eXtended Hybridizable Discontinuous Galerkin (X-HDG) for Void Problems

Ceren Gürkan¹ · Esther Sala-Lardies¹ ·
Martin Kronbichler² · Sonia Fernández-Méndez¹

Received: 22 January 2015 / Revised: 16 June 2015 / Accepted: 23 June 2015
© Springer Science+Business Media New York 2015

Abstract A strategy for the Hybridizable Discontinuous Galerkin (HDG) solution of problems with voids, inclusions or free surfaces is proposed. It is based on an eXtended Finite Element philosophy with a level-set description of interfaces. Thus, the computational mesh is not required to fit the interface (i.e. the boundary), simplifying and reducing the cost of mesh generation and, in particular, avoiding continuous remeshing for evolving interfaces. Differently to previous proposals for HDG solution with non-fitting meshes, here the computational mesh covers the domain, avoiding extrapolations, and ensuring the robustness of the method. The local problem at elements not cut by the interface, and the global problem, are discretized as usual in HDG. A modified local problem is considered at elements cut by the interface. At every cut element, an auxiliary trace variable on the boundary is introduced, which is eliminated afterwards using the boundary conditions on the interface, keeping the original unknowns and the structure of the local problem solver. An efficient and robust methodology for numerical integration in cut elements, in the context of high-order approximations, is also proposed. Numerical experiments demonstrate how X-HDG keeps the optimal convergence, superconvergence, and accuracy of HDG with no need of adapting the computational mesh to the interface boundary.

Keywords Interface · Moving boundary · Hybridizable Discontinuous Galerkin (HDG) · High-order · Level-sets · X-FEM · Numerical integration

✉ Sonia Fernández-Méndez
sonia.fernandez@upc.edu

Ceren Gürkan
ceren.gurkan@upc.edu

Esther Sala-Lardies
esther.sala-lardies@upc.edu

Martin Kronbichler
kronbichler@lnm.mw.tum.de

¹ Universitat Politècnica de Catalunya, Barcelona, Spain

² Technische Universität München, Munich, Germany

1 Introduction

Even though the Hybridizable Discontinuous Galerkin (HDG) method is a novel method proposed just a few years ago [4,5], it has nowadays been successfully applied to all kinds of problems, specially in the field of Computational Fluid Dynamics (CFD); see, for instance, [6,20,21] for its application to the Stokes and Navier–Stokes equations, or [14,17,18] for an efficiency study in front of Continuous Finite Elements (CFE) in the context of elliptic problems and wave problems. HDG inherits all the advantages of high-order Discontinuous (DG) methods [3,16,19,22] that have made them so popular in CFD in the last decade, such as local conservation of quantities of interest, intrinsic stabilization thanks to a proper definition of numerical fluxes at element boundaries, suitability for code vectorization and parallel computation, and suitability for adaptivity. But, HDG outperforms other DG methods for problems involving self-adjoint operators, due to two main peculiarities: hybridization and superconvergence properties. The hybridization process drastically reduces the number of degrees of freedom in the discrete problem, similarly to static condensation in the context of high-order CFE, see for instance [14]. For instance, in a Laplace equation the unknowns reduce to the approximation of the trace of the solution at the mesh skeleton, i.e. the sides (or faces in 3D) of the mesh; and in incompressible flow problems, the final unknowns correspond to just the trace of the velocity at the mesh skeleton plus one scalar representing the mean of the pressure at every element. On the other hand, HDG is based on a mixed formulation that, differently to CFE or other DG methods, is stable even when all variables (primal unknowns and derivatives) are approximated with polynomials of the same degree k . Consequently, convergence of order $k + 1$ in $\mathcal{L}_2$ norm is proved not only for the primal unknown, but also for its derivatives. Therefore, a simple element-by-element postprocess of the derivatives leads to a superconvergent approximation of the primal variables, with convergence of order $k + 2$ in $\mathcal{L}_2$ norm. The superconvergent solution can also be used to compute an efficient error estimator and define an adaptivity procedure [12,13].

However, despite the interest in the development and application of HDG during the last years, there is still work to be done for the efficient solution of problems with moving boundaries and interfaces. In [24], HDG is applied to the solution of Stokes interface problems, but always considering computational meshes fitting the interface. Therefore, the mesh should be adapted to properly describe the interface geometry, requiring continuous remeshing in the case of evolving interfaces. On other hand, a methodology for the solution of elliptic problems with meshes not fitting the boundary is proposed in [8,9]. The solution at the boundary is extrapolated from nodal values of the computational mesh; consequently, some restrictive requirements on the distance from the computational mesh to the boundary are necessary to achieve optimal convergence, limiting the practical applicability of the proposal.

An alternative strategy for the HDG solution of interface problems, based on an eXtended Finite Element (X-FEM) philosophy is proposed here. X-FEM has nowadays demonstrated its suitability for the solution of problems with moving boundaries and interfaces in the context of CFE (see, for instance, the overview paper [11] and the references therein), recently also with high-order approximations [1,10]. A level set is used for the geometrical representation of interfaces or interior boundaries, and the original computational mesh and unknown structure is maintained, with a proper numerical integration at elements and faces cut by the interface. In the case of material interfaces or cracks, a proper enrichment can be introduced to represent discontinuities inside cut elements.

Motivated by the demonstrated efficiency of HDG and X-FEM, this work aims to open a path for the efficient solution of problems with moving boundaries and interfaces described by

level sets. This paper develops an eXtended Hybridizable Discontinuous Galerkin (X-HDG) method for the solution of problems with voids, inclusions, or free surfaces. X-HDG inherits the advantages of X-FEM methods (the computational mesh is not required to fit the interface, simplifying and reducing the cost of mesh generation and, in particular, avoiding continuous remeshing for evolving interfaces or boundaries), while keeping the computational efficiency, stability, accuracy and optimal convergence of HDG.

Differently to [8,9], here the computational mesh always covers the domain and, therefore, no extrapolations are required, leading to a more robust method. In fact, in 2D, the X-HDG method proposed here is formally equivalent to a standard HDG method applied on a cut mesh combining triangles and quadrilateral elements with a $\mathcal{P}_k$ polynomial approximation (i.e. a complete k -th degree polynomial basis also for quadrilaterals), but organized and implemented in an alternative way to keep the original triangular computational mesh and the original unknowns structure, as usual in X-FEM methods. Thus, X-HDG keeps the superconvergence and stability properties of standard HDG, but in accordance with an X-FEM philosophy.

Section 2 presents the X-HDG formulation and discretization for void problems with Neumann boundary conditions on the interface (i.e. on the boundary not fitted by the computational mesh). The local problem at elements not cut by the interface, and the global problem, are discretized as usual in HDG. The discretization of the local problem for cut elements is developed in Sect. 2.2. At every cut element, an auxiliary trace variable on the boundary is introduced, which is eliminated afterwards using the boundary conditions on the interface, keeping the original unknowns. The X-HDG method is then extended to problems with Dirichlet interfaces in Sect. 3. The application to problems involving both Dirichlet and Neumann interfaces is straightforward. A robust and efficient methodology for numerical integration in cut elements is proposed in Sect. 4. Similarly to [1], a k -th degree parametrization for the approximation of the interface in each cut element is considered. However, here the parametrization may be piecewise polynomial, getting rid of the mesh requirements in [1], and being capable of handling more complicated situations that may appear in high-order computations, such as bubbles inside an element, or an element divided in more than two regions by the interface. Finally, numerical examples in Sect. 5 demonstrate the applicability of the method, and how X-HDG keeps the accuracy, optimal convergence, and superconvergence of HDG.

The X-HDG method keeps the convergence, accuracy and computational efficiency of HDG for high-order computations, while getting rid of the adaptation of computational mesh to the interface boundary. The application of the developed formulation to problems with evolving boundaries avoids the continuous remeshing to fit the interface, with the corresponding saving in computational time, and avoiding the loss of accuracy derived from projection of quantities from one mesh to another. The additional cost of X-HDG in front of HDG is mainly due to the modification of the numerical quadrature, since elemental computations for cut elements are substantially more expensive than for standard elements. However, it is worth noting that in practical applications the ratio of cut elements to standard elements is small, and the overcost is negligible compared to the cost of remeshing.

2 X-HDG Formulation for Problems with Neumann Voids

Let $\Omega \subset \mathbb{R}^d$ be a bounded domain with an interior boundary $\mathcal{I}$ (also referred as interface) and an exterior boundary $\partial\Omega^{ext} := \partial\Omega \setminus \mathcal{I}$. The following problem is considered,

Fig. 1 Two examples of domain with a void: a circular void boundary and a straight interface, $\mathcal{I}$ in black. The mesh covers the domain Ω (in grey) and fits the exterior boundary $\partial\Omega^{ext} = \partial\Omega \setminus \mathcal{I}$

$$\begin{aligned} -\nabla \cdot (v\nabla u) &= f \quad \text{in } \Omega \\ v\nabla u \cdot \mathbf{n} &= g \quad \text{on } \mathcal{I} \\ u &= u_D \quad \text{on } \partial\Omega^{ext} \end{aligned} \quad (1)$$

where u is the solution, v is a material coefficient, f is a given source term, u_D are prescribed values on the exterior boundary, and g is a prescribed flux on the interior boundary, i.e. the voids boundary. Neumann boundary conditions are considered in the interior boundary $\mathcal{I}$; the implementation of Dirichlet boundary conditions on $\mathcal{I}$ is developed in Sect. 3. Here, for simplicity, Dirichlet boundary conditions are considered on the exterior boundary $\partial\Omega^{ext}$; other boundary conditions at $\partial\Omega^{ext}$ do not add any difficulty, since they are implemented as in standard HDG.

The domain Ω is now assumed to be covered by a finite element mesh with n_{el} disjoint elements K_i , such that

$$\overline{\Omega} \subset \bigcup_{i=1}^{n_{el}} \overline{K_i}, \quad K_i \cap K_j = \emptyset \quad \text{for } i \neq j, \quad \partial\Omega^{ext} \subset \partial \left[\bigcup_{i=1}^{n_{el}} \overline{K_i} \right]$$

Note that the mesh fits the exterior boundary $\partial\Omega^{ext}$, but some elements may be cut by the interior boundary $\mathcal{I}$, see Fig. 1. The union of all n_{fc} faces Γ_i (sides for 2D) intersecting the domain Ω is denoted as

$$\Gamma := \bigcup_{i=1}^{n_{el}} [\partial K_i \cap \overline{\Omega}] = \bigcup_{f=1}^{n_{fc}} [\Gamma_f \cap \overline{\Omega}].$$

The discontinuous setting induces a new problem equivalent to (1), with some element-by-element equations and some global ones. The local element-by-element problems correspond to the statement of the PDE in (1) with essential boundary conditions at each element K_i , that is,

$$\left. \begin{aligned} \nabla \cdot \mathbf{q} &= f && \text{in } K_i \\ \mathbf{q} + v\nabla u &= 0 && \text{in } K_i \\ u &= \hat{u} && \text{on } \partial K_i \end{aligned} \right\} \text{ if } \mathcal{I} \cap K_i = \emptyset \quad (2a)$$

$$\left. \begin{aligned} \nabla \cdot \mathbf{q} &= f && \text{in } \Omega_i \\ \mathbf{q} + v\nabla u &= 0 && \text{in } \Omega_i \\ \mathbf{q} \cdot \mathbf{n} &= g && \text{on } \mathcal{I}_i \\ u &= \hat{u} && \text{on } \partial\Omega_i \setminus \mathcal{I}_i \end{aligned} \right\} \text{ if } \mathcal{I} \cap K_i \neq \emptyset \quad (2b)$$

for $i = 1, \dots, n_{el}$, where, for cut elements,

$$\Omega_i := \Omega \cap K_i, \quad \mathcal{I}_i := \mathcal{I} \cap K_i, \quad (3)$$

Fig. 2 Element cut by the interior boundary $\mathcal{I}$: the intersection of the element K_i with the interface and the domain (in grey) are denoted as $\mathcal{I}_i$ and Ω_i

see Fig. 2. Two new variables are introduced: $\mathbf{q}$ corresponding to the flux of u , allowing the splitting of the PDE in two first order PDEs, and $\hat{u}$ corresponding to the trace of u at the mesh faces. The local problems have been particularized for elements cut by the interior boundary (2b) and standard elements (2a).

Note that, given the trace $\hat{u}$, which is single valued at the mesh skeleton Γ , the local problems (2) can be solved at each element to determine the solution u and the flux $\mathbf{q}$. Thus, the problem now reduces to determine the trace $\hat{u}$, with the so-called *conservativity conditions* (also known as *global equations*), that is, the continuity of the flux across element boundaries

$$[[\mathbf{q} \cdot \mathbf{n}]] = 0 \quad \text{on } \Gamma \setminus \partial\Omega^{ext}, \quad (4)$$

and the boundary condition, equivalent to the exterior boundary condition in (1),

$$\hat{u} = u_D \quad \text{on } \partial\Omega^{ext}, \quad (5)$$

The *jump* $[[\cdot]]$ operator is defined at a face Γ_f as

$$[[\odot]] = \odot_{L(f)} + \odot_{R(f)} \quad \text{on } \Gamma_f,$$

where $R(f)$ and $L(f)$ are numbers of the left and right elements sharing the face, that is, $\Gamma_f = \bar{K}_{L(f)} \cap \bar{K}_{R(f)}$, and the subindex $\odot_i$ denotes the value of function $\odot$ from element K_i . In particular, $[[\mathbf{q} \cdot \mathbf{n}]] = \mathbf{q}_{L(f)} \cdot \mathbf{n}_{L(f)} + \mathbf{q}_{R(f)} \cdot \mathbf{n}_{R(f)} = (\mathbf{q}_{L(f)} - \mathbf{q}_{R(f)}) \cdot \mathbf{n}_{L(f)}$.

It is important noting that the continuity of the solution u across Γ is imposed by the Dirichlet boundary condition in the local problems (2) and the fact that $\hat{u}$ is single valued on Γ .

The discretization of the conservativity condition (4) and the local problems (2), with the boundary condition (5), leads to the X-HDG formulation. The following discrete spaces for elemental variables, u and $\mathbf{q}$, and for the trace variable, $\hat{u}$, are considered

$$\begin{aligned} \mathcal{V}^h &:= \{v \in \mathcal{L}^2(\Omega) : v|_{K_i \cap \Omega} \in \mathcal{P}_k(K_i \cap \Omega) \text{ for } i = 1, \dots, n_{e1}\} \\ \Lambda^h &:= \{\hat{v} \in \mathcal{L}^2(\Gamma) : \hat{v}|_{\Gamma_f \cap \bar{\Omega}} \in \mathcal{P}_k(\Gamma_f \cap \bar{\Omega}) \text{ for } f = 1, \dots, n_{fc}\}, \end{aligned} \quad (6)$$

where $\mathcal{P}_k$ denotes the space of polynomials of degree less or equal to k .

The next sections present the details of the X-HDG formulation, stating the discretization of the local problems for standard and cut elements and the discretization of the conservativity condition (4). The local problem at elements not cut by the interface (2a) and the global problem are discretized as usual in HDG [4,5], as recalled in Sects. 2.1 and 2.3. The discretization of the local problem for cut elements (2b) is developed in Sect. 2.2.

To simplify the presentation, in an abuse of notation, the same notation is used for the numerical approximation, belonging to the finite dimensional spaces (6), and the exact solution, that is u , $\mathbf{q}$ and $\hat{u}$.

2.1 Local Problem for Standard Elements

This section recalls the standard HDG local problem at an element K_i not cut by the interface. It corresponds to the discretization of (2a), that is: given $\hat{u} \in \Lambda^h$, find $u \in \mathcal{P}_k(K_i)$, $\mathbf{q} \in [\mathcal{P}_k(K_i)]^d$ such that

Fig. 3 Example of a third degree HDG discretization. Nodal approximation at elements (grey nodes) for u and $\mathbf{q}$, and nodal approximation at faces (black nodes) for the trace $\hat{u}$. The three faces for element K_i correspond to faces number F_{i1} , F_{i2} and F_{i3}

$$\begin{aligned} \int_{K_i} v \nabla \cdot \mathbf{q} \, dV + \int_{\partial K_i} \tau v (u - \hat{u}) \, dS &= \int_{K_i} v f \, dV \quad \forall v \in \mathcal{P}_k(K_i) \\ \int_{K_i} \mathbf{q} \cdot \mathbf{w} \, dV - \int_{K_i} v u \nabla \cdot \mathbf{w} \, dV + \int_{\partial K_i} v \hat{u} \mathbf{w} \cdot \mathbf{n} \, dS &= 0 \quad \forall \mathbf{w} \in [\mathcal{P}_k(K_i)]^d \end{aligned} \quad (7)$$

The first equation in (7) can be derived from the first equation in (2a) applying integration by parts, replacing the flux by the numerical flux

$$\hat{\mathbf{q}} := \mathbf{q} + \tau(u - \hat{u})\mathbf{n}, \quad (8)$$

and undoing the integration by parts. The second equation is obtained from the weak form of the second equation in (2a), applying integration by parts and replacing the boundary condition $u = \hat{u}$ on the element boundary.

Remark 1 The parameter τ , appearing in the definition of the numerical flux (8), is a non-negative stabilization parameter usually taken of order $\mathcal{O}(\nu)$. For each element, it may be taken as a positive constant on all faces, or positive on one arbitrary face and zero at the rest (single face). Both options lead to stable and optimally convergent solutions, with superconvergent post-processed solutions. See for instance [4, 12] for details on the influence of this parameter on the solution behavior.

The discretization of the local problem (7) leads to a system of equations of the form

$$\begin{cases} \mathbf{A}_{uu}^{K_i} \mathbf{u}^i + \mathbf{A}_{uq}^{K_i} \mathbf{q}^i + \mathbf{A}_{u\hat{u}}^{K_i} \Lambda^i = \mathbf{f}_u^{K_i} \\ \mathbf{A}_{qu}^{K_i} \mathbf{u}^i + \mathbf{A}_{qq}^{K_i} \mathbf{q}^i + \mathbf{A}_{q\hat{u}}^{K_i} \Lambda^i = \mathbf{0} \end{cases} \quad (9)$$

where $\mathbf{u}^i$ and $\mathbf{q}^i$ are the vectors of nodal values of u and $\mathbf{q}$ in element K_i , and Λ^i is the vector of nodal values of $\hat{u}$ on the n faces of the element ($n = 3$ for triangles and $n = 4$ for tetrahedra). That is,

$$\Lambda^i := \begin{bmatrix} \hat{\mathbf{u}}^{\mathbf{F}_{i1}} \\ \vdots \\ \hat{\mathbf{u}}^{\mathbf{F}_{in}} \end{bmatrix}, \quad (10)$$

where $\hat{\mathbf{u}}^f$ denotes the nodal values of $\hat{u}$ on face Γ_f , and $\mathbf{F}_{ij}$ is the number of the j -th face of element K_i , see an example in Fig. 3. Note that the subindexes in the $\mathbf{A}$ matrices refer to the space for the weighting function and the test function.

System (9) can be solved for $\mathbf{u}^i$ and $\mathbf{q}^i$ in each element, obtaining the so-called *local solver* in the element K_i

$$\mathbf{u}^i = \mathbf{U}^{K_i} \Lambda^i + \mathbf{f}_U^{K_i}, \quad \mathbf{q}^i = \mathbf{Q}^{K_i} \Lambda^i + \mathbf{f}_Q^{K_i}, \quad (11)$$

Fig. 4 X-HDG discretization in an element cut by the interior Neumann boundary: *grey* for elemental variables, u and $\mathbf{q}$, *black nodes* for trace variable $\widehat{u}$ and *white nodes* for the trace on the Neuman boundary $\widetilde{u}^i$

with

$$\begin{bmatrix} \mathbf{U}^{K_i} \\ \mathbf{Q}^{K_i} \end{bmatrix} = -\mathbb{A}^{-1} \begin{bmatrix} \mathbf{A}_{u\widehat{u}}^{K_i} \\ \mathbf{A}_{q\widehat{u}}^{K_i} \end{bmatrix}, \quad \begin{bmatrix} \mathbf{f}_U^{K_i} \\ \mathbf{f}_Q^{K_i} \end{bmatrix} = \mathbb{A}^{-1} \begin{bmatrix} \mathbf{f}_u^{K_i} \\ \mathbf{0} \end{bmatrix} \quad (12)$$

and

$$\mathbb{A} = \begin{bmatrix} \mathbf{A}_{uu}^{K_i} & \mathbf{A}_{uq}^{K_i} \\ \mathbf{A}_{qu}^{K_i} & \mathbf{A}_{qq}^{K_i} \end{bmatrix}$$

That is, for each element, the elemental values of the solution, $\mathbf{u}^i$ and $\mathbf{q}^i$, can be explicitly expressed in terms of the trace on its faces, Λ^i .

2.2 Local Problem for a Cut Element

The X-HDG local problem at an element K_i cut by the interior boundary corresponds to the discretization of (2b), that is: given $\widehat{u} \in \Lambda^h$, find $u \in \mathcal{P}_k(\Omega_i)$, $\mathbf{q} \in [\mathcal{P}_k(\Omega_i)]^d$ such that

$$\begin{aligned} \int_{\Omega_i} v \nabla \cdot \mathbf{q} \, dV + \int_{\partial\Omega_i \setminus \mathcal{I}_i} \tau v (u - \widehat{u}) \, dS + \int_{\mathcal{I}_i} \tau v (u - \widetilde{u}^i) \, dS &= \int_{\Omega_i} v f \, dV \\ \int_{\Omega_i} \mathbf{q} \cdot \mathbf{w} \, dV - \int_{\Omega_i} v u \nabla \cdot \mathbf{w} \, dV + \int_{\partial\Omega_i \setminus \mathcal{I}_i} v \widehat{u} \mathbf{w} \cdot \mathbf{n} \, dS + \int_{\mathcal{I}_i} v \widetilde{u}^i \mathbf{w} \cdot \mathbf{n} \, dS &= 0 \end{aligned} \quad (13)$$

for all $v \in \mathcal{P}_k(\Omega_i)$ and $\mathbf{w} \in [\mathcal{P}_k(\Omega_i)]^d$, where $\Omega_i = \Omega \cap K_i$, and $\widetilde{u}^i$ is a new trace variable approximating the trace of the solution on the interface $\mathcal{I}_i = \mathcal{I} \cap K_i$, see Fig. 4. Compared to the weak form for standard elements (7), the X-HDG weak form for a cut element has two additional terms corresponding to integrals along the interface $\mathcal{I}_i$, involving the new trace variable $\widetilde{u}^i$.

The discretization of the local problem (13) leads to a system of equations of the form

$$\begin{cases} \left[\mathbf{A}_{uu}^{\Omega_i} + \mathbf{A}_{uu}^{\mathcal{I}_i} \right] \mathbf{u}^i + \mathbf{A}_{uq}^{\Omega_i} \mathbf{q}^i + \mathbf{A}_{u\widehat{u}}^{\Omega_i} \Lambda^i + \mathbf{A}_{u\widetilde{u}^i}^{\mathcal{I}_i} \widetilde{u}^i = \mathbf{f}_u^{\Omega_i} \\ \mathbf{A}_{qu}^{\Omega_i} \mathbf{u}^i + \mathbf{A}_{qq}^{\Omega_i} \mathbf{q}^i + \mathbf{A}_{q\widehat{u}}^{\Omega_i} \Lambda^i + \mathbf{A}_{q\widetilde{u}^i}^{\mathcal{I}_i} \widetilde{u}^i = \mathbf{0} \end{cases}, \quad (14)$$

similar to (9), but with three new matrices corresponding to integrals on the Neumann boundary (marked with the superindex $\mathcal{I}_i$), and the nodal values for the new trace variable, $\widetilde{u}^i$.

The local problem is now closed by imposing the Neumann boundary condition on $\mathcal{I}_i$, i.e.

$$\widehat{\mathbf{q}} \cdot \mathbf{n} = g \quad \text{on } \mathcal{I}_i.$$

Replacing the expression of the numerical flux $\widehat{\mathbf{q}}$ defined in (8), i.e. $\widehat{\mathbf{q}} \cdot \mathbf{n} = \mathbf{q} \cdot \mathbf{n} + \tau(u - \widetilde{u}^i)$, the weak form of the Neumann condition on $\mathcal{I}^i$ is: given $u \in \mathcal{P}_k(\Omega_i)$, $\mathbf{q} \in [\mathcal{P}_k(\Omega_i)]^d$, find $\widetilde{u}^i \in \mathcal{P}_k(\mathcal{I}_i)$ such that

$$\int_{\mathcal{I}_i} \widetilde{v} \mathbf{q} \cdot \mathbf{n} \, dS + \tau \int_{\mathcal{I}_i} \widetilde{v} (u - \widetilde{u}^i) \, dS = \int_{\mathcal{I}_i} \widetilde{v} g \, dS \quad \forall \widetilde{v} \in \mathcal{P}_k(\mathcal{I}_i). \quad (15)$$

Thus, the discretization of the Neumann boundary condition leads to a discrete equation of the form

$$\mathbf{A}_{\tilde{u}^i u}^{\mathcal{I}_i} \mathbf{u}^i + \mathbf{A}_{\tilde{u}^i q}^{\mathcal{I}_i} \mathbf{q}^i + \mathbf{A}_{\tilde{u}^i \tilde{u}^i}^{\mathcal{I}_i} \tilde{\mathbf{u}}^i = \mathbf{g},$$

that allows expressing the new trace values $\tilde{\mathbf{u}}^i$ in terms of elemental values

$$\tilde{\mathbf{u}}^i = \mathbf{T}_u^i \mathbf{u}^i + \mathbf{T}_q^i \mathbf{q}^i - \mathbf{t}^i. \quad (16)$$

with

$$\mathbf{T}_u^i = -[\mathbf{A}_{\tilde{u}^i \tilde{u}^i}^{\mathcal{I}_i}]^{-1} \mathbf{A}_{\tilde{u}^i u}^{\mathcal{I}_i}, \quad \mathbf{T}_q^i = -[\mathbf{A}_{\tilde{u}^i \tilde{u}^i}^{\mathcal{I}_i}]^{-1} \mathbf{A}_{\tilde{u}^i q}^{\mathcal{I}_i}, \quad \mathbf{t}^i = -[\mathbf{A}_{\tilde{u}^i \tilde{u}^i}^{\mathcal{I}_i}]^{-1} \mathbf{g}.$$

Replacing (16) in (14) leads to the final discrete local problem

$$\begin{cases} \left[\mathbf{A}_{uu}^{\Omega_i} + \mathbf{A}_{uu}^{\mathcal{I}_i} + \mathbf{A}_{u\tilde{u}^i}^{\mathcal{I}_i} \mathbf{T}_u^i \right] \mathbf{u}^i + \left[\mathbf{A}_{uq}^{\Omega_i} + \mathbf{A}_{u\tilde{u}^i}^{\mathcal{I}_i} \mathbf{T}_q^i \right] \mathbf{q}^i + \mathbf{A}_{u\tilde{u}^i}^{\Omega_i} \Lambda^i = \mathbf{f}_u^{\Omega_i} + \mathbf{A}_{u\tilde{u}^i}^{\mathcal{I}_i} \mathbf{t}^i \\ \left[\mathbf{A}_{qu}^{\Omega_i} + \mathbf{A}_{q\tilde{u}^i}^{\mathcal{I}_i} \mathbf{T}_u^i \right] \mathbf{u}^i + \left[\mathbf{A}_{qq}^{\Omega_i} + \mathbf{A}_{q\tilde{u}^i}^{\mathcal{I}_i} \mathbf{T}_q^i \right] \mathbf{q}^i + \mathbf{A}_{q\tilde{u}^i}^{\Omega_i} \Lambda^i = \mathbf{A}_{q\tilde{u}^i}^{\mathcal{I}_i} \mathbf{t}^i \end{cases}$$

Now, similarly to (9), this system can be solved for $\mathbf{u}^i$ and $\mathbf{q}^i$, obtaining the *local solver* in the cut element K_i , i.e. Eq. (11) with

$$\begin{bmatrix} \mathbf{U}^{K_i} \\ \mathbf{Q}^{K_i} \end{bmatrix} = -\mathbb{A}^{-1} \begin{bmatrix} \mathbf{A}_{u\tilde{u}^i}^{\Omega_i} \\ \mathbf{A}_{q\tilde{u}^i}^{\Omega_i} \end{bmatrix}, \quad \begin{bmatrix} \mathbf{f}_U^{K_i} \\ \mathbf{f}_Q^{K_i} \end{bmatrix} = \mathbb{A}^{-1} \begin{bmatrix} \mathbf{f}_u^{\Omega_i} + \mathbf{A}_{u\tilde{u}^i}^{\mathcal{I}_i} \mathbf{t}^i \\ \mathbf{A}_{q\tilde{u}^i}^{\mathcal{I}_i} \mathbf{t}^i \end{bmatrix} \quad (17)$$

and

$$\mathbb{A} = \begin{bmatrix} \left[\mathbf{A}_{uu}^{\Omega_i} + \mathbf{A}_{uu}^{\mathcal{I}_i} + \mathbf{A}_{u\tilde{u}^i}^{\mathcal{I}_i} \mathbf{T}_u^i \right] & \left[\mathbf{A}_{uq}^{\Omega_i} + \mathbf{A}_{u\tilde{u}^i}^{\mathcal{I}_i} \mathbf{T}_q^i \right] \\ \left[\mathbf{A}_{qu}^{\Omega_i} + \mathbf{A}_{q\tilde{u}^i}^{\mathcal{I}_i} \mathbf{T}_u^i \right] & \left[\mathbf{A}_{qq}^{\Omega_i} + \mathbf{A}_{q\tilde{u}^i}^{\mathcal{I}_i} \mathbf{T}_q^i \right] \end{bmatrix}$$

It is important noting that the structure of the local solver is exactly the same as for non-cut elements (11), thanks to the fact that the internal trace variable $\tilde{\mathbf{u}}^i$ has been isolated and it is not an unknown of the problem anymore.

Remark 2 Although the approximation in a cut element, or on a cut face, is defined as polynomial functions in $\Omega_i = K_i \cap \Omega$, or on $\Gamma_f \cap \Omega$, respectively, standard nodal basis functions in the whole element K_i , or face Γ_f , are considered, see Fig. 4. Thus, as usual in X-FEM, the reference element is the standard one but with a modified numerical quadrature to integrate only in the domain Ω , see Sect. 4.

Remark 3 The trace vector Λ^i can be defined to include the nodal values only for faces intersecting the domain. However, the implementation is simpler if all faces of the element are considered. In this case, the blocks in matrices $\mathbf{U}^{K_i}$ and $\mathbf{Q}^{K_i}$ and in vectors $\mathbf{f}_U^{K_i}$ and $\mathbf{f}_Q^{K_i}$, corresponding to faces not intersecting the domain, are zero blocks; and, therefore, the trace values for these faces do not contribute to the final system.

2.3 Global Problem

The local problem, both in a standard element or in a cut element, leads to the local solver (11), that allows the computation of the solution in the element, $\mathbf{u}$ and $\mathbf{q}$, in terms of the trace values at its boundary, $\hat{\mathbf{u}}$. Thus, now the problem is reduced to determine the trace nodal values $\{\hat{\mathbf{u}}^f\}_{f=1}^{n_{rc}}$ on the mesh skeleton Γ . For this purpose the so-called *global problem* is stated, which corresponds to the discretization of the conservativity condition on Γ (4).

Replacing $\mathbf{q}$ by the numerical flux (8), the weak form for the trace variable is: find $\widehat{u} \in \Lambda^h$ such that $\widehat{u} = u_D$ on $\partial\Omega^{ext}$ and

$$\int_{\Gamma} \widehat{v} [\mathbf{q} \cdot \mathbf{n}] dS + 2\tau \int_{\Gamma} \widehat{v} (\{u\} - \widehat{u}) dS = 0 \quad \forall \widehat{v} \in \Lambda^h,$$

where $\{\cdot\}$ is the mean operator on the faces,

$$\{\odot\} = \frac{1}{2} (\odot_{L(f)} + \odot_{R(f)}) \quad \text{on } \Gamma_f.$$

As usual in HDG, the discretization of this equation for every face Γ_f leads to an equation of the form

$$\mathbf{A}_{\widehat{u}\widehat{u}}^{f,L} \mathbf{u}^{L(f)} + \mathbf{A}_{\widehat{u}\mathbf{q}}^{f,L} \mathbf{q}^{L(f)} + \mathbf{A}_{\widehat{u}\mathbf{u}}^{f,R} \mathbf{u}^{R(f)} + \mathbf{A}_{\widehat{u}\mathbf{q}}^{f,R} \mathbf{q}^{R(f)} + \mathbf{A}_{\widehat{u}\widehat{u}}^f \widehat{\mathbf{u}}^f = 0. \quad (18)$$

Replacing the local solver (11), for the elements $K_{L(f)}$ and $K_{R(f)}$, in (18) for every face Γ_f , leads to a system of equations involving only the trace variables $\{\widehat{\mathbf{u}}^f\}_{f=1}^{n_{fc}}$.

As usual in a standard HDG code, the implementation of the method involves a loop over elements. For each element, the matrices and vectors for the local solver (11) are computed, and the contribution to the Eq. (18) is assembled for each one of the faces of the element. Once the system is assembled for all elements, and Dirichlet boundary conditions (5) are imposed, the system can be solved. Then, given the trace variables $\{\widehat{\mathbf{u}}^f\}_{f=1}^{n_{fc}}$, the solution, $\mathbf{u}^i$ and $\mathbf{q}^i$, can be computed for each element using (11). It is important noting that X-HDG keeps the structure of a standard HDG code. The only difference is the modified local problem at cut elements (14), and the corresponding matrices in the local solver (17).

Remark 4 As commented in Remark 3, the simplest implementation assembles for all faces, even if they do not intersect the domain. The matrices assembled for faces not intersecting the domain are null matrices and, therefore, the corresponding rows and columns in global system are null. Those rows and columns are to be removed from the system, reducing its size and rendering the system solvable with unique solution, after imposing the Dirichlet boundary condition (5).

Remark 5 Boundary conditions on the exterior boundary $\partial\Omega^{ext} = \partial\Omega \setminus \mathcal{I}$ are implemented as in standard HDG. However, special care has to be taken in the presence of exterior faces cut by the interface $\mathcal{I}$, see Fig. 1 right. If a face Γ_f in the exterior Neumann boundary (i.e. $\Gamma_f \cap \overline{\Omega} \subset \partial\Omega^{ext}$ with Neumann boundary conditions) is cut by the interface, its contribution to the r.h.s. of the system is integrated only in the domain, that is on $\Gamma_f \cap \overline{\Omega}$, with a suitable numerical quadrature. For a face Γ_f intersecting the exterior boundary, with Dirichlet boundary conditions and cut by the interface, nodal values approximating the prescribed value on $\Gamma_f \cap \overline{\Omega}$ are set for all nodes of the face.

Remark 6 A second element-by-element postprocessing can be done to compute an X-HDG superconvergent solution. Similarly to standard HDG the superconvergent solution can be computed in every element K_i as the solution of: find $u^* \in \mathcal{P}_{k+1}(\Omega_i)$ such that

$$\begin{aligned} \int_{\Omega_i} v \nabla u^* \cdot \nabla v dV &= - \int_{\Omega_i} \mathbf{q} \cdot \nabla v dV \quad \forall v \in \mathcal{P}_{k+1}(\Omega_i), \\ \int_{\Omega_i} u^* dV &= \int_{\Omega_i} u dV, \end{aligned}$$

with $\Omega_i = K_i$ for standard elements, and $\Omega_i = K_i \cap \Omega$ for cut elements. The solution of this element-by-element computation, u^* , converges with order $k+2$ in the $\mathcal{L}_2$ norm. See [4, 7] for details and other possible computations of a superconvergent solution.

3 X-HDG Formulation for Problems with Dirichlet Voids

Let us now consider the problem with Dirichlet boundary conditions at the interior boundary $\mathcal{I}$,

$$\begin{aligned} -\nabla \cdot (v \nabla u) &= f && \text{in } \Omega \\ u &= u_{\mathcal{I}} && \text{on } \mathcal{I} \\ u &= u_D && \text{on } \partial\Omega \setminus \mathcal{I} \end{aligned} \quad (19)$$

where $u_{\mathcal{I}}$ is the prescribed value.

Similarly to the Neumann case, the problem is split element-by-element into local problems that are linked through the conservativity condition. The local problem for elements not cut by $\mathcal{I}$ is again the standard HDG local problem, leading to the same HDG local solver (11) with (12). For elements K_i cut by $\mathcal{I}$, the strong form of the local problem is now

$$\begin{aligned} \nabla \cdot \mathbf{q} &= f && \text{in } \Omega_i \\ \mathbf{q} + v \nabla u &= 0 && \text{in } \Omega_i \\ u &= u_{\mathcal{I}} && \text{on } \mathcal{I}_i \\ u &= \widehat{u} && \text{on } \partial\Omega_i \setminus \mathcal{I}_i, \end{aligned}$$

where the trace on the interior boundary $u_{\mathcal{I}}$ is a given data in this case.

The weak form of the local problem is then: given $\widehat{u} \in \Lambda^h$, find $u \in \mathcal{P}_k(\Omega_i)$, $\mathbf{q} \in [\mathcal{P}_k(\Omega_i)]^d$ such that

$$\begin{aligned} \int_{\Omega_i} v \nabla \cdot \mathbf{q} \, dV + \int_{\partial\Omega_i \setminus \mathcal{I}_i} \tau v (u - \widehat{u}) \, dS + \int_{\mathcal{I}_i} \tau v u \, dS &= \int_{\Omega_i} v f \, dV + \int_{\mathcal{I}_i} \tau v u_{\mathcal{I}} \, dS \\ \int_{\Omega_i} \mathbf{q} \cdot \mathbf{w} \, dV - \int_{\Omega_i} v u \nabla \cdot \mathbf{w} \, dV + \int_{\partial\Omega_i \setminus \mathcal{I}_i} v \widehat{u} \mathbf{w} \cdot \mathbf{n} \, dS &= - \int_{\mathcal{I}_i} v u_{\mathcal{I}} \mathbf{w} \cdot \mathbf{n} \, dS \end{aligned}$$

for all $v \in \mathcal{P}_k(\Omega_i)$ and $\mathbf{w} \in [\mathcal{P}_k(\Omega_i)]^d$. Compared to the weak form for standard elements (7), the X-HDG weak form for a cut element has three additional terms (one on the r.h.s., and two on the l.h.s. involving the data $u_{\mathcal{I}}$) corresponding to integrals along the interior boundary $\mathcal{I}_i$.

The discretization of the local problem on a cut element now leads to a system of equations of the form

$$\begin{cases} \left[\mathbf{A}_{uu}^{\Omega_i} + \mathbf{A}_{uu}^{\mathcal{I}_i} \right] \mathbf{u}^i + \mathbf{A}_{uq}^{\Omega_i} \mathbf{q}^i + \mathbf{A}_{u\widehat{u}}^{\Omega_i} \Lambda^i = \mathbf{f}^{\Omega_i} + \mathbf{f}_u^{\mathcal{I}_i} \\ \mathbf{A}_{qu}^{\Omega_i} \mathbf{u}^i + \mathbf{A}_{qq}^{\Omega_i} \mathbf{q}^i + \mathbf{A}_{q\widehat{u}}^{\Omega_i} \Lambda^i = \mathbf{f}_q^{\mathcal{I}_i} \end{cases},$$

similar to (9), but with an additional matrix $\mathbf{A}_{uu}^{\mathcal{I}_i}$ and two additional force vectors.

Now, this system can be solved for $\mathbf{u}^i$ and $\mathbf{q}^i$, obtaining the *local solver* in the cut element K_i , i.e. Eq. (11) with

$$\begin{bmatrix} \mathbf{U}^{K_i} \\ \mathbf{Q}^{K_i} \end{bmatrix} = -\mathbb{A}^{-1} \begin{bmatrix} \mathbf{A}_{u\widehat{u}}^{\Omega_i} \\ \mathbf{A}_{q\widehat{u}}^{\Omega_i} \end{bmatrix}, \quad \begin{bmatrix} \mathbf{f}_U^{K_i} \\ \mathbf{f}_Q^{K_i} \end{bmatrix} = -\mathbb{A}^{-1} \begin{bmatrix} \mathbf{f}_u^{\Omega_i} + \mathbf{f}_u^{\mathcal{I}_i} \\ \mathbf{f}_q^{\mathcal{I}_i} \end{bmatrix}$$

and

$$\mathbb{A} = \begin{bmatrix} \left[\mathbf{A}_{uu}^{\Omega_i} + \mathbf{A}_{uu}^{\mathcal{I}_i} \right] & \mathbf{A}_{uq}^{\Omega_i} \\ \mathbf{A}_{qu}^{\Omega_i} & \mathbf{A}_{qq}^{\Omega_i} \end{bmatrix}.$$

Finally the global HDG problem is formed completely and analogously to Sect. 2.3.

Fig. 5 3-th order triangle cut by an interface (*left*). Interface representation and numerical quadrature with (*center*) piecewise linear approximation based on an oct-tree partition, and (*right*) 3-th degree parametrization. The color on the nodes refer to the sign of the level set function. Crosses are integration points

Remark 7 The extension of X-HDG for problems with both Neumann and Dirichlet interfaces is straightforward, just considering the local problem stated in Sect. 2.2 for elements cut by a Neumann interface, and the local problem in this section for elements cut by a Dirichlet interface.

4 Numerical Integration on Cut Elements

In this work the usual X-FEM strategy for the geometrical description of interfaces and numerical integration is adopted. An interface $\mathcal{I}$ is represented as the iso-zero values of a level set function φ , that is $\mathcal{I} = \{x \mid \varphi(x) = 0\}$. The level set function φ is assumed to be given by its nodal values in the computational mesh, see Fig. 5 left. The nodal values of the level set function may correspond, for instance, to the signed distance of the nodes to the interface $\mathcal{I}$.

A proper representation of the interface inside cut elements is crucial. Those elements are split by the interface in two subregions, corresponding to the domain and the void, and a numerical quadrature has to be defined to integrate only over the material domain. The usual practice for first order computations is considering a linear interface representation in each cut element. This strategy provides optimal convergence rates for linear approximations, but it is clearly not suitable for high-order computations. The geometrical error due to the low resolution representation of the interface leads to poor accuracy and convergence rates limited to order $\mathcal{O}(h^{3/2})$ in $\mathcal{H}^1$ norm, see for instance [10]. Two main strategies have been proposed in the literature to properly represent an interface for integration purposes in k -th order computations: (i) a fine enough piecewise linear representation of the interface in each cut element [10, 15] or (ii) a k -th degree parametrization to approximate the interface [1].

In [10], an octree-like partition of the element in integration cells is recursively defined to get a piecewise linear representation of the interface, with segments of the desired size $\tilde{h}$. Cells intersected by the interface are divided in two subregions for integration, see for example Fig. 5 center. Special care has to be taken to the level of refinement in order to get accurate results and optimal convergence rates. In fact, optimal asymptotic convergence can not be obtained with a constant ratio $\tilde{h}/h$ and, as noted in [10], further refinement of the integration cells is necessary as the computational mesh is refined. In fact, in practice, for a given problem and computational mesh, the accuracy of the solution may show strong dependence on the integration cell size $\tilde{h}$, and the selection of a small enough cell size may be not straightforward.

Fig. 6 Some examples of interfaces described by a 4-th order level set. The *color* on the nodes refer to the sign of the level set function

Fig. 7 Example of recursive division for an element cut by the interface in a complex manner

Remark 8 In Figs. 5, 6 and 7, illustrating strategies for the approximation of the interface and numerical integration, the representation is done in the reference element, where all computations are actually done.

Alternatively a k -th degree parametrization for the representation of the interface in each element is proposed in [1], see for example Fig. 5 right, leading to accurate results and optimal convergence rates with less integration points (i.e. lower computational cost). However, the strategy proposed in [1] requires a fine enough mesh such that the interface cuts the element boundary twice and not within one side, splitting the triangle in a triangle and a quadrilateral. Thus, this strategy may fail in complicated situations that can usually appear with a high-order level set on a coarse mesh, such as a bubble inside one element, or an interface cutting more than two sides of an element, see Fig. 6. To overcome this limitation, a robust and efficient strategy, based on the strategy in [1] but capable to handle more complicated situations, is proposed next.

4.1 Numerical Integration at Cut Elements with Piecewise k -th Degree Interfaces

In [1], Cheng and Fries consider a k -th degree polynomial parametrization for the representation of the interface in triangular elements. In their work, the mesh is assumed to be fine enough so that all cut elements correspond to what here will be called *basic* element; that is, the interface splits the triangle in a triangular and quadrilateral region or, in other words, the interface cuts two sides of the triangle with just one intersection on each side. Under this assumption, $k + 1$ points on the interface are to be found first, including the two intersections with the element boundary. The $k + 1$ points are used as base points for a k -th degree polynomial parametrization ϕ approximating the interface, and dividing the element in two integration subdomains sharing one curved side, see Fig. 5 right.

A transformation from a straight-sided element can now be used to define the numerical quadrature in each subdomain, or just in the region in the domain Ω in our case. To avoid generating all the nodes necessary for the use of the standard isoparametric transformation,

which would require a proper location of interior nodes [2] in each subregion, the use of specially designed transformations for elements with only one curved side is recommended here, see for instance [23]. In particular, in a *basic* element cut by an interface parametrized by $\phi(s)$ for $s \in (-1, 1)$, the following transformations can be used:

$$(\xi(s, t), \eta(s, t)) = \frac{1-t}{2} \mathbf{C} + \frac{1+t}{2} \phi(s), \quad s, t \in (-1, 1)$$

for a triangular subdomain corresponding to the convex hull of the curved side and node $\mathbf{C}$, and

$$(\xi(s, t), \eta(s, t)) = \frac{1-t}{2} \left[\frac{1-s}{2} \mathbf{C} + \frac{1+s}{2} \mathbf{D} \right] + \frac{1+t}{2} \phi(s), \quad s, t \in (-1, 1)$$

for a curved quadrilateral corresponding to the convex hull of the curved size and nodes $\mathbf{C}$ and $\mathbf{D}$ (for a quadrilateral $\{\phi(-1), \phi(1), \mathbf{D}, \mathbf{C}\}$ being properly oriented, or switching $\mathbf{C}$ and $\mathbf{D}$ otherwise).

Numerical examples in [1] and Sect. 5 illustrate how the approximation of the interface with a piecewise k -th degree parametrization in each element leads to optimal convergence rates. Nevertheless, in practical applications, the assumption of a fine enough mesh, such that all cut elements are *basic* elements, is too restrictive for high-order computations and hampers the robustness of the method. High-order level sets may lead to very complex interfaces, see some examples in Fig. 6, that are not contemplated in [1]. In a more general context, an interface can split an element in more than two regions. Moreover, even if it is split in only two regions, casuistic (interior bubble, two cuts within one side, etc) may impede the implementation of a unique k -th degree polynomial parametrization.

Here a simple idea is proposed: divide and conquer. That is, any element intersected by the interface in a complex manner (i.e. not a *basic* element) is recursively divided in integration cells until all cells can be considered either *basic* cells—i.e. cells whose boundary is cut by the interface twice not within one side—or not intersected by the interface. Nodal values of the level-set function are interpolated from the original element to get nodal values in the integration cells. Then, the strategy considering a k -th degree parametrization of the interface is applied for all *basic cells*, leading to a piecewise k -th degree interface representation in the element. Note that, if the recursive division in cells is done using the same type of element, the resulting level set is exactly the same polynomial level set, and no information is lost during the process.

Remark 9 The recursive division in cells is done only for integration purposes—that is, to define a numerical quadrature in the original k -th order element—and it does not represent any change in the computational mesh or the approximation space in the element. The only variable that can be affected by the splitting is the approximation of the trace along the interface in the case of Neumann boundary conditions, $\tilde{u}^i$ in (13). For simplicity, this variable can be approximated using a piecewise k -th order approximation, given by the $k + 1$ nodes defining the interface parametrization in each cut basic cell. Nevertheless, recall that this variable is isolated in (16) at elemental level, thus the number of degrees of freedom of the elemental variables and trace variables in the mesh skeleton is not affected by the splitting.

For an easy implementation, the decision of splitting an element or cell can be based on the changes of sign of the level-set nodal values, first looking to nodal values at sides and then looking to interior nodal values. An element or cell is to be split if:

- (i) one of its sides is cut more than once by the interface (there are more than one changes of sign of the nodal values of the level set in the side), or

Fig. 8 Basic tetrahedrons for 3D computations. A tetrahedron cut by the interface in any other situation is recursively divided

- (ii) it has more than two sides cut by the interface, or
- (iii) its boundary is not intersected by the interface, but it has one or more interior nodes with sign of the level set different to the boundary sign (interior bubble or void).

Figure 7 shows an example of recursive division and the corresponding numerical quadrature for a 3rd order finite element. Nodal values with different sign are marked with different colors.

The strategy considering a piecewise k -th degree parametrization to approximate the interface provides optimal convergence rates for any order, in a robust manner. Moreover, in practical applications most of the elements cut by the interface will be in the *basic* situation, and paying little attention to the casuistic for the recursive division in the complex elements, they will usually be divided in few integration cells, with little increase in computational cost.

Remark 10 It is worth noting that X-HDG has an additional cost compared to HDG, mainly due to the modification of the numerical quadrature for cut elements, and the evaluation of the nodal basis functions and derivatives at the new integration points. That is, elemental computations for cut elements are substantially more expensive than for standard elements, for which the reference element information and all associated pre-computations can be used. However, in practical applications the ratio of cut elements to standard elements is small so that the extra cost becomes negligible compared with the cost of mesh adaptation or re-meshing in case of moving boundaries.

A similar strategy can be followed for 3D computations. That is, non-trivial elements are recursively divided into integration cells until all cells correspond to either a *basic* cell or not cut by the interface. For instance, for tetrahedral elements two types of *basic* cells can be considered corresponding to a tetrahedron divided by the interface in (i) a tetrahedron and a triangular prism, sharing a triangular curved face, or (ii) two triangular prisms, sharing a curved quadrilateral face, see Fig. 8. A tetrahedron cut by the interface in any other situation (interior bubbles, only one face is intersected, more complex intersection in a face, etc) is recursively divided into integration cells. Then, at each basic integration cell the interface is approximated with a k -th degree parametrization, given by $k(k+1)/2$ nodes for a triangular curved face in case (i), or $(k+1)^2$ nodes for a quadrilateral curved face in case (ii).

5 Numerical Tests

The performance of the novel X-HDG method is tested on three numerical examples. A Laplace equation with known analytical solution is solved over a square domain with a

Fig. 9 Circular void example: domain Ω and void boundary $\mathcal{I}$, and analytical solution

circular void. Boundary condition on the circular boundary are of Neumann type in the first example, and of Dirichlet type in the second example. The accuracy and the convergence of X-HDG is tested and compared to HDG with a mesh adapted to the void boundary.

Eventually, a potential flow, or flow through a porous medium, is modeled using the proposed X-HDG method. Impermeable stones are modeled as voids in the computational domain, with homogeneous Neumann boundaries. In this case, the analytical solution is unknown and comparison with standard HDG is done with respect to a highly resolved reference solution on a fitted mesh.

In all numerical tests, the stabilization parameter is $\tau = 1$ on all faces.

5.1 Neumann Void on a Square Domain

The first numerical example is designed to test the performance of X-HDG with Neumann boundary conditions imposed at the void interface. The Laplace equation (1), with $v = 1$, is solved over a square domain with a centred circular void with radius 0.41, $\Omega = (-1, 1)^2 \setminus B((0, 0), 0.41)$. Neumann boundary conditions are imposed on the void boundary $\mathcal{I} = \partial B((0, 0), 0.41)$, and Dirichlet boundary conditions are imposed on the exterior boundary $\partial\Omega \setminus \mathcal{I} = \partial((-1, 1)^2)$. Dirichlet and Neumann values and the source term f are set so that the analytical solution is

$$u(x, y) = \exp(0.1 \sin(5.1x - 6.2y) + 0.3 \cos(4.3x + 3.4y)).$$

Figure 9 shows the domain and the analytical solution.

Figure 11 shows the computational mesh for X-HDG and for standard HDG. For the X-HDG computation, a regular triangular mesh in the square domain $(-1, 1)^2$ is considered, covering the domain Ω and fitting the exterior boundary $\partial\Omega \setminus \mathcal{I}$. A level set function is used to describe the boundary of the void, $\mathcal{I}$. Three kinds of elements appear in the computational mesh. Elements inside the domain (dark gray) are treated as standard HDG elements. For elements cut by the interior boundary $\mathcal{I}$ (light gray) the modified X-HDG local problem is considered, see Sect. 2.2. The elements that are totally inside the void (white) have no contribution to the solution, so they are simply disregarded. The computational mesh for standard HDG is adapted to fit the void boundary, with similar uniform mesh size.

Figure 10 shows the evolution of the $\mathcal{L}_2$ error for decreasing uniform mesh size, for X-HDG and HDG, with degree $k = 2, 3, 4$, for both the solution and the postprocessed

Fig. 10 Neumann circular void example: convergence plots for X-HDG (left) and HDG (right). The numbers correspond to the slope of each segment, and they are *underlined* for postprocessed solution

Fig. 11 Circular void example: computational mesh for X-HDG and for standard HDG. The X-HDG mesh is not adapted to the void boundary. Elements in the interior of the domain are colored in dark grey. Elements in softer gray are elements cut by the interface $\mathcal{I}$. Elements in white are inside the void, and are not considered in the computation

superconvergent solution, see Remark 6. X-HDG keeps HDG optimal convergence with rates close to $k + 1$ for the solution, and $k + 2$ for the postprocessed solution, with similar levels of accuracy, while getting rid of adapting the mesh to the void boundary.

Remark 11 To ensure $k + 2$ convergence rate of the postprocessed solution, a $k + 1$ degree approximation of the interface should be considered. For standard HDG this is equivalent to requiring a $k + 1$ degree mesh, properly fitting the description of the boundary, for the superconvergence postprocessing.

5.2 Dirichlet Void on a Square Domain

In this second example, Dirichlet boundary conditions are imposed on the void boundary $\mathcal{I}$, testing the performance of X-HDG for Dirichlet interfaces. The domain, boundary conditions on exterior boundary and source term definitions are the same as for the first example, with same analytical solution.

Fig. 12 Dirichlet circular void example: convergence plots for X-HDG (left) and HDG (right). The numbers correspond to the slope of each segment, and they are underlined for postprocessed solution

Fig. 13 Domain for the potential flow example. Homogeneous Neumann boundary conditions are imposed on the voids interfaces $\mathcal{I}$ and on the top and bottom boundaries. Dirichlet boundary conditions driving the flow are imposed at left and right sides

Following the X-HDG methodology for Dirichlet voids explained in Sect. 3, the numerical solution is obtained for degree $k = 2, 3, 4$ for different mesh sizes. Like in the Neumann case, convergence plots for X-HDG and standard HDG (with mesh non-adapted and adapted to the circular boundary respectively, see Fig. 11) are presented in Fig. 12. Again, one to one resemblance is observed, keeping optimal HDG convergence properties with an unfitted computational mesh.

5.3 Potential Flow Example

A real-life example is considered to study the performance of X-HDG: a potential flow problem. The domain is the rectangle $(0, 10) \times (-3, 3)$ with several voids with boundaries $\mathcal{I}$, see Fig. 13. The voids correspond to three circles with center and radius $\{(2, -1.5), 0.6\}$, $\{(3, 1.5), 1.3\}$ and $\{(5, -1.5), 0.8\}$, and an ellipsoid centered at $(9, 0)$ with x-radius 0.5 and y-radius 2.4. Homogeneous Neumann boundary conditions are set at the interior boundaries $\mathcal{I}$, modeling for instance impermeable rocks in the domain, and also at the top and bottom boundaries, $\{y = -3\}$ and $\{y = 3\}$, to avoid leakage. The flow is driven by setting the potential to $u = 10$ at the inflow boundary $\{x = 0\}$ and $u = 0$ at the outflow boundary $\{x = 10\}$. The source term in this test is $f = 0$, and the viscosity is set to $\nu = 1$.

A structured triangle mesh in the rectangle $(0, 10) \times (-3, 3)$ is considered for the X-HDG solution, representing the voids boundaries $\mathcal{I}$ with a level-set function. For comparison, an adapted mesh with similar mesh size is considered for a standard HDG solution. Both computational meshes are shown in Fig. 14.

Fig. 14 Flow example: computational mesh for X-HDG (left) and HDG (right)

Fig. 15 Potential flow example: streamlines for a degree $k = 3$ computation with HDG (solid gray line) and X-HDG (dashed black line)

In this example the output of interest is the velocity field $\mathbf{q}$. Figure 15 shows the velocity field and the streamlines for a computation with degree $k = 3$ for HDG, with the mesh fitting to the boundary, and for X-HDG, with the structured unfitted mesh. The streamlines for HDG and X-HDG overlap, demonstrating the applicability and good performance of the X-HDG method. To further compare the X-HDG and HDG solutions, the horizontal and vertical component of the velocity along the vertical line $\{x = 7.7\}$ are plotted in Fig. 16. No analytical solution is available for this problem; thus, a standard HDG reference solution computed on an adapted highly resolved computational mesh is considered as reference solution. Again, both HDG and X-HDG provide accurate results. The error of the HDG and the X-HDG solution along $\{x = 7.7\}$ when compared to the reference solution is plotted in Fig. 17. Similar accuracy, with errors of order 10^{-3} , are obtained with both standard HDG and X-HDG with similar mesh size, demonstrating again the reliability of X-HDG.

These results assert that X-HDG keeps the accuracy of HDG without the need to adapt the mesh to the boundary.

6 Conclusions and Final Remarks

A strategy for the HDG solution of boundary value problems with voids, inclusions or free boundaries is proposed. It is based on an X-FEM philosophy with a level-set description of

Fig. 16 Potential flow example: horizontal (*left*) and vertical (*right*) components of the velocity along the vertical line $\{x = 7.7\}$, for HDG, X-HDG and a reference solution, with degree $k = 3$

Fig. 17 Potential flow example: error of the HDG and the X-HDG solution along $\{x = 7.7\}$ when compared to the reference solution, for the horizontal component (*left*) and vertical component (*right*) of the velocity

interfaces. With this new strategy, the computational mesh does not need to be adapted to the interface (i.e. the boundary) simplifying and reducing the cost of mesh generation. In case of evolving interfaces, the cost of continuous re-meshing, and the approximation errors due to the projection from one mesh to another, are avoided.

Differently to previous proposals for HDG solution with meshes non-fitting the boundary, here the computational mesh covers the domain, avoiding extrapolations, and ensuring the robustness of the method. The local problem at elements not cut by the interface, and the global problem, are discretized as usual in HDG. A modified local problem is considered at elements cut by the interface. At every cut element, an auxiliary trace variable on the boundary is introduced, which is eliminated afterwards using the boundary conditions on the interface, keeping the original unknowns and the structure of the local problem solver. An efficient and robust methodology for numerical integration in cut elements, in the context of high-order approximations, is also proposed. Although this modified numerical integration in cut elements causes slight increase in CPU time, in practical applications the extra cost becomes negligible compared with the cost of mesh adaptation or re-meshing.

The proposed method is compared to standard HDG in three numerical tests: the solution of a Laplace problem with known analytical solution in a square domain with a circular void, with Dirichlet and Neumann boundary conditions, and the computation of the velocity

field for a potential flow problem in a rectangular domain with several voids. The HDG computations are done on a mesh fitting the voids boundaries, whereas with X-HDG a regular mesh covering the domain can be considered. Numerical tests assert that X-HDG keeps the HDG optimal convergence rates for the solution, the gradient and the post-processed superconvergent solution, without the need to adapt the mesh to the boundary. In all tests, similar accuracy is observed for similar mesh size.

The X-HDG method is fully developed here for the solution of steady void problems, describing the main ideas and fundamentals of the method, that can now be extended to moving boundaries and bimaterial problems.

Acknowledgments This work was supported by the DAFOH2 Project (Ministerio de Economía y Competitividad, MTM2013-46313-R), and the Erasmus Mundus Joint Doctorate SEED Project (European Commission)

References

1. Cheng, K.W., Fries, T.P.: Higher-order XFEM for curved strong and weak discontinuities. *Int. J. Numer. Methods Eng.* **82**(5), 564–590 (2010)
2. Ciarlet, P.G., Raviart, P.A.: Interpolation theory over curved elements, with applications to finite element methods. *Comput. Methods Appl. Mech. Eng.* **1**(2), 217–249 (1972)
3. Cockburn, B.: Discontinuous Galerkin methods for computational fluid dynamics. In: Stein, E., De Borst, R., Hughes, T.J.R. (eds.) *Encyclopedia of Computational Mechanics*, vol. 3 (Fluids), chap. 4. Wiley, New York (2004)
4. Cockburn, B., Dong, B., Guzmán, J.: A superconvergent LDG-hybridizable Galerkin method for second-order elliptic problems. *Math. Comput.* **77**(264), 1887–1916 (2008)
5. Cockburn, B., Gopalakrishnan, J., Lazarov, R.: Unified hybridization of discontinuous Galerkin, mixed, and continuous Galerkin methods for second order elliptic problems. *SIAM J. Numer. Anal.* **47**(2), 1319–1365 (2009)
6. Cockburn, B., Gopalakrishnan, J., Nguyen, N.C., Peraire, J., Sayas, F.J.: Analysis of HDG methods for Stokes flow. *Math. Comput.* **80**(274), 723–760 (2011)
7. Cockburn, B., Qiu, W., Shi, K.: Conditions for superconvergence of HDG methods for second-order elliptic problems. *Math. Comput.* **81**(279), 1327–1353 (2012)
8. Cockburn, B., Qiu, W., Solano, M.: A priori error analysis for HDG methods using extensions from subdomains to achieve boundary conformity. *Math. Comput.* **83**(286), 665–699 (2014)
9. Cockburn, B., Solano, M.: Solving convection–diffusion problems on curved domains by extensions from subdomains. *J. Sci. Comput.* **59**(2), 512–543 (2014)
10. Dreaux, K., Chevaugnon, N., Moës, N.: Studied X-FEM enrichment to handle material interfaces with higher order finite element. *Comput. Methods Appl. Mech. Eng.* **199**(29–32), 1922–1936 (2010)
11. Fries, T.P., Belytschko, T.: The extended/generalized finite element method: an overview of the method and its applications. *Int. J. Numer. Methods Eng.* **84**(3), 253–304 (2010)
12. Giorgiani, G., Fernández-Méndez, S., Huerta, A.: Hybridizable discontinuous Galerkin p-adaptivity for wave propagation problems. *Int. J. Numer. Methods Fluids* **72**(12), 1244–1262 (2013)
13. Giorgiani, G., Fernández-Méndez, S., Huerta, A.: Hybridizable discontinuous Galerkin with degree adaptivity for the incompressible Navier–Stokes equations. *Comput. Fluids* **98**, 196–208 (2014)
14. Giorgiani, G., Modesto, D., Fernández-Méndez, S., Huerta, A.: High-order continuous and discontinuous Galerkin methods for wave problems. *Int. J. Numer. Methods Fluids* **73**(10), 883–903 (2013)
15. Gross, S., Reusken, A.: An extended pressure finite element space for two-phase incompressible flows with surface tension. *J. Comput. Phys.* **224**(1), 40–58 (2007)
16. Hesthaven, J., Warburton, T.: Nodal high-order methods on unstructured grids: I. Time-domain solution of Maxwell’s equations. *J. Comput. Phys.* **181**(1), 186–221 (2002)
17. Huerta, A., Angeloski, A., Roca, X., Peraire, J.: Efficiency of high-order elements for continuous and discontinuous Galerkin methods. *Int. J. Numer. Methods Eng.* **96**(9), 529–560 (2013)
18. Kirby, R.M., Sherwin, S.J., Cockburn, B.: To CG or to HDG: a comparative study. *J. Sci. Comput.* **51**(1), 183–212 (2011)
19. Montlaur, A., Fernández-Méndez, S., Huerta, A.: Discontinuous Galerkin methods for the Stokes equations using divergence-free approximations. *Int. J. Numer. Methods Fluids* **57**(9), 1071–1092 (2008)

20. Nguyen, N., Peraire, J., Cockburn, B.: A hybridizable discontinuous Galerkin method for Stokes flow. *Comput. Methods Appl. Mech. Eng.* **199**(9–12), 582–597 (2010)
21. Nguyen, N.C., Peraire, J., Cockburn, B.: An implicit high-order hybridizable discontinuous Galerkin method for the incompressible Navier–Stokes equations. *J. Comput. Phys.* **230**(4), 1147–1170 (2011)
22. Peraire, J., Persson, P.O.: The compact discontinuous Galerkin (CDG) method for elliptic problems. *SIAM J. Sci. Comput.* **30**(4), 1806–1824 (2008)
23. Sevilla, R., Fernández-Méndez, S., Huerta, A.: Comparison of high-order curved finite elements. *Int. J. Numer. Methods Eng.* **87**(8), 719–734 (2011)
24. Wang, B., Khoo, B.C.: Hybridizable discontinuous Galerkin method (HDG) for Stokes interface flow. *J. Comput. Phys.* **247**, 262–278 (2013)