$$\partial_1 S_{11}(\mathbf{x}_{i,j}) = \sum_{k=-w}^{w} D_{0k} S_{11}\left(\sum_{l=-w}^{w} D_{k\ell}\mathbf{y}_{i+\ell,j}, \partial_2\mathbf{y}_{i+k,j} \right) + \mathcal{O}(\Delta x^{2w}). \tag{49}$$

The term $\partial_2 S_{12}$ is obtained by permuting the axes, so that $\partial_1\mathbf{y}$ is treated as a known quantity while $\partial_2 S_{12}(\cdot, \partial_2\mathbf{y})$ is treated as a nonlinear second derivative operator applied along the x_2 axis. The result is

$$\partial_2 S_{12}(\mathbf{x}_{i,j}) = \sum_{k=-w}^{w} D_{0k} S_{12}\left(\partial_1\mathbf{y}_{i,j+k}, \sum_{l=-w}^{w} D_{kl}\mathbf{y}_{i,j+l} \right) + \mathcal{O}(\Delta x^{2w}). \tag{50}$$

The terms $\partial_1 S_{21}$ and $\partial_2 S_{22}$ are obtained from Eqs. (49) and (50) by changing the first index on the stress function, and the residual vector at each point is taken to be $\mathbf{r}^f_{i,j} = \bar{\mathbf{f}}_{i,j} - (\nabla \cdot \mathbf{S})_{i,j}$. Including the precomputation of $\partial_1\mathbf{y}$ and $\partial_2\mathbf{y}$, an order n discretization of the residual depends on an $n+1$ by $n+1$ block of displacements centered on point $\mathbf{x}_{ij}$, and requires $n+1$ evaluations of each component of the stress tensor.

Note that while the analysis so far has treated materials with a fixed constitutive relation, the extension to spatially-dependent material parameters is straightforward. In this case the constitutive relation is defined by a strain energy density function $W(\mathbf{F}, \boldsymbol{\theta})$, where $\boldsymbol{\theta}(\mathbf{x})$ is a vector of spatially variable material properties defined at the grid points and the control points. For the interior discretization the values $\boldsymbol{\theta}_{i,j}$ are incorporated into computations of the stress and elasticity tensors as needed. For the boundary treatment proposed in the next section, ghost values of $\boldsymbol{\theta}$ can be calculated with the usual boundary stencil operations.

5.3. Immersed interface boundary treatment

The boundary treatment for the residual $\mathbf{r}^f$ follows the boundary treatment for cross derivatives defined in Section 4.2. At each control point, the displacement field $\mathbf{y}$ is interpolated using points from the set $\bar{\mathcal{X}}_c$. The derivatives $\partial_1\mathbf{y}$ and $\partial_2\mathbf{y}$ are precomputed at the affected points using centered difference stencils, evaluating the interpolant at each control point as needed to provide ghost values of $\mathbf{y}$. To evaluate Eqs. (49) and (50) at the affected points, ghost values of $\partial_1\mathbf{y}$ and $\partial_2\mathbf{y}$ are obtained as needed by differentiating the same set of interpolants.

To reduce implementation complexity, Eqs. (49) and (50) are most easily viewed as a sum over $2w+1$ computation stages, each comprised of two stencil operations and a nonlinear function evaluation. For illustration, consider the computation of $\partial_1 S_{11}$. During stage k, the 1D difference stencil $\{D_{kl}\}_{l=-w}^{w}$ is applied to $\mathbf{y}$ along the x_1 axis. The field $\partial_2\mathbf{y}$ is then shifted by k points along the same axis, which can be accomplished by applying the stencil with coefficients $\{\delta_{kl}\}_{l=-w}^{w}$. Finally, the function $S_{11}(\cdot,\cdot)$ is applied pointwise to the results of both stencil operations, and the output is weighted by the constant D_{0k}. This is repeated for $-w \le k \le w$, and the results of all stages are summed to form the field $(\partial_1 S_{11})_{i,j}$. For efficiency, the $k=0$ stage can be omitted by noting that $D_{00} = 0$ for any stencil width w. At each stage both stencil operations are subject to the boundary treatment outlined above, which provides ghost points for the fields $\mathbf{y}$ and $\partial_2\mathbf{y}$.

For control points on Γ_D, displacement boundary conditions are enforced by incorporating the known displacement $\bar{\mathbf{y}}(\mathbf{x}_c)$ into each polynomial interpolant. To discretize traction boundary conditions, let $\{s_c^j, s_\alpha^j\}$ be a set of vector stencil coefficients defined over the points in $\bar{\mathcal{X}}_c$ which approximate the gradient operator ∂_j on the boundary, so that

$$F_{ij}(\mathbf{x}_c) = \partial_j u_i(\mathbf{x}_c) = s_c^j u_i(\mathbf{x}_c) + \sum_{\alpha=1}^{n} s_\alpha^j u_i(\mathbf{x}_\alpha) + \mathcal{O}(\Delta x^k). \tag{51}$$

The residual $\mathbf{r}_c^t$ for each control point is computed from the discretized deformation gradient,

$$\mathbf{r}_c = \bar{\mathbf{t}}_c - \mathbf{S}(\mathbf{F}_c)\mathbf{n}_c. \tag{52}$$

Similarly, the residual vector for each control point on the material interface is defined by

$$\mathbf{r}_c^{[t]} = \left[\bar{\mathbf{t}}_c\right] - \left[\mathbf{S}^+(\mathbf{F}_c^+) - \mathbf{S}^-(\mathbf{F}_c^-)\right]\mathbf{n}_c, \tag{53}$$

with the displacement gradients $\mathbf{F}^+(\mathbf{x}_c)$ and $\mathbf{F}^-(\mathbf{x}_c)$ defined as in Eq. (51) for each side of the interface. In contrast to the linear elastic case, we do not solve the equations $\mathbf{r}_c^t = 0$ or $\mathbf{r}_c^{[t]} = 0$ individually at each control point. Instead, each control point on Γ_N or Γ_M adds two unknown displacement components $\mathbf{y}_c$ and two nonlinear equations (one for each residual component) to the global nonlinear system, which is solved simultaneously for the displacement in the domain and on the boundaries.

5.4. Nonlinear solution procedure

The spatial discretization provides a system of nonlinear equations which must be solved for the unknown displacements. As is typically the case for nonlinear elastic problems, a robust way to obtain solutions is to begin in the reference configuration with

homogeneous boundary conditions, then incrementally approach the final solution by a series of load steps. During each load step the applied displacements, loads, and body forces are increased by a small increment, and Newton's method is used to solve for the equilibrium configuration. This strategy takes advantage of the fact that Newton's method converges quickly when the initial guess is close to the final solution, and avoids the slow and unpredictable convergence behavior that results from a poor initial guess. For the examples shown below load stepping is only required to increment a prescribed displacement, and the corresponding algorithm is described below.

For the load stepping procedure, the displacements are partitioned into three groups $(y_\Omega, y_\Gamma^D, y_\Gamma^N)$, which include the displacements at grid points, at control points with displacement boundary conditions, and at control points with traction boundary conditions, respectively. The displacements y_Γ^D are prescribed and serve as known parameters, while y_Ω and y_Γ^N are the system unknowns. The residual vector can be partitioned into vectors (r_Ω, r_Γ^N), defined analogously, so that the full nonlinear system takes the form

$$\begin{aligned} r_\Omega(y_\Omega, y_\Gamma^N; y_\Gamma^D) &= 0, \\ r_\Gamma^N(y_\Omega, y_\Gamma^N) &= 0. \end{aligned} \tag{54}$$

Let $y_\Gamma^{D,k-1}$ be the prescribed boundary condition at step $k-1$, and $(y_\Omega^{k-1}, y_\Gamma^{N,k-1})$ be the corresponding solution. At step k, the displacements are incremented by $\delta y_\Gamma^D = y_\Gamma^{D,k} - y_\Gamma^{D,k-1}$. For all results below the load path is linear, so that δy_Γ^D is identical at each step. The initial guesses for the unknowns at step k can be written as $y_\Omega^k = \delta y_\Omega + y_\Omega^{k-1}$ and $y_\Gamma^{N,k} = \delta y_\Gamma^N + y_\Gamma^{N,k-1}$, where the increments $(\delta y_\Omega, \delta y_\Gamma^N)$ must be determined to satisfy Eq. (54) as closely as possible. Linearizing the system about the previous solution leads to a linear system for the increments,

$$\begin{aligned} \left(\frac{\partial r_\Omega}{\partial y_\Omega}\right)\delta y_\Omega + \left(\frac{\partial r_\Omega}{\partial y_\Gamma^N}\right)\delta y_\Gamma^N &= -\left(\frac{\partial r_\Omega}{\partial y_\Gamma^D}\right)\delta y_\Gamma^D, \\ \left(\frac{\partial r_\Gamma^N}{\partial y_\Omega}\right)\delta y_\Omega + \left(\frac{\partial r_\Gamma^N}{\partial y_\Gamma^N}\right)\delta y_\Gamma^N &= 0, \end{aligned} \tag{55}$$

where the Jacobian matrices are evaluated at $(y_\Omega^{k-1}, y_\Gamma^{N,k-1})$. After solving Eq. (55) to calculate the resulting initial guess, the solution at step k is obtained with Newton's method.

Both the load stepping procedure and Newton's method require the derivative of the residual vector with respect to each unknown. Returning to Eq. (49), an infinitesimal perturbation $\delta\mathbf{y}_{i,j}$ to the discrete displacement field results in a perturbation to the stress divergence $\partial_1 S_{(\cdot)1}$ given by

$$\delta(\partial_1 S_{(\cdot)1})_{i,j} = \sum_{k=-w}^{w} D_{0k} \frac{\partial \boldsymbol{S}_{(\cdot)1}}{\partial \mathbf{F}} \left(\widetilde{\partial_1 \mathbf{y}}_{i+k,j}, \partial_2 \mathbf{y}_{i+k,j}\right) : \left[\widetilde{\partial_1 \delta \mathbf{y}}_{i+k,j}, \partial_2 \boldsymbol{\delta} \mathbf{y}_{i+k,j}\right], \tag{56}$$

where : denotes tensor contraction and $[\mathbf{a}, \mathbf{b}]$ denotes the horizontal concatenation of two vectors to form a two by two tensor. The expression for $\delta(\partial_2 S_{(\cdot)2})_{i,j}$ is obtained by swapping axes. Eq. (56) is linear in the perturbation $\delta\mathbf{y}$, and can be used to assemble the Jacobian matrices required by the load stepping procedure. In practice, the derivatives $(\widetilde{\partial_1 \mathbf{y}}_{i+k,j}, \partial_2 \mathbf{y}_{i+k,j})$ are evaluated only once at each grid point, and used to evaluate both the residual and the components of the elasticity tensor that enter the Jacobian matrices. The derivatives of the boundary residual r_Γ^N are comparatively straightforward: for each control point on Γ^N, differentiating Eq. (52) yields

$$\frac{\partial r_i^t}{\partial y_k^c} = A_{ijkl}(\mathbf{F}_c) n_j s_l^c, \quad \frac{\partial r_i^t}{\partial y_k^\alpha} = A_{ijkl}(\mathbf{F}_c) n_j s_l^\alpha. \tag{57}$$

These derivatives are assembled to form the matrices $\frac{\partial r_\Gamma^N}{\partial y_\Gamma^N}$ and $\frac{\partial r_\Gamma^N}{\partial y_\Omega}$, respectively.

5.5. Verification for nonlinear elasticity

As a test case for the nonlinear discretization, an elastic body is subjected to a prescribed deformation of the form

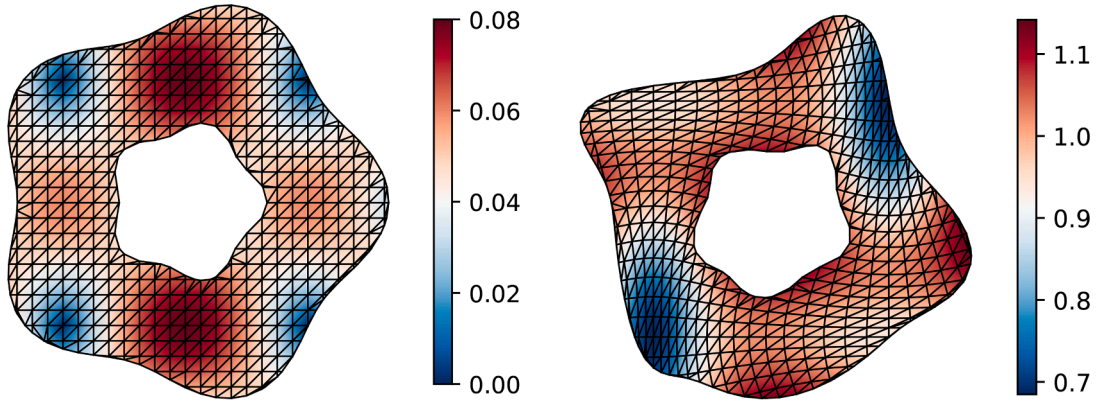
$$u_1(\mathbf{x}) = A_1 \cos(2\pi x_1)\sin(2\pi x_2), \quad u_2(\mathbf{x}) = -A_2 \sin(2\pi x_1)\cos(2\pi x_2), \tag{58}$$

with $A_1 = 0.08$ and $A_2 = 0.06$. The geometry and solution domain are identical to the linear elastic test case shown in Fig. 4. A traction boundary condition is prescribed on the inner boundary, while a displacement boundary condition is prescribed on the outer boundary. The constitutive relation is a compressible Neo-Hookean model with strain energy density

$$W(\mathbf{F}) = \frac{\mu}{2}\left(I_1 - 2 - 2\log J\right) + \frac{\lambda}{2}(J-1)^2, \tag{59}$$

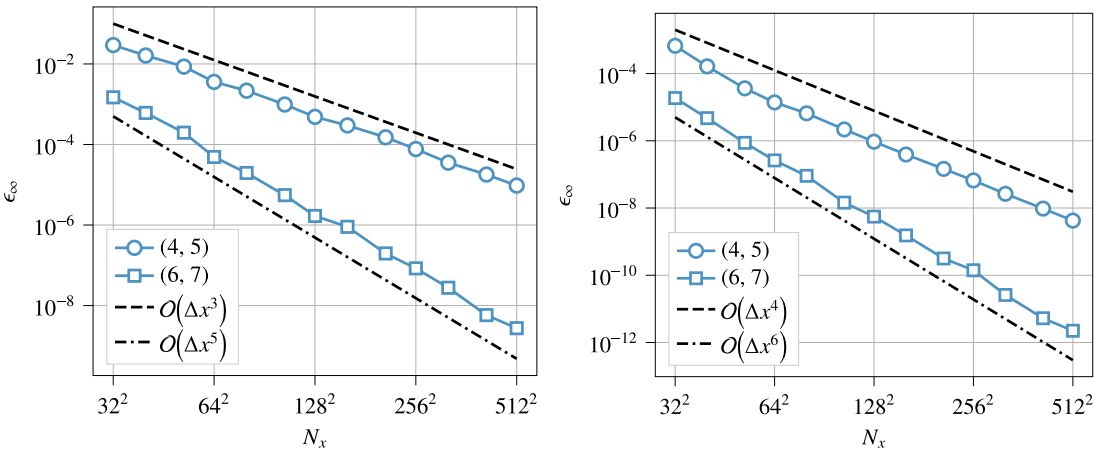
with Lamé parameters $\lambda = 4/3$ and $\mu = 1$. Here $I_1 = \mathrm{tr}(\mathbf{F}^T\mathbf{F})$ is the first invariant of the right Cauchy-Green deformation tensor and $J = \det \mathbf{F}$. The prescribed traction and body force are calculated so that Eq. (58) is an equilibrium configuration, leading to residual fields $\mathbf{r}^f(\mathbf{x}) = 0$ on Ω and $\mathbf{r}^t(\mathbf{s}) = 0$ on Γ_N. Fig. 8 illustrates both the deformed and undeformed configurations for this test case, as well as the magnitude of the deformation $\|\mathbf{u}(\mathbf{x})\|_2$ and the Jacobian determinant $J = \det \mathbf{F}$.

Fig. 9(a) plots the L_∞ norm of the residual vector that results when the discretization from the previous section is applied to the equilibrium configuration. This residual reflects the truncation error in the discretization of both the equilibrium equation and the



(a) Undeformed, colored by $\|\mathbf{u}\|_2$. (b) Deformed, colored by $J = \det \mathbf{F}$.

Fig. 8. Manufactured solution used to test the convergence of the nonlinear elastic discretization. While all results are obtained using a uniform Cartesian grid, a post-processed body-fitted triangle mesh is overlaid here to illustrate the deformation of the material.



(a) Truncation error. (b) Solution error.

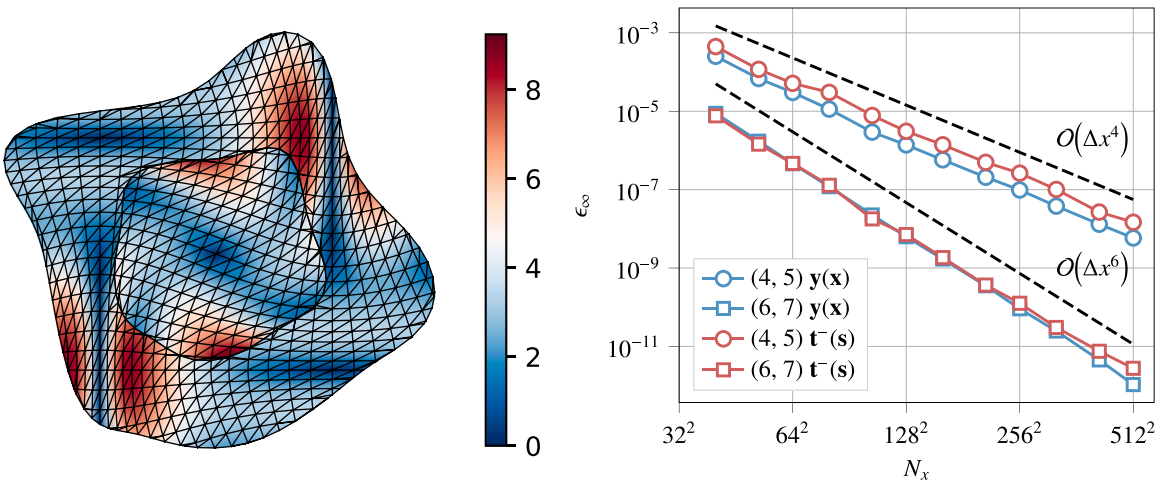
Fig. 9. Convergence of (a) the truncation error and (b) the solution error for a nonlinear elastic test case. Both the (4, 5) and (6, 7) discretizations achieve fourth and sixth order convergence in the L_∞ norm for the displacement field.

traction boundary condition. For $(4,5)$ and $(6,7)$ discretizations, the residual exhibits clean third or fifth order spatial convergence, matching the convergence rates observed for linear elasticity. To evaluate the solution error, the displacement field is initialized using the analytical solution from Eq. (58), and Newton iterations are performed until the update at each iteration satisfies $\|\delta \mathbf{y}\|_\infty < 10^{-12}$. The convergence of the L_∞ error in the displacement field is shown in Fig. 9(b), demonstrating that both the fourth and sixth order discretizations achieve their nominal order of accuracy.

To test the nonlinear discretization with material interfaces, the same deformation is applied to the geometry from the linear elastic test case illustrated in Fig. 6. The outer material is Neo-Hookean with parameters $\lambda^+ = 1.1$ and $\mu^+ = 0.6$, while the inner material has a Saint Venant-Kirchhoff strain energy density function

$$W = \frac{\lambda}{2}(\mathrm{tr}\mathbf{E})^2 + \mu \mathrm{tr}(\mathbf{E}^2), \quad \mathbf{E} = \frac{1}{2}\left(\mathbf{F}^T\mathbf{F} - \mathbf{I}\right) \tag{60}$$

with parameters $\mu^- = 0.8$ and $\lambda^- = 1.5$. Fig. 10(a) illustrates the body in the deformed configuration, colored by the magnitude of the body force in the deformed configuration $\bar{\mathbf{f}}/J$. While the deformation is continuous, both the traction and body force are discontinuous across the material interface. Fig. 10(b) plots the convergence of the displacement field and the first Piola-Kirchhoff traction on the inner surface of the material interface, after solving the nonlinear system to a tolerance of $\|\delta \mathbf{y}\|_\infty < 10^{-12}$. Both quantities converge at fourth or sixth order for the (4, 5) and (6, 7) discretizations.



(a) Deformed configuration. (b) Solution and traction errors.

Fig. 10. Nonlinear elasticity with material interfaces. (a) The deformed configuration of the elastic body, colored by the body force per deformed volume $\|\hat{\mathbf{f}}\|_2/J$. A post-processed body-fitted triangle mesh is overlayed here to illustrate the deformation of the material, while all computations are performed on a uniform Cartesian grid. (b) The L_∞ error in both the deformation field and the first Piola-Kirchhoff traction vector on the inner sided of the material interface. The (4, 5) and (6, 7) discretizations achieve fourth and sixth order convergence for both quantities.

6. Examples

In this section we show several examples of the ability of our code to solve challenging problems across single and multiple materials, and linear and nonlinear responses.

6.1. Linear periodic lattice

This linear elastic test case showcases the ability of high order immersed methods to capture complex geometry without mesh generation. The (4, 5) discretization is applied to a gyroid lattice structure with material interfaces, represented implicitly by the level sets

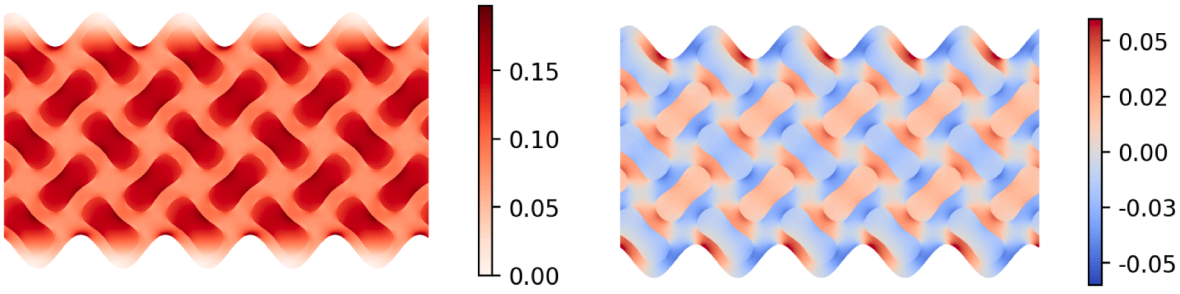
$$\begin{aligned} \psi_{\text{top}}(\mathbf{x}) &= x_2 - 0.14 - 0.04\cos(10\pi(x_1 + 0.02)), \\ \psi_{\text{bottom}}(\mathbf{x}) &= -x_2 + 0.66 + 0.04\cos(10\pi(x_1 - 0.02)), \\ \psi_{\text{lattice}}(\mathbf{x}) &= 1 - \left[\sin(10\pi x_1) + \cos(10\pi x_2) + \sqrt{2}\cos(10\pi x_1)\sin(10\pi x_2)\right]^2, \end{aligned} \tag{61}$$

which form the top surface, bottom surface, and lattice structure illustrated in Fig. 11. The lattice consists of two isotropic elastic materials which occupy the regions $\psi_{\text{lattice}}(\mathbf{x}) > 0$ and $\psi_{\text{lattice}}(\mathbf{x}) < 0$, with material properties defined by the Poisson ratios and Young's moduli $\nu^- = 0.3$, $E^- = 3$, $\nu^+ = 0.35$, $E^+ = 1$. Displacement boundary conditions are prescribed on both the top and bottom surfaces, with the top surface displaced by a distance $\delta u_1 = 0.05$ along the x_1 axis and the bottom surface fixed in place. On material interfaces there are no jumps in the displacement or traction, so that $[\![\tilde{\mathbf{u}}]\!] = 0$ and $[\![\tilde{\mathbf{t}}]\!] = 0$. The unit square solution domain is discretized with $N_x = 176$ grid points along each axis, and the resulting linear system is inverted with a sparse direct solver.

For post-processing purposes, the displacement gradient $\partial_j u_i$ can be calculated at each grid point using a centered difference stencil with an immersed interface boundary treatment. At the control points, the displacement gradient is calculated via boundary stencil operations. This allows the stress tensor $\sigma_{ij} = C_{ijkl}\partial_l u_k$ to be computed at each grid point and each control point. Fig. 11 plots the stress distribution inside the lattice structure, in particular the Von Mises stress $\sigma_{VM} = \sqrt{\sigma_{11}^2 + \sigma_{22}^2 - \sigma_{11}\sigma_{22} + 3\sigma_{12}^2}$ and a hydrostatic stress measure $\sigma_H = (\sigma_{11} + \sigma_{22})/2$. While there is no jump in the traction $t_i = \sigma_{ij}n_j$ across the material interfaces, discontinuities in other stress measures are present and sharply resolved by the high order immersed interface discretization.

6.2. Nonlinear periodic lattice

In this section, the fourth order nonlinear discretization is applied to the lattice structure introduced in previous section. The geometry and spatial resolution are kept identical, and the inner material is removed to yield a single material problem. The hyperelastic material is Neo-Hookean with parameters that correspond to a linear elastic response with $E = 1$ and $\nu = 0.35$, and the upper surface of the lattice is subject to a deformation $\delta\mathbf{y} = [0.15, 0.45]$ applied over five load steps. The deformed and undeformed configuration are shown in Fig. 12, colored by the Jacobian determinant $J = \det\mathbf{F}$, which illustrates the extreme stretching of the material.



(a) Von Mises stress σ_{VM}/E^+.

(b) Hydrostatic stress σ_H/E^+.

Fig. 11. Stress distribution inside of a multi-material gyroid lattice. The structure has a fixed lower surface and upper surface which displaced horizontally. The (a) von Mises stress (b) hydrostatic stress are both discontinuous across material interfaces, and both are sharply resolved by the fourth order immersed interface discretization.

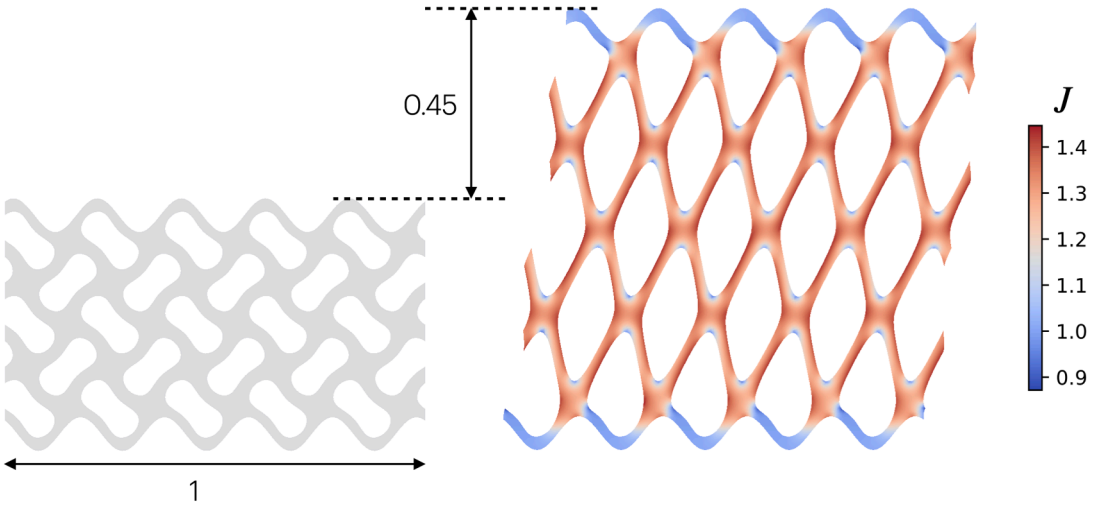


Fig. 12. The undeformed (left) and deformed (right) configurations of a hyperelastic gyroid lattice, colored by the Jacobian determinant $J = \det \mathbf{F}$. The deformation is based on a displacement of the top boundary $\delta \mathbf{y} = [0.15, 0.45]$.

6.3. Honeycomb lattice

To demonstrate the flexibility and accuracy of our approach, we apply it to an architectured structure consisting of interconnected finite-width rods. The structure is defined as a list of N line segments $\mathcal{L} = \{\mathbf{p}_s^{(i)}, \mathbf{p}_e^{(i)}, w^{(i)} \mid 1 \le i \le N\}$, with $\mathbf{p}_{s,e}^{(i)}$ the start and end position and $w^{(i)}$ the width of the ith line segment in the list. Different line segments may share the same starting or ending position, though they may not intersect. In this example, for simplicity, we set $w^{(i)} = w$ as a constant throughout the lattice.

To create our discretization, we define a signed-distance function based on $\mathcal{L}$. Since our discretization requires a minimum radius of curvature of the interface (Eq. (3)), we apply a small rounding by replacing every rod-rod intersection with a circular arc of a specified radius of curvature. This procedure leads to a pre-processing step that is further detailed in Appendix A. After the signed-distance function is evaluated on the grid, the intersections between the interface and the grid lines are evaluated by a level-set based interpolation strategy [30,35, appendices].

We compare our results to those obtained by the commercial finite element solver Abaqus. The Abaqus results are obtained from a traditional triangular body-fitted mesh on the original rod-based geometry. The finite elements used are 6-node quadratic plain stress elements (CPS6), and a standard linear solve is performed. For simulations where the interior of the structure is filled with a second material, the embedded element approach is used.

We choose a geometry based on a honeycomb lattice with 29×29 cells, each with an edge length of 22.7 μm, and some edges randomly removed. The width of the rods is set to $w = 0.01$ mm. The lattice is made of an isotropic linearly elastic model with Poisson's ratio $\nu = 0.3$ and Young's modulus $E = 70$ MPa. Displacement boundary conditions are prescribed on the top and bottom of the lattice, with the top surface displaced horizontally by $\delta u_1 = 0.01$ mm and the bottom surface fixed in place.

6.3.1. Single material

For the single material case, all edges except those on the top and bottom of the material are traction free. The IIM results are run with a grid size of, effectively, 1.085 μm, using a fourth-order discretization described as above. The total number of active grid points

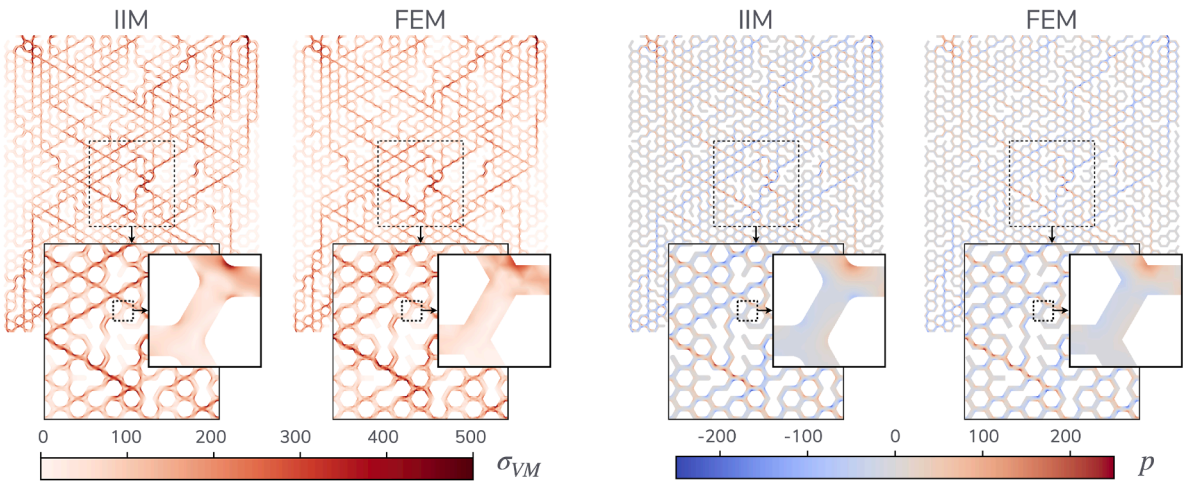


Fig. 13. The Von-Mises stress (left) and pressure (right) in the sheared honeycomb-like lattice structure, compared between the IIM results (left panel in each) and the FEM results (right panel in each). Insets provide successive close-ups, highlighted with dashed squares. Values are provided in units of kPa.

in the IIM grid, and hence the system size, is 416 204, whereas the FEM results are run with 37 645 elements for a total of 91 312 nodes. We note the geometry is especially unfavorable for the IIM approach, since it has to resolve the small radius of curvature at the corners. In this light, we will focus on an accuracy comparison, demonstrating the ability of IIM to produce accurate results even in this extreme case, without accounting for performance. Future work would be needed to investigate further the efficient handling of geometries with sharp corners in a more explicit way, and/or using adaptive grids to locally refine the grid near the corners.

Fig. 13 shows the Von Mises stress (left) and the pressure (right) compared between the IIM and the FEM results on the undeformed lattice. For the IIM results, a contour plot is created from a triangulated grid containing both the grid points and the control points similar to Figs. 8 and 10(a) above. The FEM results are visualized directly from Abaqus. The figures are rendered with the same colormap so that a one-to-one comparison is possible. The comparison shows that the IIM results match very well with the reference FEM results, both for the stress distribution within the individual rods, and the macroscopic stress field of the entire structure.

6.3.2. *Composite structure*

Extending the previous example, we now fill in all closed cavities in the honeycomb structure with a softer material of Young's modulus $E_2 = 10\,\text{MPa}$ and Poisson's ratio $\nu_2 = 0.3$. In the IIM, we apply jump boundary conditions on all material interfaces, and in Abaqus embedded elements are used. All exterior boundaries are traction free, and as above, displacement boundary conditions are applied at the top and bottom boundaries. The IIM results are run with an effective grid spacing of $0.72\,\mu\text{m}$, again using a fourth-order discretization. The total number of active grid points in the IIM grid, including both material regions, is 2 132 986. The FEM results are run using 541 364 elements for a total of 1 428 947 nodes.

Fig. 14 shows the same visual comparison as for a single material, demonstrating the ability of the IIM to recover micro- and macro-scale stress distributions also in multi-material composite structures. To perform a more quantitative comparison, we plot the stresses and vertical displacements evaluated across a line at $y = 0.5\,\text{mm}$ throughout the composite structure. For the IIM, this line corresponds to a gridline so that the stress jumps at the control points can be included in the plot wherever the grid line intersects a material boundary (or interface). The FEM results are obtained using the built-in 'X-Y data along a path' routine in Abaqus. Fig. 15 shows the stress and displacement distributions along this line, across the entire structure (left) and for the center $0.2\,\text{mm}$ (right). Again, the IIM results match well with the embedded element results, both inside the soft matrix and inside the stiffer lattice structure. Further, even more noticeable here is the embedded element method's treatment of material interfaces, leading to some oscillations in the visualized stresses near interface boundaries , though this could also be partially attributed to the interpolation routine. On the other hand, the IIM results remain smooth up to the material interfaces.

7. Discussion

The analysis and results above demonstrate that the proposed immersed method is able to achieve high fidelity results with high convergence orders for a variety of single and multi-material, linear and nonlinear elasticity problems.

As mentioned in the introduction, prevalent immersed methods in solid mechanics are typically based on immersed finite element methods. The main benefit of our approach is the simplicity of the required geometric information of the problem. Specifically, the proposed algorithm requires only local information: a list of intersection locations between the geometry and the Cartesian grid, with each intersection tagged by a position, surface normal vector, a type of boundary/interface condition, and an associated displacement, stress, or jump value. Notably, the algorithm itself does not require curvature information of the geometry, nor does it require any information about the topology of the intersections, as the corrections are handled locally for each finite difference stencil. Thus,

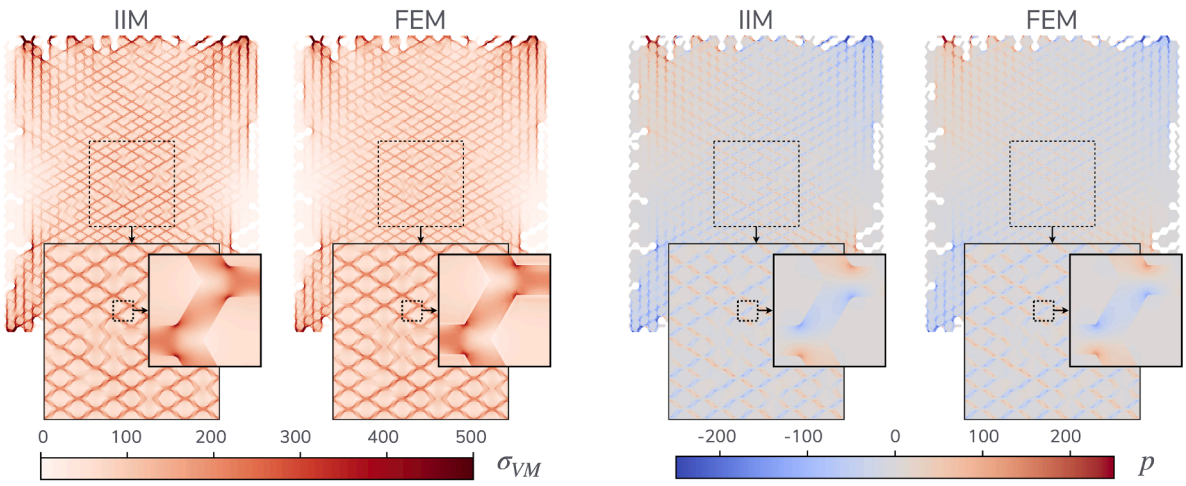


Fig. 14. The Von-Mises stress (left) and pressure (right) in the sheared honeycomb-like lattice structure with a soft matrix material, compared between the IIM results (left panel in each) and the FEM results (right panel in each). Insets provide successive close-ups, highlighted with dashed squares. Values are provided in units of kPa.

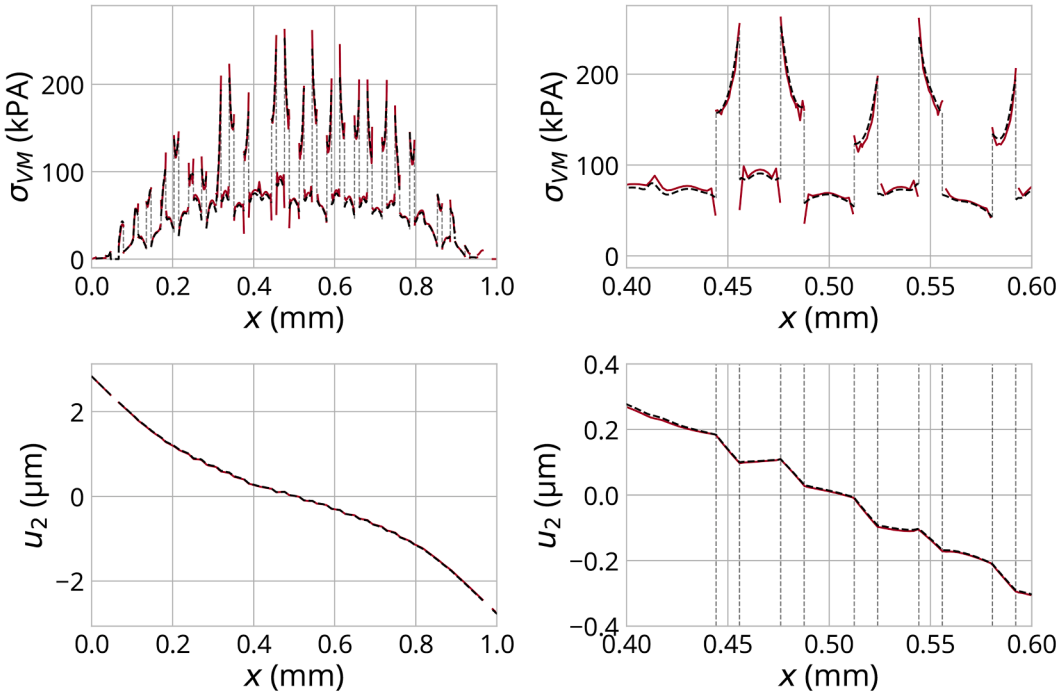


Fig. 15. Comparison of the Von Mises stress (top) and y-displacement (bottom) along a line crossing through the composite honeycomb structure at $y = 0.5$ mm. The IIM results are shown in black, including thin lines connecting the jump in stress values across material interfaces. The FEM results, obtained with an embedded element method, are shown in red. The right plots show zoom-ins of the left plots, with vertical dashed lines marking material transitions. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the approach avoids all numerical integration challenges common to immersed finite element methods, while retaining high order accuracy in the interior and boundary solutions.

Throughout this work, we use the algorithm proposed in [30] to extract the geometric intersection information from a level set function. Functional geometry descriptions like signed distance functions are increasingly used natively in various modeling packages, driven by organic shapes unlocked in additive manufacturing [41]. Further, signed distance function based geometries are extensively used in generative modeling, e.g. [42]. For geometries that are defined traditionally using e.g. spline or constructive solid geometry representations (CSG), one could develop specialized geometric tools to find intersections directly. More readily, algorithms to compute signed distance functions or level-set fields from 2D or 3D explicit surface representations are prevalent in the computer

graphics community, e.g. based on the software *EBGeometry* [43,44], after which the required intersections can be found directly from the level-set field.

As with most other immersed approaches, our algorithm requires that geometries with high curvatures or thin features are discretized with sufficient resolution. In our case, the resolution requirement Eq. (3) ensures that the regions sketched in Fig. 1 contain enough points to form a polynomial of the desired order. In addition, our algorithm detects whether the least-squares system based on $\mathcal{X}_c^{\pm}$ is ill-posed, thus identifying the need to refine in thin regions. Alternatively, one could also reduce the order of accuracy of the polynomial $p_c(\mathbf{x})$ locally to the available points in the regions $\mathcal{X}_c^{\pm}$.

At a corner, i.e. in the limit of infinite curvature, the resolution requirement Eq. (3) is always violated, and moreover a unique normal vector can not be defined. Corner features can thus not directly be discretized with the same order of accuracy as smooth geometries in our method, and in practice we therefore explicitly disallow such geometries. Similar concerns about non-smooth geometric features are common across immersed methods, e.g. as imposed by [13] for high order integration of implicit surfaces, recognized in [45] for level-set reinitialization, and discussed in [46] for cut-cell finite volume methods. To treat robustly corners in practice, as noted in [46], a mollification strategy is an effective approach. In Section 6.3 we use an *ad-hoc* specialized algorithm for lattice-type structures of constant thickness by introducing a single length scale for replacing the corners with circular arcs. More generally, smooth minimum approaches can be used on signed distance functions, which are common in geometric analyses and cut-cell methods. For instance, [46] introduces smooth minimum function that restores the high order convergence rate on their Poisson's equation solution. Similarly, [47] and more recently [48] provide strategies to systematically obtain differentiable smooth level-set or distance-based geometries in two and three dimensions.

Finally, we note that even with the sharp corner constraint in place the resulting subset of possible geometries contains many cases of practical interest: these include shell-based metamaterials [49,50], biological shapes [51], and organically designed shapes for additive manufacturing [52]

8. Conclusion

This work poses several contributions to the field of solving elasticity problems using immersed finite difference/volume discretization methods. First, we show high order accuracy of the solution in the domain, as well as solution quantities on the immersed interface. Second, we demonstrate robust treatment of boundary conditions (prescribed displacement or traction) and interface conditions (jump conditions in displacement and traction) on complex geometries. Third, our method can handle non-homogeneous isotropic and anisotropic material properties in each solution domain. Lastly, the method extends to nonlinear large strain problems. This makes the proposed approach a robust platform for use in, for instance, level-set based pipelines for topology optimization or data-driven machine learning models.

This work also creates a promising starting point for further developments. The extension to three-dimensional static elastic problems is largely an implementation problem, as all important discretizations extend naturally to 3D. In fact, we have already shown 3D results of high order immersed discretizations of scalar elliptic problems in [32], and similarly for hyperbolic and parabolic PDEs in [30,35]; these are all based on the same approach as presented here. Moreover, the approach integrates well with high order grid refinement techniques, such as our wavelet-based grid adaptation method developed in [53] and used in [32].

The approach also provides a viable starting ground for fluid-structure interaction problems. For this, an Eulerian method such as the reference map technique [54,55] can be used. Our immersed discretizations could offer a sharp, high order interface approach for handling continuity of displacement and stress across the fluid-solid boundary. The treatment of moving boundary problems while maintaining high order was discussed in [31].

Open challenges in our method pertain to the explicit treatment of sharp corners (in 2D and 3D) and edges (in 3D). The immersed interface method as discussed here relies both on a finite curvature and a well-defined normal vector to exist along the embedded boundary. In principle, different interpolation schemes could be used to alleviate the first constraint, and local regularization methods can be developed for cases where control points do not have well-defined normal vectors. These approaches are left for future work.

CRediT authorship contribution statement

James Gabbard: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Conceptualization; **Wim M. van Rees:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Methodology, Funding acquisition, Formal analysis, Conceptualization.

Data availability

Data will be made available on request.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Wim M. van Rees reports financial support was provided by US Department of Energy. James Gabbard reports financial support was provided by US Department of Energy. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We thank Andrew Chen and Dr. Carlos Portela for setting up, running, and postprocessing the finite element simulations used in the comparison for the honeycomb lattice structure of section 6.3. We wish to acknowledge financial support from an Early Career Award from the Department of Energy, Program Manager Dr. Steven Lee, award number DE-SC0020998.

Appendix A. Appendix

To create a honeycomb lattice signed distance function with rounded corners, we perform a geometric pre-processing approach.

We start from a rod-based structure defined as a list of N centerline segments $\mathcal{L} = \{\mathbf{p}_s^{(i)}, \mathbf{p}_e^{(i)}, w^{(i)} \mid 1 \le i \le N\}$, with $\mathbf{p}_{s,e}^{(i)}$ the start and end position and $w^{(i)}$ the width of the ith centerline segment in the list. As an example, Fig. A.16 shows a simple constant-thickness three-rod lattice whose centerline segments are drawn using black dashed lines. Our pre-processing approach starts by extruding all line segments in their normal direction, creating a 'top' and 'bottom' line segment for each rod as shown in Fig. A.16a. In the next step, we identify all interior corners and open edges, and determine which of the extruded line segments need to be connected. Finally, we use the tangent vectors of the extruded line segments to compute the center and opening angle of the arc connecting the tangent vectors. The curvature of the arc is provided as an input argument, so that the solution is unique. Lastly, we shorten the original extruded line segments to ensure they smoothly connect to the arcs. This leads to Fig. A.16 (right), where the shortened line segments are highlighted in red, and the arcs in blue.

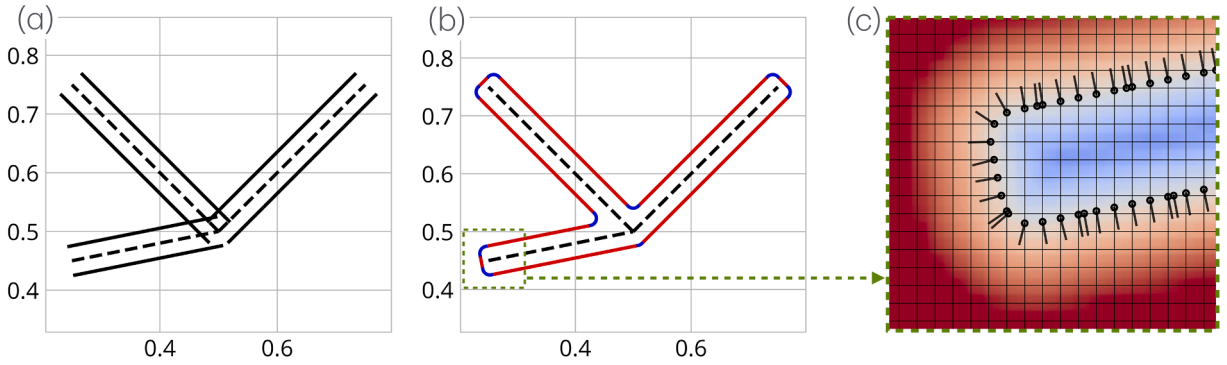


Fig. A.16. Geometric processing approach for rod-based lattice, shown for a three-rod structure with centerline segments drawn using black dashed lines. (a) naive extrusion of the centerlines to create a thickened geometry. (b) Final structure after connecting edge segments (in red) by rounding arcs (in blue). Here the thickness $w = 0.05$ and the radius of curvature $R = 0.0125$, using the same units as the plot. (c) After constructing the signed distance function (colors) the intersections (black circles) and normal vectors (lines) are constructed, here shown on a grid with uniform spacing $h = 1/128$ and zoomed-in to the bottom-left part of the geometry. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

After this processing step, the thickened geometry is represented in a list $\mathcal{R} = \{\mathbf{q}_s^{(i)}, \mathbf{q}_e^{(i)} \parallel 1 \le i \le N_q \; ; \mathcal{A}^{(j)} \mid 1 \le j \le N_a\}$. Here $\mathbf{q}_{s,e}^{(i)}$ denotes the start/end position of the N_q straight line segments (the red lines in Fig. A.16b). Further, $\mathcal{A}^{(j)}$ defines the jth arc connecting the straight line segments at each intersection (the blue arcs in Fig. A.16b). Each arc is defined by its center, radius of curvature, and starting/ending angle.

We compute the signed distance function at any point on our Cartesian grid through a search for the closest element in $\mathcal{R}$, Fig. A.16c. This search is accelerated using a cell-list approach. Finally, once we have the signed distance function evaluated on the grid, we compute the location of the intersections and the normal vectors through high order interpolation, as detailed in [30]. The resulting intersections and normal vectors for part of the geometry discussed here are shown in Fig. A.16c as the black circles and black lines, respectively.

References

[1] A. Duster, J. Parvizian, Z. Yang, E. Rank, The finite cell method for three-dimensional problems of solid mechanics, Comput. Methods Appl. Mech. Eng. 197 (2008) 3768–3782. https://doi.org/10.1016/j.cma.2008.02.036
[2] D. Schillinger, M. Ruess, The finite cell method: a review in the context of higher-order structural analysis of CAD and image-based geometric models, Arch. Comput. Methods Eng. 22 (2014) 391–455. https://doi.org/10.1007/s11831-014-9115-y
[3] P. Hansbo, M.G. Larson, S. Zahedi, A cut finite element method for a stokes interface problem, Appl. Numer. Math. 85 (2014) 90–114. https://doi.org/10.1016/j.apnum.2014.06.009
[4] E. Burman, S. Claus, P. Hansbo, M.G. Larson, A. Massing, CutFEM: discretizing geometry and partial differential equations, Int. J. Numer. Methods Eng. 104 (2014) 472–501. https://doi.org/10.1002/nme.4823
[5] I. Babuska, J.M. Melenk, The partition of unity method, Int. J. Numer. Methods Eng. 40 (1997) 727–758.
[6] N. Moës, J. Dolbow, T. Belytschko, A finite element method for crack growth without remeshing, Int. J. Numer. Methods Eng. 46 (1999) 131–150.
[7] K.W. Cheng, T. Fries, Higher-order XFEM for curved strong and weak discontinuities, Int. J. Numer. Methods Eng. 82 (2009) 564–590. https://doi.org/10.1002/nme.2768

[8] T. Fries, T. Belytschko, The extended/generalized finite element method: an overview of the method and its applications, Int. J. Numer. Methods Eng. 84 (2010) 253–304. https://doi.org/10.1002/nme.2914
[9] F. de Prenter, C.V. Verhoosel, E.H. van Brummelen, M.G. Larson, S. Badia, Stability and conditioning of immersed finite element methods: analysis and remedies, Arch. Comput. Methods Eng. 30 (2023) 3617–3656. https://doi.org/10.1007/s11831-023-09913-0
[10] A. Abedian, J. Parvizian, A. Duster, H. Khademyzadeh, E. Rank, Performance of different integration schemes in facing discontinuities in the finite cell method, Int. J. Comput. Methods 10 (2013) 1350002. https://doi.org/10.1142/s0219876213500023
[11] L. Kudela, N. Zander, S. Kollmannsberger, E. Rank, Smart octrees: accurately integrating discontinuous functions in 3D, Comput. Methods Appl. Mech. Eng. 306 (2016) 406–426. https://doi.org/10.1016/j.cma.2016.04.006
[12] S.C. Divi, C.V. Verhoosel, F. Auricchio, A. Reali, E.H. van Brummelen, Error-estimate-based adaptive integration for immersed isogeometric analysis, Comput. Math. Appl. 80 (2020) 2481–2516. https://doi.org/10.1016/j.camwa.2020.03.026
[13] T. Fries, S. Omerovic, Higher-order accurate integration of implicit geometries, Int. J. Numer. Methods Eng. 106 (2015) 323–371. https://doi.org/10.1002/nme.5121
[14] T. Fries, S. Omerovic, D. Schöllhammer, J. Steidl, Higher-order meshing of implicit geometries-Part I: integration and interpolation in cut elements, Comput. Methods Appl. Mech. Eng. 313 (2017) 759–784. https://doi.org/10.1016/j.cma.2016.10.019
[15] R.I. Saye, High-order quadrature methods for implicitly defined surfaces and volumes in hyperrectangles, SIAM J. Sci. Comput. 37 (2015) A993–A1019. https://doi.org/10.1137/140966290
[16] R.I. Saye, High-order quadrature on multi-component domains implicitly defined by multivariate polynomials, J. Comput. Phys. 448 (2022) 110720. https://doi.org/10.1016/j.jcp.2021.110720
[17] J.E. Fromm, N. Wunsch, R. Xiang, H. Zhao, K. Maute, J.A. Evans, D. Kamensky, Interpolation-based immersed finite element and isogeometric analysis, Comput. Methods Appl. Mech. Eng. 405 (2023) 115890. https://doi.org/10.1016/j.cma.2023.115890
[18] M. Berger, Cut Cells: Meshes and Solvers, Elsevier, 2017. https://doi.org/10.1016/bs.hna.2016.10.008
[19] E. Burman, Ghost penalty, C.R. Math. 348 (2010) 1217–1220. https://doi.org/10.1016/j.crma.2010.10.006
[20] S. Fernández-Méndez, A. Huerta, Imposing essential boundary conditions in mesh-free methods, Comput. Methods Appl. Mech. Eng. 193 (2004) 1257–1275. https://doi.org/10.1016/j.cma.2003.12.019
[21] F. de Prenter, C. Lehrenfeld, A. Massing, A note on the stability parameter in Nitsche's method for unfitted boundary value problems, Comput. Math. Appl. 75 (2018) 4322–4336. https://doi.org/10.1016/j.camwa.2018.03.032
[22] E. Burman, A penalty-free nonsymmetric Nitsche-type method for the weak imposition of boundary conditions, SIAM J. Numer. Anal. 50 (2012) 1959–1981. https://doi.org/10.1137/10081784x
[23] D. Schillinger, I. Harari, M.-C. Hsu, D. Kamensky, S.K. Stoter, Y. Yu, Y. Zhao, The non-symmetric nitsche method for the parameter-free imposition of weak boundary and coupling conditions in immersed finite elements, Comput. Methods Appl. Mech. Eng. 309 (2016) 625–652. https://doi.org/10.1016/j.cma.2016.06.026
[24] M. Theillard, L.F. Djodom, J.-L. Vié, F. Gibou, A second-order sharp numerical method for solving the linear elasticity equations on irregular domains and adaptive grids - application to shape optimization, J. Comput. Phys. 233 (2013) 430–448. https://doi.org/10.1016/j.jcp.2012.09.002
[25] X. Yang, Immersed interface method for elasticity problems with interfaces, Ph.D. thesis, 2004. https://www.proquest.com/dissertations-theses/immersed-interface-method-elasticity-problems/docview/305165073/se-2
[26] X. Yang, B. Li, Z. Li, The immersed interface method for elasticity problems with interfaces, Dyn. Contin. Discrete Impulsive Syst., Series A – Math. Anal. 10 (2003) 783–808. Conference on Partial Differential Equations (PDE), Washington State University, Pullman, Washington, May 2002.
[27] B. Wang, K. Xia, G.W. Wei, Matched interface and boundary method for elasticity interface problems, J. Comput. Appl. Math. 285 (2015) 203–225. https://doi.org/10.1016/j.cam.2015.02.005
[28] B. Wang, K. Xia, G.W. Wei, Second order method for solving 3D elasticity equations with complex interfaces, J. Comput. Phys. 294 (2015) 405–438. https://doi.org/10.1016/j.jcp.2015.03.053
[29] Y. Xing, L. Song, C.M. Fan, A generalized finite difference method for solving elasticity interface problems, Eng. Anal. Bound. Elem. 128 (2021) 105–117. https://doi.org/10.1016/j.enganabound.2021.03.026
[30] J. Gabbard, W.M. van Rees, A high-order 3D immersed interface finite difference method for the advection-diffusion equation, in: AIAA SCITECH 2023 Forum, 2023, p. 2480. https://doi.org/10.2514/6.2023-2480
[31] J. Gabbard, W.M. van Rees, A high-order finite difference method for moving immersed domain boundaries and material interfaces, J. Comput. Phys. 507 (2024) 112979. https://doi.org/10.1016/j.jcp.2024.112979
[32] J. Gabbard, A. Paris, W.M. van Rees, A high order multigrid-preconditioned immersed interface solver for the Poisson equation with boundary and interface conditions, 2025, arXiv:2503.22455
[33] R.J. LeVeque, Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (1994) 1019–1044. https://doi.org/10.1137/0731054
[34] Z. Li, K. Ito, The immersed interface method: numerical solutions of PDEs involving interfaces and irregular domains, SIAM, 2006.
[35] J. Gabbard, T. Gillis, P. Chatelain, W.M. van Rees, An immersed interface method for the 2D vorticity-velocity Navier–Stokes equations with multiple bodies, J. Comput. Phys. 464 (2022) 111339. https://doi.org/10.1016/j.jcp.2022.111339
[36] W. Thacher, H. Johansen, D. Martin, A high order cartesian grid, finite volume method for elliptic interface problems, J. Comput. Phys. 491 (2023) 112351.
[37] W. Thacher, H. Johansen, D. Martin, A high order cut-cell method for solving the shallow-shelf equations, J. Comput. Sci. 80 (2024) 102319.
[38] C.-W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, II, J. Comput. Phys. 83 (1989) 32–78.
[39] B. Merriman, Understanding the shu–osher conservative finite difference form, J. Sci. Comput. 19 (2003) 309–322.
[40] H. Johansen, P. Colella, A cartesian grid embedded boundary method for Poisson's equation on irregular domains, J. Comput. Phys. 147 (1998) 60–85. https://doi.org/10.1006/jcph.1998.5965
[41] F. Veloso, J. Gomes-Fonseca, P. Morais, J. Correia-Pinto, A.C. Pinho, J.L. Vilaca, Overview of methods and software for the design of functionally graded lattice structures, Adv. Eng. Mater. 24 (2022) 2200483. https://doi.org/10.1002/adem.202200483
[42] G. Chou, Y. Bahat, F. Heide, Diffusion-SDF: conditional generative modeling of signed distance functions, 2023,
[43] R. Marskar, EBGeometry repository, 2023, https://rmrsk.github.io/EBGeometry/
[44] R. Marskar, An adaptive cartesian embedded boundary approach for fluid simulations of two- and three-dimensional low temperature plasma filaments in complex geometries, J. Comput. Phys. 388 (2019) 624–654. https://doi.org/10.1016/j.jcp.2019.03.036
[45] R. Saye, High-order methods for computing distances to implicitly defined surfaces, Commun. Appl. Math. Comput. Sci. 9 (2014) 107–141. https://doi.org/10.2140/camcos.2014.9.107
[46] D. Devendran, D. Graves, H. Johansen, T. Ligocki, A fourth-order Cartesian grid embedded boundary method for Poisson's equation, Commun. Appl. Math. Comput. Sci. 12 (2017) 51–79. https://doi.org/10.2140/camcos.2017.12.51
[47] Q. Li, Smooth piecewise polynomial blending operations for implicit shapes, Comput. Graphics Forum 26 (2007) 157–171. https://doi.org/10.1111/j.1467-8659.2007.01011.x
[48] J.E. Hicken, S. Kaur, An explicit level-set formula to approximate geometries, 2022, https://doi.org/10.2514/6.2022-1862
[49] C. Bonatti, D. Mohr, Mechanical performance of additively-manufactured anisotropic and isotropic smooth shell-lattice materials: simulations & experiments, J. Mech. Phys. Solids 122 (2019) 1–26. https://doi.org/10.1016/j.jmps.2018.08.022
[50] S. Dhulipala, C.M. Portela, Curvature-guided mechanics and design of spinodal and shell-based architected materials, 2025, arXiv:2505.21509
[51] G. Domokos, A. Goriely, A.G. Horvath, K. Regos, Soft cells and the geometry of seashells, PNAS Nexus 3 (2024) pgae311. https://doi.org/10.1093/pnasnexus/pgae311

[52] O. Al-Ketan, D.-W. Lee, R.K. Abu Al-Rub, Mechanical properties of additively-manufactured sheet-based gyroidal stochastic cellular materials, Addit. Manuf. 48 (2021) 102418. https://doi.org/10.1016/j.addma.2021.102418
[53] T. Gillis, G. Winckelmans, P. Chatelain, Fast immersed interface Poisson solver for 3D unbounded problems around arbitrary geometries, J. Comput. Phys. 354 (2018) 403–416. https://doi.org/10.1016/j.jcp.2017.10.042
[54] K. Kamrin, J.C. Nave, An Eulerian approach to the simulation of deformable solids: application to finite-strain elasticity, 2009, https://doi.org/10.48550/ARXIV.0901.3799
[55] K. Kamrin, C.H. Rycroft, J.C. Nave, Reference map technique for finite-strain elasticity and fluid-solid interaction, J. Mech. Phys. Solids 60 (2012) 1952–1969. https://doi.org/10.1016/j.jmps.2012.06.003