J Sci Comput (2012) 51:183–212
DOI 10.1007/s10915-011-9501-7

To CG or to HDG: A Comparative Study

Robert M. Kirby · Spencer J. Sherwin · Bernardo Cockburn

Received: 27 January 2011 / Revised: 26 May 2011 / Accepted: 27 May 2011 /
Published online: 1 July 2011

Abstract Hybridization through the border of the elements (hybrid unknowns) combined with a Schur complement procedure (often called *static condensation* in the context of continuous Galerkin linear elasticity computations) has in various forms been advocated in the mathematical and engineering literature as a means of accomplishing domain decomposition, of obtaining increased accuracy and convergence results, and of algorithm optimization. Recent work on the hybridization of mixed methods, and in particular of the discontinuous Galerkin (DG) method, holds the promise of capitalizing on the three aforementioned properties; in particular, of generating a numerical scheme that is discontinuous in both the primary and flux variables, is locally conservative, and is computationally competitive with traditional continuous Galerkin (CG) approaches. In this paper we present both implementation and optimization strategies for the Hybridizable Discontinuous Galerkin (HDG) method applied to two dimensional elliptic operators. We implement our HDG approach within a spectral/*hp* element framework so that comparisons can be done between HDG and the traditional CG approach.

We demonstrate that the HDG approach generates a global trace space system for the unknown that although larger in rank than the traditional static condensation system in CG, has significantly smaller bandwidth at moderate polynomial orders. We show that if one ignores set-up costs, above approximately fourth-degree polynomial expansions on triangles

Work of R.M. Kirby accomplished while on sabbatical at Imperial College London.
Work of B. Cockburn accomplished in part while visiting the Research Institute of Mathematical Sciences, Kyoto University, Kyoto, Japan, as part of a Single Semester Leave from the University of Minnesota.

R.M. Kirby (✉)
School of Computing, Univ. of Utah, Salt Lake City, UT, USA
e-mail: kirby@cs.utah.edu

S.J. Sherwin
Department of Aeronautics, Imperial College London, London, UK
e-mail: s.sherwin@imperial.ac.uk

B. Cockburn
School of Mathematics, Univ. of Minnesota, Minneapolis, MN, USA
e-mail: cockburn@math.umn.edu

and quadrilaterals the HDG method can be made to be as efficient as the CG approach, making it competitive for time-dependent problems even before taking into consideration other properties of DG schemes such as their superconvergence properties and their ability to handle *hp*-adaptivity.

Keywords High-order finite elements · Spectral/*hp* elements · Discontinuous Galerkin method · Hybridization · Domain decomposition

1 Introduction

Given a particular scientific or engineering problem of interest, the current computational science and engineering practitioner has at his or her disposal a large array of numerical methodologies from which to choose to tackle the problem. The choice of which method to employ is normally not made solely by one selection criterion such as asymptotic convergence rate, but rather upon a host of different competing criteria including numerical properties, geometric flexibility, robustness, time-to-implement, and computational cost. The purpose of this paper is to examine in the context of (symmetric second-order) elliptic partial differential equations (PDEs) two commonly used numerical techniques—the continuous Galerkin (CG) method and the discontinuous Galerkin (DG) method—and to provide guidance as to why one might select to use one versus the other when run-time versus a fixed error tolerance is the major discriminating criterion.

The literature concerning the continuous Galerkin (CG) methods, and in particular the finite element method (FEM), is vast. We refer the interested reader to [35, 50–52] for comprehensive discussions of the formulation and implementation of the method, and to [30, 37, 42, 48] for extensions to high-order (polynomial-enriched) finite elements. The salient feature of the finite element method for elliptic problems with respect to this work is that the FEM method provides a continuous, piecewise polynomial approximation of the solution which minimizes the residual of the solution in the integral sense. It is considered the mainstay numerical technique against which to make both accuracy and computational cost evaluations. On the other hand, the discontinuous Galerkin (DG) methods [20] provide what some consider to be a compact high-order extension of the finite volume method in a way similar to how high-order or spectral/*hp* elements [37, 48] extend standard finite elements. Like the finite element method, the DG methodology allows for a dual path to convergence through both elemental 'h' and polynomial 'p' refinement by providing a discontinuous, piecewise polynomial approximation of the solution that similarly minimizes the residual in the integral sense. In the overview of the development of the discontinuous Galerkin method [27] there is a succinct discussion of the merits of this compact extension of finite volumes; see also [15, 16]. In the context of diffusion problems, there are many different DG choices as presented in [3, 4] within a unified framework.

Since the inception of DG methods for elliptic problems, there have been speculations made as to whether DG methods can be made more computationally efficient than CG methods. That is, for a given error tolerance, can the run-time necessary to compute a DG solution be made more efficient than what is required to find a corresponding CG solution. This speculation has been fueled by the claim that although a DG solution has more degrees of freedom than a corresponding CG solution on the same mesh and at the same polynomial degree (due to not enforcing inter-element continuity and therefore not removing global degrees of freedom), the corresponding linear system that one generates is more weakly coupled and hence more amenable to various optimization techniques. DG advocates often highlight the fact that as one increases the polynomial degree per (geometric) element,

the number of degrees of freedom in the volumetric interior of an element increases more rapidly than the degrees of freedom on element boundaries, and hence, in spite of the extra degrees of freedom the DG methodology becomes more efficient for high-degree polynomial approximations. Advocates of the CG method have always pointed to the fact that, in addition to the lower degree of freedom count due to constraining the solution at the element boundaries, years of employing CG methods have led to a tremendous number of optimization techniques—not least among them *static condensation*—that, combined with the approximation properties of CG methods for elliptic problems, make finite elements a very tenable choice.

Until recently, domain decomposition and optimization techniques that exploit the decoupling of local and global degrees of freedom have only been available to the CG method and hence made any comparison between CG and DG methods unrealistic. However, the recent introduction of a static-condensation-amenable DG method—the hybridizable discontinuous Galerkin (HDG) method [22]—now makes a realistic comparison possible. The HDG method is a DG method that can be rendered hybrid since its *numerical trace of the scalar variable,* λ, can be expressed as the only globally coupled *unknown*; see the definition of a hybrid method in [14, p. 421]. It is the ability to hybridize the formulation that makes it possible to implement CG and HDG solvers in very similar ways.

In this paper, we present both implementation and optimization strategies for the HDG method applied to a two-dimensional model elliptic problem. We implement our HDG approach within a spectral/*hp* element framework so that comparisons can be done between HDG and the traditional CG approach. We demonstrate that the HDG approach generates a global system for the approximation on the boundaries of the elements that although larger in rank than the traditional static condensation system in CG, has significantly smaller bandwidth at moderate polynomial degrees. We show that, if one ignores set-up costs above approximately fourth-degree polynomial expansions on triangles and quadrilaterals the HDG method can be made as efficient as the CG approach, making it attractive for time-dependent problems.

1.1 Background

Let us provide a brief a review of the relevant literature related to the HDG method, with a particular emphasis on its connection with the traditional concept of static condensation as used in finite elements.

Hybridization was first introduced back in 1965 as a technique to efficiently implement and solve finite element approximations for linear elasticity [32]. Roughly speaking, the idea consists in expressing the unknowns in terms of the *hybrid* unknown λ in an element-by-element fashion; the function λ is called hybrid because it is defined only on the boundaries of the elements. The original variables are subsequently eliminated, giving rise to a global system of equations involving only λ.

In 1985, it was noted [2] that the hybrid unknown λ super-converged to a projection of the solution for the hybridized version of the mixed method of Raviart-Thomas (RT) for symmetric second-order elliptic problems [41]. This observation was then used to construct a means of enhancing the accuracy of the finite element approximation by using a local post-processing [2]. Similar post-processing results were obtained for the so-called Brezzi-Douglas-Marini (BDM) mixed method in [9]. Variations on post-processing have been proposed in [8, 33, 46, 47] and an extension to the Hellan-Herrmann-Johnson method in [28].

In 2004, hybridized RT and BDM methods of *arbitrary* order were revisited and a weak formulation for the hybrid unknown λ was found [17]; see [13] for the lowest order RT method. This weak formulation allowed for a new, effective assembly of the matrix equation for the degrees of freedom of λ. It also allowed for establishing unsuspected relations between the RT and BDM methods and opened the way for the devising and analysis of new, variable-degree versions of those methods [18].

Soon after, the above approach was extended to produce a unifying framework for the hybridization of mixed, discontinuous and continuous Galerkin methods [22]. All the methods in this framework were such that the only globally coupled unknown was an approximation to the scalar variable on the boundaries of the elements, λ. In fact, when dealing with the RT and BDM methods, the characterization of λ is exactly the one previously obtained in [17], and, when dealing with the CG method, the way of obtaining λ coincides with the traditional static condensation procedure.

In this paper, we take advantage of this unifying framework to compare the CG and a particular HDG method, the so-called LDG-H method; see [22]. In carrying out the comparison, we will attempt to balance the effects of the different advantages of these methods. The first issue to take into account in this balance is that for a given polynomial degree and a given mesh, the unknown λ for the CG method has less degrees of freedom than the unknown λ for the HDG method. The competing second issue is that the corresponding global stiffness matrix of the CG method contains a higher degree of information coupling due to interactions between the vertex degrees of freedom not present in the HDG method, making it possibly less amenable to optimization than its HDG counterpart. The third competing issue is that both the approximations to the scalar variable and its gradient given by the CG method converge with one order less than the locally post-processed approximation to the scalar variable and the approximate gradient provided by the HDG method for simplices; see [19, 23, 26]. (This is unlike all the DG methods considered in the unified analysis of DG methods [4], which provide approximations converging with the *same* orders than the CG method.) A typically cited disadvantage of the CG method, namely, that it does not provide an approximation of the gradient whose normal component is single valued on the boundaries of the elements, can now be overcome [24] by using an L^2-like global projection. It is hence not considered here as a competing issue.

To the best knowledge of the authors, the only comparison of the CG and HDG methods was carried out in [21], for a convection-diffusion-reaction model problem in a diffusion-dominated regime. This paper however only considers up to third-order polynomial expansions and considers counting non-zero matrix entries as a reflection of computation time rather than any direct timing comparisons.

To end, let us point out that if the stabilization function of the h-version of the HDG method are taken to be of order one, the method converges with optimal convergence orders when using simplicial elements. However, if the stabilization function is taken to be of order $1/h$, the method looses one order of convergence in both the locally post-processed approximation to the scalar variable and the approximate gradient; see [23, 26]. A similar effect is produced when instead of taking λ in the space of discontinuous functions, it is taken in a subspace of continuous functions; see [25]. The resulting methods, introduced in [45], were called the embedded discontinuous Galerkin (EDG) methods, and have a stiffness matrix for λ with a sparsity structure identical to that of the statically condensed CG method. A similar idea prompted the introduction of the so-called multi-scale discontinuous Galerkin (MDG) method [10, 36]. The EDH and MDG methods coincide in many cases.

1.2 Outline

The paper is organized as follows. In Sect. 2 we present the CG method and outline the process of static condensation. We then use this as the model to mimic for the presentation of the DG methods and their hybridization. In Sect. 3, the discretization provided by the resulting HDG method is expressed in terms of a matrix representation which is more amenable to numerical implementation. Once the elemental building blocks are in place, we present in Sect. 4 the approach which is used to assemble the linear system which one solves for the trace-space degrees of freedom. In Sect. 5 we then proceed to discuss the patterns and algorithms that one might employ when trying to optimize the construction and assembly of the HDG system. In Sect. 6 we present a comparison of our HDG implementation against a CG implementation within the same code base, Nektar++. We then conclude in Sect. 7 by discussing the pros and cons of the HDG approach as revealed through this implementation and comparison study.

2 Formulation of the Building Blocks

In this section we introduce the CG and HDG methods for the following elliptic diffusion problems with mixed Dirichlet and Neumann boundary conditions:

$$-\nabla^2 u(\boldsymbol{x}) = f(\boldsymbol{x}) \quad \boldsymbol{x} \in \Omega, \tag{1a}$$

$$u(\boldsymbol{x}) = g_D(\boldsymbol{x}) \quad \boldsymbol{x} \in \partial\Omega_D, \tag{1b}$$

$$\boldsymbol{n} \cdot \nabla u(\boldsymbol{x}) = g_N(\boldsymbol{x}) \quad \boldsymbol{x} \in \partial\Omega_N, \tag{1c}$$

where $\partial\Omega_D \cup \partial\Omega_N = \partial\Omega$ and $\partial\Omega_D \cap \partial\Omega_N = \emptyset$. The formulation above can be generalized in many ways which can be treated in a similar manner. For example, by considering a diffusion tensor which is given by a symmetric positive definite matrix and by adding convection and reaction terms.

In order to stress the similarities between the HDG methods and the well-known CG method, we proceed as follows. In Sect. 2.3 we first highlight how to restructure the standard weak global formulation of the CG method to a weak formulation that captures the static condensation procedure. In doing so we introduce a set of *local problems* needed to express the degrees of freedom inside the element in terms of the degrees of freedom on the border of the element, and then, we provide the *global* formulation used to determine the degrees of freedom on the border of the elements.

In Sect. 2.4 we then *mimic* this procedure to *define* the HDG methods. In other words, starting from the original global weak formulation of the HDG methods, we show how to define a static condensation procedure. Thus, we consider a set of *local problems* to express the approximation inside each element in terms of the approximation at its border, and then we provide a *global* formulation with which we determine the approximation on the border of the elements.

From this perspective, both the CG and HDG methods can be viewed as following the same pipeline: construction of a collection of elemental (local) operators which are then judiciously assembled through a static-condensation-aware procedure to yield a global system whose solution determines the degrees of freedom on the boundary of the elements. This boundary system is significantly smaller than the full system one would solve without employing the Schur-complement (the linear algebra underpinning of the static-condensation

procedure) of the corresponding assembled global system. Once the solution has been obtained on the boundaries of the elements, the primary solution over each element can be determined independently through a forward-application of the elemental operators. However before proceeding we first define the partitioning of the domain in Sect. 2.1 and the finite element spaces in Sect. 2.2.

2.1 Partitioning of the Domain

We begin by discretizing our domain. We assume $\mathcal{T}(\Omega)$ is a two-dimensional tessellation of Ω. (Note that although we will describe things in terms of two-dimensions for the purposes of this paper, all mathematical formulations trivially extend to three dimensions.) Let $\Omega^e \in \mathcal{T}(\Omega)$ be a non-overlapping element within the tessellation such that if $e_1 \neq e_2$ then $\Omega^{e_1} \cap \Omega^{e_2} = \emptyset$. By N_{el}, we denote the number of elements (or cardinality) of $\mathcal{T}(\Omega)$. Let $\partial\Omega^e$ denote the boundary of the element Ω^e (i.e. $\bar{\Omega}^e \setminus \Omega^e$) and $\partial\Omega^e_i$ denote an individual edge of $\partial\Omega^e$ such that $1 \leq i \leq N^e_b$ where N^e_b denotes the number of edges of element e. We then denote by Γ the set of boundaries $\partial\Omega^e$ of all the elements Ω^e of $\mathcal{T}(\Omega)$. Finally, we denote by N_Γ the number of edges (or cardinality) of Γ.

For simplicity, we assume that the tessellation $\mathcal{T}(\Omega)$ consists of conforming elements. We say that Γ^l is an *interior edge* of the tessellation $\mathcal{T}(\Omega)$ if there are two elements of the tessellation, Ω^e and Ω^f, such that $\Gamma^l = \Omega^e \cap \Omega^f$ and the length of Γ^l is not zero. We say that Γ^l is a *boundary edge* of the tessellation $\mathcal{T}(\Omega)$ if there is an element of the tessellation, Ω^e, such that $\Gamma^l = \Omega^e \cap \partial\Omega$ and the length of Γ^l is not zero. In Fig. 1, we present a diagram of the domain and elemental tessellation demonstrating the use of our notation.

As it will be useful later, let us define a collection of index mapping functions, graphically depicted in Fig. 2, that allow us to relate the local edges of an element Ω^e, namely, $\partial\Omega^e_1, \ldots, \partial\Omega^e_{N^e_b}$, with the global edges of Γ, that is, with $\Gamma^1, \ldots, \Gamma^{N_\Gamma}$. Thus, since the j-th edge of the element Ω^e, $\partial\Omega^e_j$, is the l-th edge Γ^l of the set of edges Γ, we set $\sigma(e, j) = l$ so that we can write $\partial\Omega^e_j = \Gamma^{\sigma(e,j)}$. Similarly, since the interior edge Γ^l is the intersection of the boundaries of the two elements Ω^e and Ω^f, we set $\eta(l, +) = e$ and $\eta(l, -) = f$ so that we can write $\Gamma^l = \partial\Omega^{\eta(l,+)} \cap \partial\Omega^{\eta(l,-)}$. Here the $\pm$ convention is arbitrary.

Note that everything we are going to do can be extended to non-conforming discretizations; the definition of the set of edges Γ and index mapping functions will have to be modified accordingly. Also note that the definitions above do not preclude the use of more non-traditional element types (e.g. diamond or hexagonal elements).

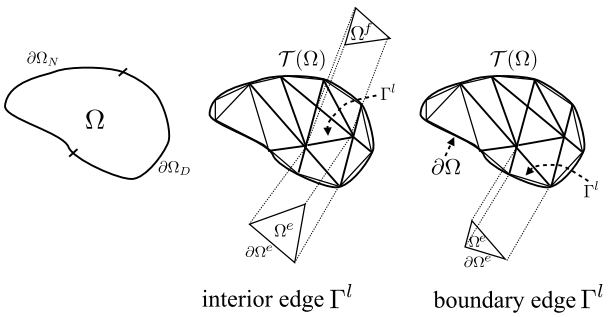

Fig. 1 Diagram of the domain (*left*) and corresponding tessellation (*center* and *right*) demonstrating the use of our notation. Note that the tessellation is a polygonal approximation consisting of straight-sided elements of the original domain

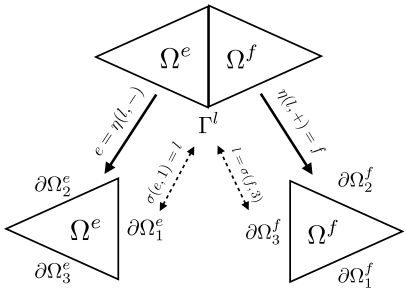

Fig. 2 Diagram demonstrating the use of the index mappings described in the text

2.2 The Finite Element Spaces

Next, we define the finite element spaces associated with the partition $\mathcal{T}(\Omega)$. To begin, for a two-dimensional problem we set

$$V_h := \{v \in L^2(\Omega) : \ v|_{\Omega^e} \in P(\Omega^e)\ \forall \Omega^e \in \mathcal{T}(\Omega)\}, \tag{2a}$$

$$\boldsymbol{\Sigma}_h := \{\tau \in [L^2(\Omega)]^2 : \ \tau|_{\Omega^e} \in \boldsymbol{\Sigma}(\Omega^e)\ \forall \Omega^e \in \mathcal{T}(\Omega)\}, \tag{2b}$$

$$\mathcal{M}_h := \{\mu \in L^2(\Gamma) : \ \mu|_{\Gamma^l} \in P(\Gamma^l)\ \forall \Gamma^l \in \Gamma\}, \tag{2c}$$

where $P(\Gamma^l) = \mathcal{S}_P(\Gamma^l)$ is the polynomial space over the standard segment, $P(\Omega^e) = \mathcal{T}_P(\Omega^e)$ is the space of polynomials of total degree P defined on a standard triangular region and $P(\Omega^e) = \mathcal{Q}_P(\Omega^e)$ is the space of tensor-product polynomials of degree P on a standard quadrilateral region, defined as

$$\mathcal{S}_P(\Gamma^l) = \{s^p;\ 0 \le p \le P;\ (x_1(s), x_2(s)) \in \Gamma^l;\ -1 \le s \le 1\},$$

$$\mathcal{T}_P(\Omega^e) = \{\xi_1^p \xi_2^q;\ 0 \le p+q \le P;\ (x_1(\xi_1,\xi_2), x_2(\xi_1,\xi_2)) \in \Omega^e;\ -1 \le \xi_1+\xi_2 \le 0\},$$

$$\mathcal{Q}_P(\Omega^e) = \{\xi_1^p \xi_2^q;\ 0 \le p,q \le P;\ (x_1(\xi_1,\xi_2), x_2(\xi_1,\xi_2)) \in \Omega^e;\ -1 \le \xi_1,\xi_2 \le 1\}.$$

Similarly $\boldsymbol{\Sigma}(\Omega^e) = [\mathcal{T}_P(\Omega^e)]^2$ or $\boldsymbol{\Sigma}(\Omega^e) = [\mathcal{Q}_P(\Omega^e)]^2$. For curvilinear regions the expansions are only polynomials when mapped to a straight-sided standard region [37, 44].

2.3 The CG Method and Its Static Condensation

It is well known that the approximation u^{CG} given by the CG method is the element of the space V_h^0 of continuous functions in V_h satisfying

$$\begin{aligned} u^{\mathrm{CG}} &= \mathcal{I}_h(g_D) \quad \text{on } \partial\Omega_D, \\ \int_\Omega \nabla v \cdot \nabla u^{\mathrm{CG}} &= \int_{\partial\Omega_N} v g_N + \int_\Omega v f, \end{aligned} \tag{3}$$

for all $v \in V_h^0$ such that $v = 0$ on $\partial\Omega_D$. Here $\mathcal{I}_h$ is a suitably defined interpolation operator whose image is the space of traces on $\partial\Omega_D$ of functions in V_h^0. We next show how to transition from this standard formulation to the formulation that defines the values of u^{CG} on the boundaries of all elements, λ. This is nothing more than the well-known *static condensation* procedure. Although this procedure is traditionally presented in terms of *matrices*, here we also present it in terms of *weak formulations* since in this manner its similarities with the corresponding procedure used in the HDG method will became more clear.

To do this, we begin by noting that, if we assume that the function λ, which belongs to the space $\mathcal{M}_h^0$ of continuous functions in $\mathcal{M}_h$, is known, the equation satisfied by the restriction of u^{CG} to an arbitrary element $\Omega^e \in \mathcal{T}_h$ is the *solution* of the following *local problem*:

$$\begin{aligned} u^{\mathrm{CG}} &= \lambda \quad \text{on } \partial\Omega^e, \\ \int_{\Omega^e} \nabla v \cdot \nabla u^{\mathrm{CG}} &= \int_{\Omega^e} vf \quad \text{for all } v \in P(\Omega^e) \text{ such that } v = 0 \text{ on } \partial\Omega^e. \end{aligned} \tag{4}$$

This formulation follows from the standard global formulation of the CG method by taking the test functions v different from zero only on the element Ω^e. This implies that, if we define by U_λ and by U_f the local solutions to (4) when $f = 0$ and when $\lambda = 0$, respectively (i.e. the homogeneous and heterogeneous solutions), we can write

$$u^{\mathrm{CG}} = U_\lambda + U_f. \tag{5}$$

The discrete problem represented by (4) can also be recast into an elemental matrix problem. To do so, we first define

$$u^{\mathrm{CG}} = \sum_n \phi_n^e(\boldsymbol{x})\hat{\underline{u}}^{\mathrm{CG}}[n] = \sum_n \phi_n^b(\boldsymbol{x})\hat{\underline{u}}^b[n] + \sum_n \phi_n^i(\boldsymbol{x})\hat{\underline{u}}^i[n] \tag{6}$$

where the ϕ_n^i are functions which are defined to be zero on the element boundary $\partial\Omega^e$ and ϕ_n^b are functions that have support on the element boundaries. The array $\hat{\underline{u}}^{\mathrm{CG}}[n]$ holds the degrees of freedom (modes) of the solution; $\hat{\underline{u}}^b[n]$ and $\hat{\underline{u}}^i[n]$ holds the degrees of freedom (modes) on the boundaries and in the interior, respectively. We next introduce

$$\mathbb{L}[n,m] = \int_{\Omega^e} \nabla\phi_n \cdot \nabla\phi_m, \qquad \mathbb{L} = \begin{bmatrix} \mathbb{L}^{b,b} & \mathbb{L}^{b,i} \\ \mathbb{L}^{i,b} & \mathbb{L}^{i,i} \end{bmatrix}, \qquad \underline{f}[n] = \int_{\Omega^e} \phi_n f \tag{7}$$

where the superscripts on the matrix $\mathbb{L}$ correspond to the decomposition of the functions ϕ_n into the sets ϕ_n^b and ϕ_n^i. We can now restate (4) as

$$\hat{\underline{v}}^t \mathbb{L} \hat{\underline{u}} = \hat{\underline{v}}^t \underline{f} \tag{8}$$

where $v = \sum_n \phi_n^e(\boldsymbol{x})\hat{\underline{v}}[n]$. Considering (5), we can express U_λ and U_f in terms of their approximating expansions as follows: $U_\lambda = \sum_n \phi_n(\boldsymbol{x})\hat{\underline{U}}_\lambda[n] = \sum_n \phi_n^b(\boldsymbol{x})\hat{\underline{U}}_\lambda^b[n] + \sum_n \phi_n^i(\boldsymbol{x})\hat{\underline{U}}_\lambda^i[n]$ and $U_f = \sum_n \phi_n(\boldsymbol{x})\underline{U}_f[n] = \sum_n \phi_n^b(\boldsymbol{x})\underline{U}_f^b[n] + \sum_n \phi_n^i(\boldsymbol{x})\underline{U}_f^i[n]$. Let $\lambda = \sum_n \phi_n^b(\partial\Omega^e)\hat{\underline{\lambda}}[n]$. By substituting these expressions into (8) and solving for $\hat{\underline{U}}_\lambda$ assuming $f = 0$ with known boundaries based upon λ, and solving for $\underline{U}_f$ assuming $\lambda = 0$ with known right-hand-side f, we arrive at (respectively):

$$\hat{\underline{U}}_\lambda = \begin{bmatrix} \mathbb{I} \\ -(\mathbb{L}^{i,i})^{-1}\mathbb{L}^{i,b} \end{bmatrix} \hat{\underline{\lambda}}, \qquad \hat{\underline{U}}_f = \begin{bmatrix} 0 & 0 \\ 0 & (\mathbb{L}^{i,i})^{-1} \end{bmatrix} \underline{f}. \tag{9}$$

If we know f and λ (or equivalently $\underline{f}$ and $\hat{\underline{\lambda}}$) we would be able to construct the solution u^{CG}, and so it therefore remains to find a way to characterize λ. It is reasonably straightforward to see that λ is the element of the space $\mathcal{M}_h^0$ such that

$$\begin{aligned} \lambda &= \mathcal{I}_h(g_D) \quad \text{on } \partial\Omega_D, \\ \int_\Omega \nabla U_\mu \cdot \nabla U_\lambda &= \int_\Omega U_\mu f + \int_{\partial\Omega_N} U_\mu g_N \quad \text{for all } \mu \in \mathcal{M}_h^0 \text{ such that } U_\mu = \mu \text{ on } \partial\Omega^e, \end{aligned} \tag{10}$$

where we note that U_λ is related to λ through problem (4) when $f = 0$ and U_μ is similarly related to μ. Indeed, to see that the weak formulation (10) holds, insert the expression of the approximate solution u^{CG} given by (5) into the standard formulation of the CG method given by (3), take the test function v to be U_μ, and note that we have $\int_\Omega \nabla U_f \cdot \nabla U_\mu = 0$, by definition of the local solutions U_f and U_μ. This last result can also be demonstrated by evaluating $\int_\Omega \nabla U_f \cdot \nabla U_\mu = \underline{\hat{U}}_f \mathbb{L} \underline{\hat{U}}_\mu$ using the definitions (7) and (9).

We can further highlight the connection between (10) and the statically condensed CG problem by considering the elemental contribution to (10) using the matrix form we introduced above. We can express the component of problem (10) restricted to element Ω^e as

$$\underline{\hat{\mu}}^t \left[\mathbb{I}, \; -\mathbb{L}^{b,i}(\mathbb{L}^{i,i})^{-1}\right] \begin{bmatrix} \mathbb{L}^{b,b} & \mathbb{L}^{b,i} \\ \mathbb{L}^{i,b} & \mathbb{L}^{i,i} \end{bmatrix} \begin{bmatrix} \mathbb{I} \\ -(\mathbb{L}^{i,i})^{-1}\mathbb{L}^{i,b} \end{bmatrix} \underline{\hat{\lambda}} = \underline{\hat{\mu}}^t \left[\underline{f}^b - \mathbb{L}^{b,i}(\mathbb{L}^{i,i})^{-1}\underline{f}^i\right]$$

where $\mu = \sum_n \phi_n^b(\partial\Omega^e)\underline{\hat{\mu}}$. Multiplying out this equation then leads us to the standard elemental Schur complement formulation of the statically condensed problem for $\underline{\hat{\lambda}}$:

$$\underline{\hat{\mu}}^t \left\{ \left[\mathbb{L}^{b,b} - \mathbb{L}^{b,i}(\mathbb{L}^{i,i})^{-1}\mathbb{L}^{i,b}\right] \underline{\hat{\lambda}} = \underline{f}^b - \mathbb{L}^{b,i}(\mathbb{L}^{i,i})^{-1}\underline{f}^i \right\}. \tag{11}$$

In summary, we have seen that the static condensation procedure for the CG method has three main steps: The introduction of the local problems (4) (or equivalently (8)), the definition of the global approximation in terms of its local solutions (6) (see also (9)), and the global formulation for the approximate trace λ as given by (10) and (11). Next, we show that a similar procedure can be used to define the HDG methods.

2.4 The Global Formulation of the DG Methods

Following [4], we use the so-called flux formulation to define the DG methods. Thus, we rewrite (1a) in auxiliary or mixed form as two first-order differential equations by introducing an auxiliary flux variable $\boldsymbol{q} = \nabla u$. We readily obtain

$$-\nabla \cdot \boldsymbol{q} = f(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Omega, \tag{12a}$$

$$\boldsymbol{q} = \nabla u(\boldsymbol{x}) \qquad \boldsymbol{x} \in \Omega, \tag{12b}$$

$$u(\boldsymbol{x}) = g_D(\boldsymbol{x}) \qquad \boldsymbol{x} \in \partial\Omega_D, \tag{12c}$$

$$\boldsymbol{q} \cdot \boldsymbol{n} = g_N(\boldsymbol{x}) \qquad \boldsymbol{x} \in \partial\Omega_N. \tag{12d}$$

Note that the solution of this first-order system is amenable to approximation by a *mixed method*, that is, by a method using approximations for both the scalar variable u and its gradient $\boldsymbol{q}$. This point is explicitly made in [22] when referring to DG methods as discontinuous Galerkin mixed methods. In the unified analysis of DG methods [4], the mixed form of the DG method is called the *flux* formulation.

Note that in most HDG presentations, e.g., [22, 23, 26], the flux $\boldsymbol{q}$ is defined as the negative gradient of the primitive solution and not as its gradient, as we do here. This was done to stay close to some examples of physical relevance, like heat flow or Darcy's flow, but here we prefer to follow [4] and take $\boldsymbol{q}$ to be the gradient of u.

The DG method seeks an approximation to $(u, \boldsymbol{q})$, $(u^{\mathrm{DG}}, \boldsymbol{q}^{\mathrm{DG}})$, in the space $V_h \times \boldsymbol{\Sigma}_h$, and determines it by requiring that

$$\sum_{\Omega^e \in \mathcal{T}(\Omega)} \int_{\Omega^e} (\nabla v \cdot \boldsymbol{q}^{\mathrm{DG}}) d\boldsymbol{x} - \sum_{\Omega^e \in \mathcal{T}(\Omega)} \int_{\partial\Omega^e} v(\boldsymbol{n}^e \cdot \widetilde{\boldsymbol{q}}^{\mathrm{DG}}) ds = \sum_{\Omega^e \in \mathcal{T}(\Omega)} \int_{\Omega^e} v f d\boldsymbol{x}, \tag{13a}$$

$$\sum_{\Omega^e\in\mathcal{T}(\Omega)}\int_{\Omega^e}(\boldsymbol{w}\cdot\boldsymbol{q}^{\mathrm{DG}})d\boldsymbol{x}=-\sum_{\Omega^e\in\mathcal{T}(\Omega)}\int_{\Omega^e}(\nabla\cdot\boldsymbol{w})u^{\mathrm{DG}}d\boldsymbol{x}+\sum_{\Omega^e\in\mathcal{T}(\Omega)}\int_{\partial\Omega^e}(\boldsymbol{w}\cdot\boldsymbol{n}^e)\widetilde{u}^{\mathrm{DG}}ds, \tag{13b}$$

for all $(v,\boldsymbol{w})\in V_h(\Omega)\times\boldsymbol{\Sigma}_h(\Omega)$, where the numerical traces $\widetilde{u}^{\mathrm{DG}}$ and $\widetilde{\boldsymbol{q}}^{\mathrm{DG}}$ have to be suitably defined in terms of the approximate solution $(u^{\mathrm{DG}},\boldsymbol{q}^{\mathrm{DG}})$. As discussed in [4], many of the specific DG schemes found in the literature can be characterised by appreciating what choices are made for defining $\widetilde{u}^{\mathrm{DG}}$ and $\widetilde{\boldsymbol{q}}^{\mathrm{DG}}$.

Let us briefly discuss the choice with which we are concerned in this paper. First, let us assume that we are at an interface between two elements. Then if we require the numerical traces to be (i) linearly dependent on the available left and right values of the primitive u^{DG} and flux values $\boldsymbol{q}^{\mathrm{DG}}$ on either side of the interface, (ii) *consistent* and (iii) singled valued, we find that the numerical traces *must* be of the form

$$\widetilde{u}=\left(\frac{1}{2}-\boldsymbol{C}_{21}\cdot\boldsymbol{n}^+\right)u^+ +\left(\frac{1}{2}-\boldsymbol{C}_{21}\cdot\boldsymbol{n}^-\right)u^- -C_{22}(\boldsymbol{q}^+\cdot\boldsymbol{n}^+ +\boldsymbol{q}^-\cdot\boldsymbol{n}^-), \tag{14a}$$

$$\widetilde{\boldsymbol{q}}=\left(\frac{1}{2}-\boldsymbol{C}_{12}\cdot\boldsymbol{n}^+\right)\boldsymbol{q}^+ +\left(\frac{1}{2}-\boldsymbol{C}_{12}\cdot\boldsymbol{n}^-\right)\boldsymbol{q}^- -C_{11}(u^+\boldsymbol{n}^+ +u^-\boldsymbol{n}^-), \tag{14b}$$

where C_{11}, C_{22} and $\boldsymbol{C}_{12}$, $\boldsymbol{C}_{21}$ are some scalar and vector parameters to be chosen as to ensure the stability of the method as well as its best possible convergence properties. Note that we have dropped the super-index DG to simplify the notation. On the Dirichlet border $\partial\Omega_D$, we would then have

$$\widetilde{u}=\left(\frac{1}{2}+\boldsymbol{C}_{21}\cdot\boldsymbol{n}\right)u+\left(\frac{1}{2}-\boldsymbol{C}_{21}\cdot\boldsymbol{n}\right)g_D, \tag{15a}$$

$$\widetilde{\boldsymbol{q}}=\boldsymbol{q}-C_{11}(u-g_D)\boldsymbol{n}, \tag{15b}$$

and on the Neumann border $\partial\Omega_N$,

$$\widetilde{u}=u-C_{22}(\boldsymbol{q}\cdot\boldsymbol{n}^+-g_N), \tag{16a}$$

$$\widetilde{\boldsymbol{q}}=\left(\frac{1}{2}-\boldsymbol{C}_{12}\cdot\boldsymbol{n}\right)\boldsymbol{q}+\left(\frac{1}{2}+\boldsymbol{C}_{12}\cdot\boldsymbol{n}\right)g_N\boldsymbol{n}. \tag{16b}$$

Next, we apply a procedure similar to the static condensation procedure for the CG method discussed in Sect. 2.3.

2.5 The HDG Methods and Their Local Problems

We begin by introducing the local problems. Assuming that the function

$$\lambda:=\widetilde{u}^{\mathrm{DG}}\in\mathcal{M}_h, \tag{17a}$$

is known, for any element Ω^e, from the global formulation of the DG method, that the restriction of the DG solution to the element Ω^e, $(u^e,\boldsymbol{q}^e)$ to be the function in $P(\Omega^e)\times\boldsymbol{\Sigma}(\Omega^e)$ satisfies the following equations:

$$\int_{\Omega^e}(\nabla v\cdot\boldsymbol{q}^e)d\boldsymbol{x}-\int_{\partial\Omega^e}v(\boldsymbol{n}^e\cdot\widetilde{\boldsymbol{q}}^e)ds=\int_{\Omega^e}vf\,d\boldsymbol{x}, \tag{17b}$$

$$\int_{\Omega^e} (\boldsymbol{w}\cdot\boldsymbol{q}^e)d\boldsymbol{x} = -\int_{\Omega^e} (\nabla\cdot\boldsymbol{w})u^e d\boldsymbol{x} + \int_{\partial\Omega^e} (\boldsymbol{w}\cdot\boldsymbol{n}^e)\lambda ds, \tag{17c}$$

for all $(v, \boldsymbol{w}) \in P(\Omega^e) \times \boldsymbol{\Sigma}(\Omega^e)$. Note that if we want to be able to solve these equations locally, the numerical trace of the flux *must* depend only on λ and on $(u^e, \boldsymbol{q}^e)$. This property is the *distinctive feature* of the HDG methods. For the example discussed above, this implies that we *must* have

$$\widetilde{\boldsymbol{q}}^e(\boldsymbol{x}) = \boldsymbol{q}^e(\boldsymbol{x}) - \tau(u^e(\boldsymbol{x}) - \lambda(\boldsymbol{x}))\boldsymbol{n}^e \quad \text{on } \partial\Omega^e \tag{17d}$$

for some positive function τ. Because we define the flux as the gradient of the primitive variable (and not as the negative of the gradient of u), we flip the definition of the flux from that given in [22] so that the penalty function τ is still always assumed to be positive so that the existence and uniqueness of the approximate solution is guaranteed. This is so because this local problem is obtained by applying the so-called local discontinuous Galerkin (LDG) method to the single element Ω^e with Dirichlet boundary data λ, as we can see from the formula for the numerical trace given by (15b).

Let us point out that we have tacitly assumed that the DG methods with which we are dealing are those for which their numerical trace $\widetilde{u}^{\mathrm{DG}}$ is single valued. This follows as a trivial consequence of (17a). Also, note that, as it is well known for the LDG method, we can penalise differences between the elemental primitive variable approximation u^e and the trace-space approximation λ by increasing a (possibly spatially changing) parameter τ. For the hybridizable discontinuous Galerkin method taking τ to be positive ensures that the method is well defined. The results in [19, 23, 26] indicate that the best choice is to take τ to be of order one. Note that τ is a function of the set of borders of the elements of the discretization, and so, it is allowed to be different per element and per edge. Thus, if we are dealing with the element whose *global number* is e, we denote the value of τ on the edge whose *local number* is i by $\tau^{e,i}$.

2.6 The Global Formulation for λ

Similar to the CG formulation in Sect. 2.3 we denote by $(U_\lambda, \boldsymbol{Q}_\lambda)$ and $(U_f, \boldsymbol{Q}_f)$ the solution of the (17b) and (17c) local problem when $f = 0$ and when $\lambda = 0$, respectively, and define our approximation to be

$$(u^{\mathrm{HDG}}, \boldsymbol{q}^{\mathrm{HDG}}) = (U_\lambda, \boldsymbol{Q}_\lambda) + (U_f, \boldsymbol{Q}_f).$$

We note that for the HDG decomposition, unlike the CG case, the local problem involve an approximation over $\partial\Omega^e$. However similar to the CG problem solution to (17b) and (17c) when $f = 0$ allows us to express U_λ, $\boldsymbol{Q}_\lambda$ in terms of λ.

It remains to determine λ. To do so, we require that the boundary conditions be weakly satisfied *and* that the normal component of the numerical trace of the flux $\widetilde{\boldsymbol{q}}$ given by (17d) be single valued. This renders this numerical trace *conservative*, a highly valued property for this type of methods; see [4].

So, we say that λ is the element of $\mathcal{M}_h$ such that

$$\lambda = \mathsf{P}_h(g_D) \quad \text{on } \partial\Omega_D, \tag{18a}$$

$$\sum_{\Omega_e\in\mathcal{T}_h} \int_{\partial\Omega^e} \mu\widetilde{\boldsymbol{q}}\cdot\boldsymbol{n} = \int_{\partial\Omega_N} \mu g_N, \tag{18b}$$

for all $\mu \in \mathcal{M}_h^0$ such that $\mu = 0$ on $\partial\Omega_D$. Here P_h denotes the L^2-projection into the space of restrictions to $\partial\Omega_D$ of functions of $\mathcal{M}_h$.

To appreciate what this statement implies with respect to (17a) and (17d), we can examine (18b) on a given interior interface Γ^l. Taking μ equal to zero outside Γ^l, the above equations becomes

$$\int_{\Gamma^l} \mu^l(\boldsymbol{q}^+ \cdot \boldsymbol{n}^+ + \boldsymbol{q}^- \cdot \boldsymbol{n}^-)ds + (\tau^+ + \tau^-)\int_{\Gamma^l} \mu^l \lambda ds - \int_{\Gamma^l} \mu^l(\tau^+ u^+ + \tau^- u^-)ds = 0$$

of the interior interface Γ^l. This implies that on Γ^l we must have

$$\widetilde{u} = \left(\frac{\tau^+}{\tau^- + \tau^+}\right)u_h^+ + \left(\frac{\tau^-}{\tau^- + \tau^+}\right)u_h^- - \left(\frac{1}{\tau^+ + \tau^-}\right)(\boldsymbol{q}^+ \cdot \boldsymbol{n}^+ + \boldsymbol{q}^- \cdot \boldsymbol{n}^-),$$

$$\widetilde{\boldsymbol{q}} = \left(\frac{\tau^-}{\tau^- + \tau^+}\right)\boldsymbol{q}^+ + \left(\frac{\tau^+}{\tau^- + \tau^+}\right)\boldsymbol{q}^- - \left(\frac{\tau^+\tau^-}{\tau^- + \tau^+}\right)(u^+\boldsymbol{n}^+ + u^-\boldsymbol{n}^-).$$

If Γ^l lies on the Neumann boundary, (18b) states that $\widetilde{\boldsymbol{q}} \cdot \boldsymbol{n}$ is nothing but the L^2-projection of the data g_N into $\mathcal{S}_P(\Gamma^l)$.

Finally, note that the weak formulation of (18b) can be rewritten as:

$$\sum_{\Omega_e \in \mathcal{T}_h} \int_{\partial\Omega^e} \mu \widetilde{\boldsymbol{q}}_\lambda \cdot \boldsymbol{n} = -\sum_{\Omega_e \in \mathcal{T}_h} \int_{\partial\Omega^e} \mu \widetilde{\boldsymbol{q}}_f \cdot \boldsymbol{n} + \int_{\partial\Omega_N} \mu g_N, \tag{19}$$

for all $\mu \in \mathcal{M}_h^0$ such that $\mu = 0$ on $\partial\Omega_D$. The fact that the matrix associated to this formulation is symmetric and positive definite was shown in [22].

2.7 Postprocessing

To end the presentation of the HDG methods, we highlight how one can postprocess the approximate solution to obtain a new approximation of the scalar variable converging with an additional order when the elements are simplexes and the polynomial degree P is larger than zero; see [33, 46, 47] for the introduction of this procedure in the context of mixed methods and [19, 23, 26] for its application to HDG methods.

The postprocessing u_h^* on the element Ω^e is the function in $\mathcal{T}_{P+1}(\Omega^e)$ defined by

$$(\nabla u_h^*, \nabla w)_{\Omega^e} = (\boldsymbol{q}_h, \nabla w)_{\Omega^e} \quad \forall \mathcal{T}_{P+1}(\Omega^e), \tag{20a}$$

$$(u_h^*, 1)_{\Omega^e} = (u_h, 1)_{\Omega^e}. \tag{20b}$$

We note that, if $\boldsymbol{q}_h$ converges with order $P+1$ and the average of u_h on each element superconverges with order $P+2$, then the postprocessing u_h^* converges with order $P+2$. It has been shown that, when we use triangles or tetrahedra, $\boldsymbol{q}_h$ converges with order $P+1$ and the average of u_h on each element also superconverges with order $P+2$ when $P \geq 1$.

3 HDG Discrete Matrix Formulation

In this section, to get a better appreciation of the implementation of the HDG approach, we consider the matrix representation of the HDG equations.

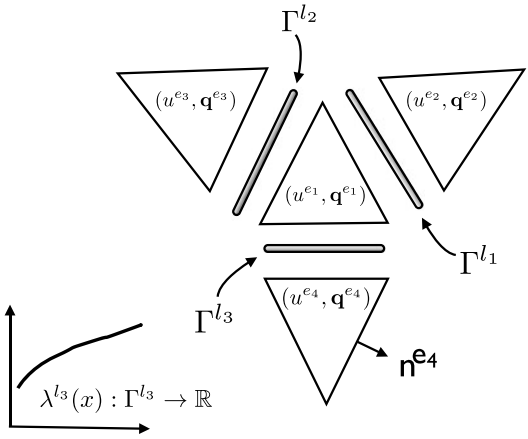

Fig. 3 Diagram showing the relationship between the local solutions and the trace-space variables

3.1 Representation of the Approximation

Consistent with our CG formulation in Sect. 2.3 we start by taking $u^e(\boldsymbol{x})$, $\boldsymbol{q}^e(\boldsymbol{x}) = [q_1, q_2]^T$, and $\lambda^l(\boldsymbol{x})$ to be finite expansions in terms of the basis $\phi_j^e(\boldsymbol{x})$ for the expansions over elements and the basis $\psi_j^l(\boldsymbol{x})$ over the traces of the form:

$$u^e(\boldsymbol{x}) = \sum_{j=1}^{N_u^e} \phi_j^e(\boldsymbol{x}) \underline{\hat{u}}^e[j], \qquad q_k^e(\boldsymbol{x}) = \sum_{j=1}^{N_q^e} \phi_j^e(\boldsymbol{x}) \underline{\hat{q}}_k^e[j], \qquad \lambda^l(\boldsymbol{x}) = \sum_{j=1}^{N_\lambda^l} \psi_j^l(\boldsymbol{x}) \underline{\hat{\lambda}}^l[j].$$

In this notation, it is understood that $u^e(\boldsymbol{x}) : \Omega^e \to \mathbb{R}$, $\mathbf{q}^e(\boldsymbol{x}) : \Omega^e \to \mathbb{R}^2$ and $\lambda^l(\boldsymbol{x}) : \Gamma^l \to \mathbb{R}$ and in Fig. 3 we present a diagram of the elemental tessellation demonstrating the use of our notation.

3.2 Elemental Polynomial Expansion Basis

In our numerical implementation, we have applied a spectral/*hp* element type discretization which is described in detail in [37]. Here we briefly describe the C^0-continuous quadrilateral and triangular expansions within the standard regions which we have adopted in this work. We have chosen this type of basis since, even though we do not wish to enforce C^0 continuity the decomposition of these expansions into an *interior* and *boundary* modes [37, 44], their structure can be useful in the HDG formulation as well.

A commonly used hierarchical C^0 polynomial expansion [37, 43] is based on the tensor product of the integral of Legendre polynomials (or equivalently generalized Jacobi polynomials $P_p^{1,1}(\xi)$) such that

$$\phi_{i(pq)}(x_1(\xi_1, \xi_2), x_2(\xi_1, \xi_2)) = \psi_p(\xi_1)\psi_q(\xi_2) \quad 0 \le p, q \le P$$

where

$$\psi_p(\xi) = \begin{cases} \frac{1-\xi}{2}, & p = 0, \\ \frac{1-\xi}{2}\frac{1+\xi}{2} P_p^{1,1}(\xi), & 0 < p < P, \\ \frac{1+\xi}{2}, & p = P, \end{cases}$$

where $x_1(\xi_1,\xi_2), x_2(\xi_1,\xi_2)$ represent the mapping from the standard region $\Omega^{st} = \{-1 \le \xi_1, \xi_2 \le 1\}$ to Ω^e.

Within a triangular domain a compatible triangular C^0 expansion can also be developed and is based on an orthogonal expansion described by, amongst others, Priorol [40], Koornwinder [39] and Dubiner [31]. This C^0 expansion take the form of a generalised tensor product

$$\phi_{i(pq)}(x_1(\xi_1,\xi_2), x_2(\xi_1,\xi_2)) = \psi_p(\eta_1)\widetilde{\psi}_{pq}(\eta_2)$$

where

$$\widetilde{\psi}_{pq}(z) = \begin{cases} \psi_q(z), & p = 0, 0 \le q \le P, \\ (\frac{1-z}{2})^{p+1}, & 1 \le p \le P, q = 0, \\ (\frac{1-z}{2})^{p+1}(\frac{1+z}{2})P_{q-1}^{2p+1,1}(z), & 1 \le p < P, 1 \le q \le P-q, \\ \psi_q(z), & p = P, 0 \le q \le P, \end{cases}$$

and

$$\eta_1 = 2\frac{(1+\xi_1)}{(1-\xi_2)} - 1, \qquad \eta_2 = \xi_2,$$

and again $x_1(\xi_1,\xi_2), x_2(\xi_1,\xi_2)$ represent a mapping from $\Omega_{st} = \{-1 \le \xi_1 + \xi_2 \le 0\}$ to Ω^e This C^0 expansion was originally proposed by Dubiner [31] and is also detailed in [37, 44].

3.3 Matrix Form of the Equations of the HDG Local Solvers

We can now define the matrix form of the local solvers. Following a standard Galerkin formulation, we set the scalar test functions v^e to be represented by $\phi_i^e(\boldsymbol{x})$ where $i = 1, \ldots, N_u^e$, and let our vector test function $\boldsymbol{w}^e$ be represented by $\boldsymbol{e}_k\phi_i$ where $\boldsymbol{e}_1 = [1,0]^T$ and $\boldsymbol{e}_2 = [0,1]^T$. We next define the following matrices:

$$\mathbb{D}_k^e[i,j] = \left(\phi_i^e, \frac{\partial \phi_j^e}{\partial x_k}\right)_{\Omega^e}, \qquad \mathbb{M}^e[i,j] = \left(\phi_i^e, \phi_j^e\right)_{\Omega^e},$$

$$\mathbb{E}_l^e[i,j] = \left\langle \phi_i^e, \phi_j^e \right\rangle_{\partial\Omega_l^e}, \qquad \widetilde{\mathbb{E}}_{kl}^e[i,j] = \left\langle \phi_i^e, \phi_j^e n_k^e \right\rangle_{\partial\Omega_l^e},$$

$$\mathbb{F}_l^e[i,j] = \left\langle \phi_i^e, \psi_j^{\sigma(e,l)} \right\rangle_{\partial\Omega_l^e}, \qquad \widetilde{\mathbb{F}}_{kl}^e[i,j] = \left\langle \phi_i^e, \psi_j^{\sigma(e,l)} n_k^e \right\rangle_{\partial\Omega_l^e}.$$

We note that we choose the trace expansion to match the expansions used along the edge of the elemental expansion and the local coordinates are aligned, that is $\psi_i^{\sigma(e,l)}(s) = \phi_{k(i)}(s)$ (which is typical of a C^0 expansion basis defined in Sect. 3.2) then $\mathbb{E}_l^e$ contains the same entries as $\mathbb{F}_l^e$ and similarly $\widetilde{\mathbb{E}}_{kl}^e$ contains the same entries as $\widetilde{\mathbb{F}}_{kl}^e$.

After inserting the finite expansion of the trial functions into (17b) and (17c), and using the hybridized definition of the flux given in (17d), the equations for the local solvers can be written in matrix form as:

$$\left[(\mathbb{D}_1^e)^T (\mathbb{D}_2^e)^T\right] \begin{bmatrix} \hat{\underline{q}}_1^e \\ \hat{\underline{q}}_2^e \end{bmatrix} - \sum_{l=1}^{N_b^e} \left[\widetilde{\mathbb{E}}_{1l}^e \widetilde{\mathbb{E}}_{2l}^e\right] \begin{bmatrix} \hat{\underline{q}}_1^e \\ \hat{\underline{q}}_2^e \end{bmatrix} + \sum_{l=1}^{N_b^e} \tau^{e,l} \left[\mathbb{E}_l^e \hat{\underline{u}}^e - \mathbb{F}_l^e \hat{\underline{\lambda}}^{\sigma(e,l)}\right] = \underline{f}^e, \quad (21a)$$

$$\mathbb{M}^e \hat{\underline{q}}_k^e = -(\mathbb{D}_k^e)^T \hat{\underline{u}}^e + \sum_{l=1}^{N_b^e} \widetilde{\mathbb{F}}_{kl}^e \hat{\underline{\lambda}}^{\sigma(e,l)}, \quad k = 0, 1 \quad (21b)$$

where $\underline{f}^e[i] = (\phi_i, f)_{\partial\Omega^e}$.

In the matrix system (21a) the matrix $\mathbb{D}^e$ denotes the elemental weak derivative commonly used in standard Galerkin implementations. Further, the matrix $\mathbb{E}^{e,f}_{kl}$ is a type of mass matrix evaluated on an element edge and projected in the normal component direction n_k. In (21b), $\mathbb{M}^e$ is the element mass matrix also used in standard Galerkin formulations.

3.4 Matrix Form of the Global Equation for λ

Using the matrices from the previous section we can formulate the transmission condition (18b) into a similar matrix description. First we introduced the matrices:

$$\bar{\mathbb{F}}^{l,e}[i,j] = \langle \psi_i^l, \phi_j^e \rangle_{\Gamma^l}, \qquad \widetilde{\bar{\mathbb{F}}}_k^{l,e}[i,j] = \langle \psi_i^l, \phi_j^e n_k^e \rangle_{\Gamma^l}, \qquad \bar{\mathbb{G}}^l[i,j] = \langle \psi_i^l, \psi_j^l \rangle_{\Gamma^l}.$$

Then defining $\underline{g_N}^l[i] = \langle g_N, \psi_i^l \rangle_{\Gamma^l \cap \partial\Omega_N}$, the transmission condition (18b) can be written as:

$$\begin{bmatrix} \widetilde{\bar{\mathbb{F}}}_1^{l,e} \widetilde{\bar{\mathbb{F}}}_2^{l,e} \end{bmatrix} \begin{bmatrix} \hat{\underline{q}}_1^e \\ \hat{\underline{q}}_2^e \end{bmatrix} + \begin{bmatrix} \widetilde{\bar{\mathbb{F}}}_1^{l,f} \widetilde{\bar{\mathbb{F}}}_2^{l,f} \end{bmatrix} \begin{bmatrix} \hat{\underline{q}}_1^f \\ \hat{\underline{q}}_2^f \end{bmatrix} + (\tau^{e,i} + \tau^{f,j}) \bar{\mathbb{G}}^l \hat{\underline{\lambda}}^l - \tau^{e,i} \bar{\mathbb{F}}^{l,e} \underline{u}^e - \tau^{f,j} \bar{\mathbb{F}}^{l,f} \underline{u}^f = \underline{g_N}^l,$$

where we are assuming that $l = \sigma(e,i) = \sigma(f,j)$, that is, that the elements e and f have the common internal edge Γ^l. Finally, by exploiting the following two identities which relate previously defined matrices

$$\mathbb{F}_l^e = (\bar{\mathbb{F}}^{\sigma(e,l),e})^T, \qquad \widetilde{\mathbb{F}}_{kl}^e = (\widetilde{\bar{\mathbb{F}}}_k^{\sigma(e,l),e})^T$$

we obtain that

$$\begin{aligned} &\left\{ \begin{bmatrix} (\widetilde{\mathbb{F}}_1^e)^T & (\widetilde{\mathbb{F}}_2^e)^T \end{bmatrix} \begin{bmatrix} \hat{\underline{q}}_1^e \\ \hat{\underline{q}}_2^e \end{bmatrix} + \tau^{e,i} \bar{\mathbb{G}}^{\sigma(e,i)} \hat{\underline{\lambda}}^{\sigma(e,i)} - \tau^{e,i} (\mathbb{F}_j^e)^T \underline{u}^e \right\} \\ &\quad + \left\{ \begin{bmatrix} (\widetilde{\mathbb{F}}_1^f)^T & (\widetilde{\mathbb{F}}_2^f)^T \end{bmatrix} \begin{bmatrix} \hat{\underline{q}}_1^f \\ \hat{\underline{q}}_2^f \end{bmatrix} + \tau^{f,j} \bar{\mathbb{G}}^{\sigma(f,j)} \hat{\underline{\lambda}}^{\sigma(f,j)} - \tau^{f,j} (\mathbb{F}_j^f)^T \underline{u}^f \right\} = g_N^l. \end{aligned} \tag{22}$$

We see that the transmission condition can be constructed from elemental contributions. In the next section, we show how to use our elemental local solvers given by (21) and (22) to obtain a matrix equation for λ only.

4 Assembling the Transmission Condition from Elemental Contributions

To highlight the process of obtaining the matrix equation for the degrees of freedom of the hybrid variable λ, we begin by rewriting the matrix equations of the HDG method as a global system involving *all* the variables. The trace space equation we seek will essentially then be obtained by calculating the corresponding Schur-complement matrix system for the degrees of freedom for λ.

Let us first re-write (21) and the elemental transmission condition (22) in matrix form. Given a triangular element e, we define the following elemental matrices:

$$\mathbb{A}^e = \begin{pmatrix} \sum_{l=1}^{N_b^e} \tau^{e,l} \mathbb{E}_l^e & (\mathbb{D}_1^e)^T - \sum_{l=1}^{N_b^e} \tilde{\mathbb{E}}_{1l}^e & (\mathbb{D}_2^e)^T - \sum_{l=1}^{N_b^e} \tilde{\mathbb{E}}_{2l}^e \\ (\mathbb{D}_1^e)^T & \mathbb{M}^e & 0 \\ (\mathbb{D}_2^e)^T & 0 & \mathbb{M}^e \end{pmatrix}, \tag{23}$$

$$\mathbb{B}^e = \begin{pmatrix} -\tau^{e,1}(\mathbb{F}_1^e)^T & (\tilde{\mathbb{F}}_{11}^e)^T & (\tilde{\mathbb{F}}_{21}^e)^T \\ -\tau^{e,2}(\mathbb{F}_2^e)^T & (\tilde{\mathbb{F}}_{12}^e)^T & (\tilde{\mathbb{F}}_{22}^e)^T \\ -\tau^{e,3}(\mathbb{F}_3^e)^T & (\tilde{\mathbb{F}}_{13}^e)^T & (\tilde{\mathbb{F}}_{23}^e)^T \end{pmatrix},$$

$$\mathbb{C}^e = \begin{pmatrix} -\tau^{e,1}\mathbb{F}_1^e & -\tau^{e,2}\mathbb{F}_2^e & -\tau^{e,3}\mathbb{F}_3^e \\ -\tilde{\mathbb{F}}_{11}^e & -\tilde{\mathbb{F}}_{12}^e & -\tilde{\mathbb{F}}_{13}^e \\ -\tilde{\mathbb{F}}_{21}^e & -\tilde{\mathbb{F}}_{22}^e & -\tilde{\mathbb{F}}_{23}^e \end{pmatrix}, \tag{24}$$

$$\mathbb{G}^e = \begin{pmatrix} \tau^{e,1}\bar{\mathbb{G}}^{\sigma(e,1)} & 0 & 0 \\ 0 & \tau^{e,2}\bar{\mathbb{G}}^{\sigma(e,2)} & 0 \\ 0 & 0 & \tau^{e,3}\bar{\mathbb{G}}^{\sigma(e,3)} \end{pmatrix}.$$

Quadrilateral elemental matrices are similarly defined with $\mathbb{B}^e$ containing four block rows corresponding to the four edges of a quadrilateral, $\mathbb{C}^e$ containing four block columns and $\mathbb{G}^e$ being of size four by four.

We can now write the local elemental solver (21) in matrix form as:

$$\mathbb{A}^e \begin{bmatrix} \underline{\hat{u}}^e \\ \underline{\hat{q}}_1^e \\ \underline{\hat{q}}_2^e \end{bmatrix} + \mathbb{C}^e \begin{bmatrix} \underline{\hat{\lambda}}^{\sigma(e,1)} \\ \underline{\hat{\lambda}}^{\sigma(e,2)} \\ \underline{\hat{\lambda}}^{\sigma(e,3)} \end{bmatrix} = \begin{bmatrix} \underline{f}^e \\ \underline{0} \\ \underline{0} \end{bmatrix}. \tag{25}$$

We note that each block $\mathbb{A}^e$ is invertible since every local solver involves the DG discretization of an elemental domain with weakly enforced Dirichlet boundary conditions of λ enforced. Therefore each local elemental problem is well-posed and invertible.

Similarly, the transmission condition (22) is given as

$$\mathbb{B}^e \begin{bmatrix} \underline{\hat{u}}^e \\ \underline{\hat{q}}_1^e \\ \underline{\hat{q}}_2^e \end{bmatrix} + \mathbb{G}^e \begin{bmatrix} \underline{\hat{\lambda}}^{\sigma(e,1)} \\ \underline{\hat{\lambda}}^{\sigma(e,2)} \\ \underline{\hat{\lambda}}^{\sigma(e,3)} \end{bmatrix} + \mathbb{B}^f \begin{bmatrix} \underline{\hat{u}}^f \\ \underline{\hat{q}}_1^f \\ \underline{\hat{q}}_2^f \end{bmatrix} + \mathbb{G}^f \begin{bmatrix} \underline{\hat{\lambda}}^{\sigma(f,1)} \\ \underline{\hat{\lambda}}^{\sigma(f,2)} \\ \underline{\hat{\lambda}}^{\sigma(f,3)} \end{bmatrix} = \underline{g}_N^l. \tag{26}$$

The efficiency of the HDG implementation then arises from obtaining an element-wise matrix systems to construct a global system for the trace space degrees of freedom. Let $\underline{\Lambda}^l$ denote the vector of degrees of freedom on the edge Γ^l and let $\underline{\Lambda}$ be the concatenation of these vectors for all the edges of the triangulation. The size of $\underline{\Lambda}$ is therefore

$$N_\lambda = \sum_{l \in \Gamma} N_\lambda^l,$$

where N_λ^l is the number degrees of freedom of λ on the interior edge Γ^l.

We define the trace space spreading operator $\mathcal{A}_{HDG}$ as a matrix of size $2N_\lambda \times N_\lambda$ which "spreads" or scatters the unique trace space values to their local edge vectors. For each element e, which consists of N_b^e edges, let $\underline{\hat{\lambda}}^{e,l}$ denote the local copy of the trace-space information as portrayed in Fig. 4. Notice that for each edge, there will be two elements on either side of that edge.

We can organize the matrix $\mathcal{A}_{HDG}$ by elements such that

$$\mathcal{A}_{HDG} = \begin{pmatrix} \mathcal{A}_{HDG}^{e_1} \\ \vdots \\ \mathcal{A}_{HDG}^{e_{N_{el}}} \end{pmatrix}$$

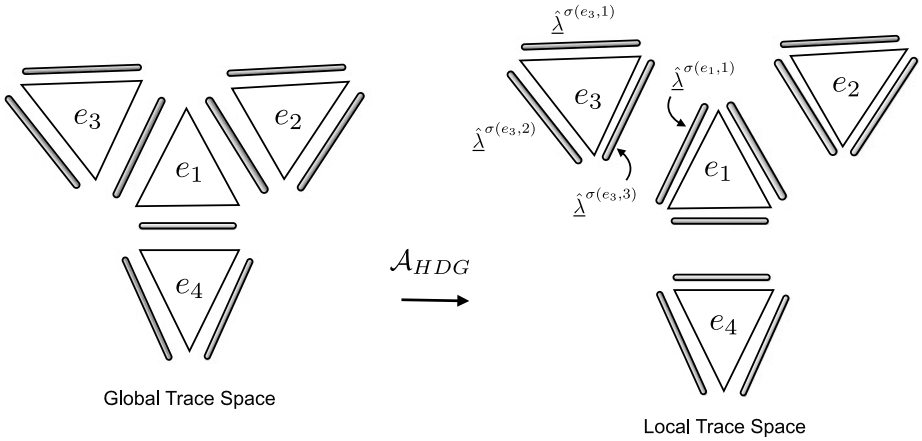

Fig. 4 Diagram showing the results of the spreading operation $\mathcal{A}_{HDG}$. Unique degrees of freedom of λ on an edge are copied to their locally-indexed counterparts

where the action of $\mathcal{A}^e_{HDG}$ is to copy global trace space information into local (elemental) storage. Note that in some cases this is merely a (identical) copy of the information whereas in other cases the copying might involve a permutation of the ordering to accommodate local orientation storage issues. This is analogous to the standard CG global assembly process [37].

With this notation in place, we can rewrite the equations for the local solvers (25) as follows:

$$\mathbb{A}^e \underline{v}^e + \mathbb{C}^e \mathcal{A}^e_{HDG} \underline{\Lambda} = \underline{w}^e \tag{27}$$

where $\underline{v}^e = (\underline{\hat{u}}^e, \underline{\hat{q}}^e_1, \underline{\hat{q}}^e_2)^T$ and $\underline{w}^e = (\underline{f}^e, 0, 0)^T$. We can similarly write the transmission conditions (26) between interfaces as:

$$\sum_{e=1}^{|\mathcal{T}(\Omega)|} (\mathcal{A}^e_{HDG})^T \left[\mathbb{B}^e \underline{v}^e + \mathbb{G}^e \mathcal{A}^e_{HDG} \underline{\Lambda}\right] = \underline{g}_N \tag{28}$$

where the sum over elements along with the left application of the transpose of the spreading operator acts to "assemble" (sum up) the elemental contributions corresponding to each trace space edge and where $\underline{g}_N$ denotes the concatenation of the individual edge Neumann conditions $\underline{g}^l_N$.

Manipulating (27) to solve for $\underline{v}^e$ and inserting it into (28) yields:

$$\sum_{e=1}^{|\mathcal{T}(\Omega)|} (\mathcal{A}^e_{HDG})^T \left[\mathbb{B}^e (\mathbb{A}^e)^{-1} \left(\underline{w} - \mathbb{C}^e \mathcal{A}^e_{HDG} \underline{\Lambda}\right) + \mathbb{G}^e \mathcal{A}^e_{HDG} \underline{\Lambda}\right] = \underline{g}_N$$

which can be reorganized to arrive at matrix equation for λ:

$$\mathbf{K}\underline{\Lambda} = \underline{F}, \tag{29}$$

where

$$\mathbf{K} = \sum_{e=1}^{|\mathcal{T}(\Omega)|} (\mathcal{A}^e_{HDG})^T \mathbb{K}^e \mathcal{A}^e_{HDG} = \sum_{e=1}^{|\mathcal{T}(\Omega)|} (\mathcal{A}^e_{HDG})^T \left[\mathbb{G}^e - \mathbb{B}^e (\mathbb{A}^e)^{-1} \mathbb{C}^e\right] \mathcal{A}^e_{HDG}$$

and

$$\underline{F} = \underline{g}_N - \sum_{e=1}^{|\mathcal{T}(\Omega)|} (\mathcal{A}^e_{HDG})^T \mathbb{B}^e (\mathbb{A}^e)^{-1} \underline{w}. \tag{30}$$

We observe that $\mathbf{K}$ is constructed elementally through the sub-matrices $\mathbb{K}^e$ which can also be considered as the Schur complement of a larger matrix system which consists of combining (27) and (28).

5 HDG Implementation Considerations

We now summarize which discrete matrix systems are required to implement the HDG scheme to obtain a solution to the primitive variable $\hat{\underline{u}}^e$. Details of the CG formulation, which follow a similar construction, can be found in [37].

We recall that the solution is given by first constructing and solving (29) to obtain the trace space solution, $\underline{\Lambda}$, which can then be used at an element level to solve (27). We note that in both these steps we require the inversion of the elemental matrix $\mathbb{A}^e$.

We therefore start our implementation discussion by noting that assuming exact numerical integration then it can be shown that:

$$\mathbb{D}^e_k = -\left[(\mathbb{D}^e_k)^T - \sum_{l=1}^{N^e_b} \tilde{\mathbb{E}}^e_{kl} \right] \tag{31}$$

which is discretely equivalent to the identity

$$\int_{\Omega^e} \phi_i \frac{\partial}{\partial x_k} \phi_j d\boldsymbol{x} = -\int_{\Omega^e} \frac{\partial}{\partial x_k} \phi_i \phi_j d\boldsymbol{x} + \int_{\partial\Omega^e} \phi_i \phi_j \boldsymbol{n}^e_k ds.$$

Using (31) in the definition of $\mathbb{A}^e$ given by (23) we can alternatively define this matrix as

$$\mathbb{A}^e = \begin{pmatrix} \sum_{l=1}^{N^e_b} \boldsymbol{\tau}^{(e,l)} \mathbb{E}^e_l & -\mathbb{D}^e_1 & -\mathbb{D}^e_2 \\ (\mathbb{D}^e_1)^T & \mathbb{M}^e & 0 \\ (\mathbb{D}^e_2)^T & 0 & \mathbb{M}^e \end{pmatrix}. \tag{32}$$

This form of the matrix $\mathbb{A}^e$ highlights the near symmetry of the operator, indeed if we define

$$\tilde{\mathbb{I}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix}$$

we then observe that $\widetilde{\mathbb{A}}^e = \tilde{\mathbb{I}}\mathbb{A}^e$ is symmetric.

Remark 1 The discrete relation (31) is only exact if the numerical quadrature is exact. Whilst this is often the case for straight-sided elements it is typically not the cases for curvilinear

elements or even straight-sided quadrilateral element that are non parallelograms. In these cases the reformulation of $\mathbb{A}^e$ given by (32) ensure the final discrete problem remains symmetric.

It can be shown that the elemental matrix $(\mathbb{A}^e)^{-1}$ can be expressed as:

$$\begin{pmatrix} \mathbb{Z}^e & \mathbb{Z}^e\mathbb{D}_1^e(\mathbb{M}^e)^{-1} & \mathbb{Z}^e\mathbb{D}_2^e(\mathbb{M}^e)^{-1} \\ -(\mathbb{M}^e)^{-1}(\mathbb{D}_1^e)^T\mathbb{Z}^e & (\mathbb{M}^e)^{-1}[\mathbb{I}-(\mathbb{D}_1^e)^T\mathbb{Z}^e\mathbb{D}_1^e(\mathbb{M}^e)^{-1}] & -(\mathbb{M}^e)^{-1}(\mathbb{D}_1^e)^T\mathbb{Z}^e\mathbb{D}_2^e(\mathbb{M}^e)^{-1} \\ -(\mathbb{M}^e)^{-1}(\mathbb{D}_2^e)^T\mathbb{Z}^e & -(\mathbb{M}^e)^{-1}(\mathbb{D}_2^e)^T\mathbb{Z}^e\mathbb{D}_1^e(\mathbb{M}^e)^{-1} & (\mathbb{M}^e)^{-1}[\mathbb{I}-(\mathbb{D}_2^e)^T\mathbb{Z}^e\mathbb{D}_2^e(\mathbb{M}^e)^{-1}] \end{pmatrix} \tag{33}$$

where

$$\mathbb{Z}^e = \left(\sum_{l=1}^{N_b^e} \tau^{(e,l)}\mathbb{E}_l^e + \mathbb{D}_1^e(\mathbb{M}^e)^{-1}(\mathbb{D}_1^e)^T + \mathbb{D}_2^e(\mathbb{M}^e)^{-1}(\mathbb{D}_2^e)^T \right)^{-1} \tag{34}$$

and we have explicitly used the fact that $\mathbb{M}^e = (\mathbb{M}^e)^T$ and $\mathbb{Z}^e = (\mathbb{Z}^e)^T$. Interestingly at first glance the matrix $(\mathbb{Z}^e)^{-1}$ with $\tau = 0$ appears to be similar to the elemental weak Laplacian, $\mathbb{L}^e$ (see (7)), however in the current notation the elemental weak Laplacian is expressed by:

$$\mathbb{L}^e = (\mathbb{D}_1^e)^T(\mathbb{M}^e)^{-1}\mathbb{D}_1^e + (\mathbb{D}_2^e)^T(\mathbb{M}^e)^{-1}\mathbb{D}_2^e$$

where we note that $\mathbb{D}_k$ and $(\mathbb{M}^e)^{-1}$ are in a reverse permutation when considering $\mathbb{L}^e$ as opposed to $(\mathbb{Z}^e)^{-1}$.

The implementation and solution of the HDG scheme can now be summarised as:

1. Construct $\mathbb{Z}^e$ given by (34) which requires the elemental matrices $(\mathbb{M}^e)^{-1}$ and $\mathbb{D}_k$. The action of $\mathbb{D}_k$ and $\mathbb{E}_l$ can be implemented in a matrix-free manner.
2. Construct

$$\mathbb{U}^e = -[\mathbb{I}\ 0\ 0](\mathbb{A}^e)^{-1}\mathbb{C}^e = -\mathbb{Z}^e[\mathbb{I}\ \mathbb{D}_1^e(\mathbb{M}^e)^{-1}\ \mathbb{D}_2^e(\mathbb{M}^e)^{-1}]\mathbb{C}^e$$

   which requires the additional matrix operation $\mathbb{C}^e$ which can also be implemented in a matrix-free manner if desired. Also note that the action of $(\mathbb{A}^e)^{-1}$ can be evaluated using (33) and so does not need to be directly constructed.
3. Construct $\mathbb{K}^e = [\mathbb{G}^e - \mathbb{B}^e(\mathbb{A}^e)^{-1}\mathbb{C}^e]$. We note that constructing $\mathbb{K}^e$ requires the action of $(\mathbb{A}^e)^{-1}\mathbb{C}^e$ which is essentially the action of $\mathbb{U}^e$ combined with

$$\mathbb{Q}_0^e = -[0\ \mathbb{I}\ 0](\mathbb{A}^e)^{-1}\mathbb{C}^e, \qquad \mathbb{Q}_1^e = -[0\ 0\ \mathbb{I}](\mathbb{A}^e)^{-1}\mathbb{C}^e$$

   such that

$$\mathbb{K}^e = \mathbb{G}^e - \mathbb{B}^e \begin{bmatrix} \mathbb{U}^e \\ \mathbb{Q}_0^e \\ \mathbb{Q}_1^e \end{bmatrix}.$$

   The matrices $\mathbb{Q}_0^e, \mathbb{Q}_1^e$ require the explicit evaluation of all terms in the second and third rows of definition (33) respectively.
4. From the elemental matrices $\mathbb{K}^e$ we can construct and invert the global matrix $\mathbf{K}$ using the assembly process discussed in Sect. 4

5. We next determine the trace space solution $\underline{\Lambda} = \mathbf{K}^{-1}\underline{F}$ where, as we shall demonstrate below, $\underline{F}$ can be evaluated using $\mathbb{U}^e$ as

$$\underline{F} = \underline{g}_N + \sum_{e=1}^{|\mathcal{T}(\Omega)|} (\mathcal{A}^e_{HDG})^T (\mathbb{U}^e)^T \underline{f}^e.$$

6. Finally recovering the elemental construction of trace solution $\underline{\lambda}^e = \mathcal{A}^e_{HDG}\underline{\Lambda}$ we obtain the elemental primitive solution $\underline{\hat{u}}^e$ from (25) as

$$\underline{\hat{u}}^e = \mathbb{Z}^e \underline{f}^e + \mathbb{U}^e \underline{\lambda}^e.$$

Although a number of elemental matrices, $(\mathbb{M}^e)^{-1}$, $\mathbb{D}^e_k$, $\mathbb{U}^e$, $\mathbb{Q}^e_0$, $\mathbb{Q}^e_1$ and $\mathbb{K}^e$, is required in the construction and solution of the HDG problem, the repeated application of the same inverse, as is typically required in solution of time dependent parabolic PDE problems, requires the use of only the matrices $\mathbf{K}^{-1}$, $\mathbb{U}^e$, $\mathbb{Z}^e$. These matrices have a direct analog in a CG discretization when hybridisation/static condensation is applied, although the rank of each matrix differs to that of the HDG problem.

It remains for us to demonstrate the relationship between $[\mathbb{B}^e(\mathbb{A}^e)^{-1}]$ and $[(\mathbb{A}^e)^{-1}\mathbb{C}^e]^T$ that we have used in evaluating $\underline{F}$ in step (5). First we note from inspection of the matrices defined in (24) that $\mathbb{B}^e = (\mathbb{C}^e)^T\widetilde{\mathbb{I}}$ we therefore observe that the action of $[\mathbb{B}^e(\mathbb{A}^e)^{-1}]\underline{w}$ in (30) can be recast as

$$\begin{aligned}
[\mathbb{B}^e(\mathbb{A}^e)^{-1}]\begin{pmatrix}\underline{f}^e\\ \underline{0}\\ 0\end{pmatrix} &= \mathbb{B}^e\begin{pmatrix}\mathbb{Z}^e\underline{f}^e\\ -(\mathbb{M}^e)^{-1}(\mathbb{D}^e_1)\mathbb{Z}^e\underline{f}^e\\ -(\mathbb{M}^e)^{-1}(\mathbb{D}^e_2)\mathbb{Z}^e\underline{f}^e\end{pmatrix}\\
&= (\mathbb{C}^e)^T\widetilde{I}\begin{pmatrix}\mathbb{Z}^e\underline{f}^e\\ -(\mathbb{M}^e)^{-1}(\mathbb{D}^e_1)\mathbb{Z}^e\underline{f}^e\\ -(\mathbb{M}^e)^{-1}(\mathbb{D}^e_2)\mathbb{Z}^e\underline{f}^e\end{pmatrix} = (\mathbb{C}^e)^T\begin{pmatrix}\mathbb{Z}^e\underline{f}^e\\ (\mathbb{M}^e)^{-1}(\mathbb{D}^e_1)\mathbb{Z}^e\underline{f}^e\\ (\mathbb{M}^e)^{-1}(\mathbb{D}^e_2)\mathbb{Z}^e\underline{f}^e\end{pmatrix}\\
&= [(\mathbb{A}^e)^{-1}\mathbb{C}^e]^T\begin{pmatrix}\underline{f}^e\\ \underline{0}\\ 0\end{pmatrix} = (\mathbb{U}^e)^T\underline{f}^e.
\end{aligned}$$

Finally we can also demonstrate that $\mathbb{K}^e = \mathbb{G}^e - \mathbb{B}^e(\mathbb{A}^e)^{-1}\mathbb{C}^e$ is symmetric. Firstly we note that $\mathbb{G}^e$ is symmetric by definition. We then recall that since $\widetilde{\mathbb{A}}^e = \tilde{\mathbb{I}}\mathbb{A}^e$ is a symmetric matrix and since $\mathbb{C}^e = \tilde{\mathbb{I}}(\mathbb{B}^e)^T$ we observe that

$$\mathbb{B}^e(\mathbb{A}^e)^{-1}\mathbb{C}^e = \mathbb{B}^e(\mathbb{A}^e)^{-1}\tilde{\mathbb{I}}(\mathbb{B}^e)^T = \mathbb{B}^e(\widetilde{\mathbb{A}}^e)^{-1}(\mathbb{B}^e)^T$$

where we have used the fact that $(\widetilde{\mathbb{A}}^e)^{-1} = (\mathbb{A}^e)^{-1}\tilde{\mathbb{I}}$.

6 Results

We consider a square region $0 \le x, y \le 1$ discretized using a variety of regular quadrilateral and unstructured triangular elements. We define the level of mesh refinement by the number of equispaced segments along each side of the domain. In Fig. 5 we show a representative quadrilateral mesh using $15 \times 15 = 225$ elements in the quadrilateral mesh and a representative unstructured triangular mesh using $431 \approx 2 \times 15 \times 15$ elements. We have also

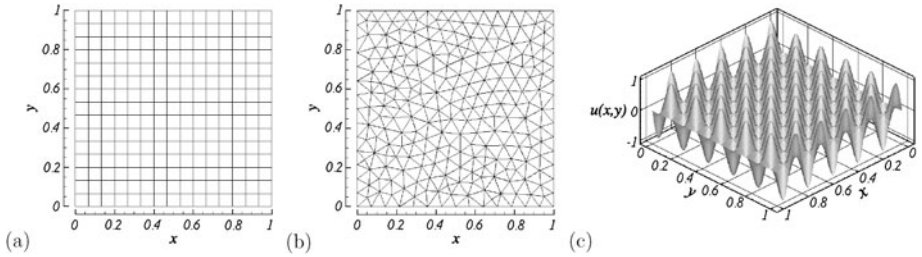

Fig. 5 Discretizations of the unit square where each edge is divided into 15 segments using (**a**) regular quadrilateral regions and (**b**) unstructured triangular regions. A representative solution field $u(x, y) = \sin(10\pi x)\cos(10\pi y)$ is shown in (**c**)

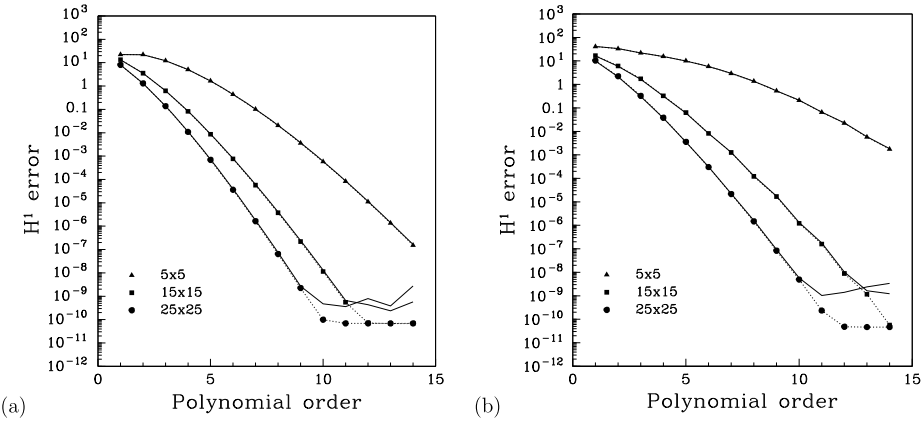

Fig. 6 Comparison of the element-wise H^1 error from using a HDG (*solid lines*) and CG (*symbols with dotted lines*) projections as a function of polynomial order on a (**a**) quadrilateral mesh and (**b**) triangular meshes

considered a coarse mesh where the sides of the domain are separated into just five segments and so we have 25 quadrilateral elements and 38 triangular elements, we will refer to this as the 5×5 mesh. In addition we also consider a finer mesh where each side of the domain is divided into 25 equispaced segments and so we have 625 quadrilateral elements and 1326 triangular elements and we will refer to this as the 25×25 mesh.

As the first illustrative problem we will consider the "elliptic" Helmholtz problem

$$-\nabla^2 u + \lambda u = f$$

where $\lambda = 1$ and $f(x, y)$ is selected to give an exact solution of the form

$$u(x, y) = \sin(10\pi x)\cos(10\pi y) \tag{35}$$

as shown in Fig. 5(c). In all tests we have imposed Dirichlet boundary conditions.

We recall that the HDG method asymptotically tends to the CG method as $\tau \to \infty$ and in our test we therefore first consider the case where $\tau = 1000$. A comparison of element-wise H^1 error as a function of polynomial order is shown in Fig. 6 where we observe that the errors for the CG and HDG implementations on both the quadrilateral and triangular meshes are very close down to a tolerance of 1×10^{-9}.

We next consider the CPU time of the CG and HDG implementations. In evaluating the CPU times we consider the solve, rather than the matrix setup CPU time, which is reasonable for time dependent problems where the matrix does not change every time step. This is typical for example in a velocity-correction scheme [34, 38] to time integrate the incompressible Navier-Stokes equations where the Stokes operator is handled implicitly. This method has been extensively used in the direct numerical solution of canonical problems within essentially two-dimensional problems in for example flow past cylinders [11, 29], backwards facing steps [6] or flow within constricted pipes [5]. These are also examples of where the direct inverse of a series of two-dimensional operators of a similar size to the example shown in Fig. 5 can be used to understand three-dimensional flow physics. It would however not be a suitable comparison for a stationary problem requiring a single solve, such as commonly arise in structural mechanics problems or if a fully implicit time stepping scheme were applied to a non-linear problem.

In gathering our data we have ensured that we made a sufficient number of solve calls so that a minimum of a second of CPU time is used. This is important for low polynomial orders to ensure the sensitivity of the timing routines do not influence the results. In addition we average the data over three separate runs. In this initial set of tests we contrast two solver techniques. The first one using a banded Cholesky factorization using the LAPACK [1] routine *dpbtrs* where the bandwidth of the boundary system was minimized using a reverse Cuthill-McGee algorithm implemented within the Boost library [7]. The second solver uses a multi-level static condensation/substructuring technique [37] which uses the compact nature of the spectral/hp element discretisation to identify grouping of degrees of freedom that can be recursively inverted and in concept is similar to a bisection method for inverting the matrix.

All tests were performed on a Mac Pro with two dual core 2.66 GHz processor, 4 Gb RAM and 4 MB L2 Cache per processor. Finally we have implemented both the CG and HDG solvers to use elemental/local matrix routines where possible. We note however that it is possible to have an increased performance in the CG discretization at lower orders using a global matrix operations techniques [49] for further details.

Figure 7 shows the comparison of the HDG over the CG CPU times expressed as a percentage for both the 15×15 and the 25×25 quadrilateral and triangular meshes. We observe that for first-order polynomial expansions based on both the quadrilateral and triangular meshes the HDG solver is significantly (2–2.5 times) more expensive than the CG implementation independent of the solver strategy. When using the Cholesky solver in the case of a quadrilateral mesh the HDG solve becomes faster than the CG implementation at a polynomial order $4 < P < 5$ where as the same cross over point arises between $5 < P < 6$ in the triangular mesh. For higher polynomial orders the HDG solver asymptotes to between 90–95% of the CG CPU time on the 15×15 mesh and between 85–90% faster on the 25×25 mesh. However when using the multilevel static condensation methods, which is typically faster than the Cholesky approach, the HDG method is always more expensive than the CG method but is within 20% of the method by a polynomial order of $4 < P < 5$ for quadrilateral elements and $5 < P < 6$ for triangular elements.

To gain a better understanding of how the HDG efficiency arises due to the Choleskly solve using a minimum bandwidth ordering we consider the matrix data shown in Tables 1 and 2. These tables provide data on the rank and upper bandwidth of the boundary/trace system for the quadrilateral and triangular meshes respectively. We observe from the data in the second to fourth columns that the HDG problem always leads to a larger rank system which is to be expected form the construction of the two schemes since the vertex degrees of freedom are independent in the HDG method. For first-order polynomials the CG system contains approximately half the degrees of freedom of the HDG problem for the same

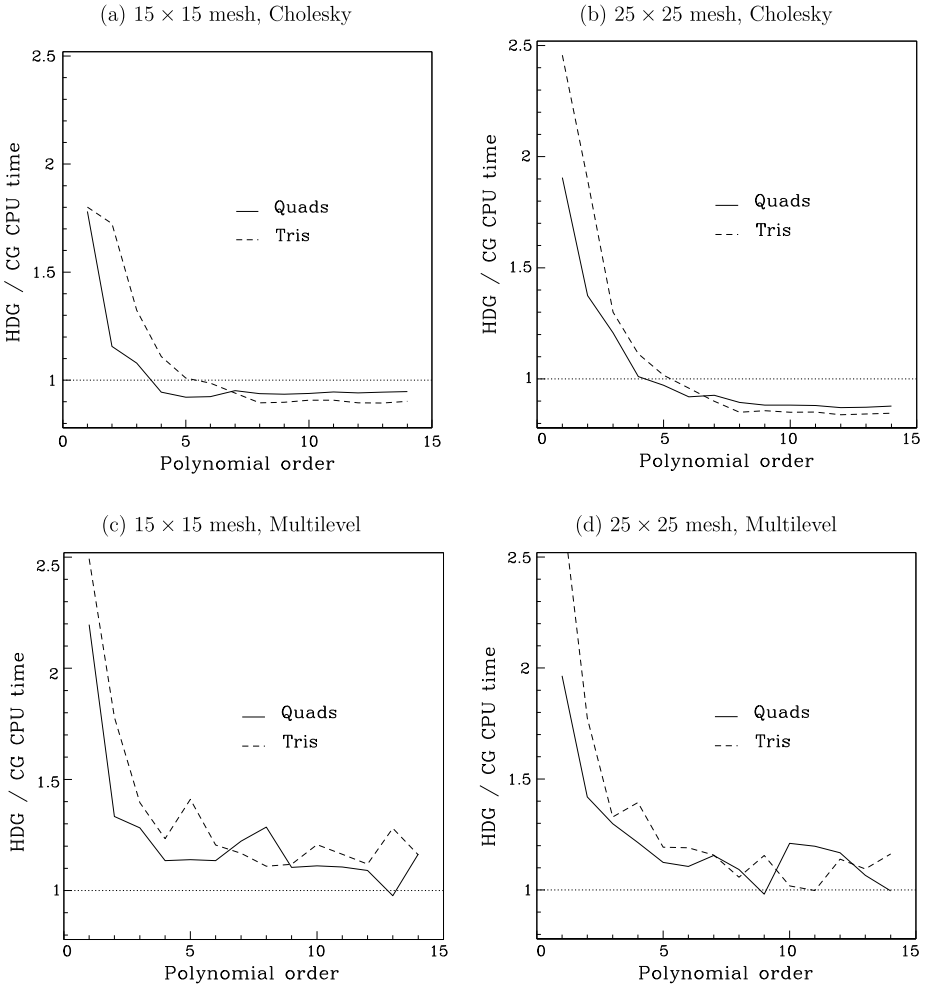

Fig. 7 Relative CPU time of the HDG versus CG solve times as a function of polynomial order for the quadrilateral mesh (*solid line*) and the triangular mesh (*dashed line*) on a 15 × 15 mesh (*left*) and 25 × 25 mesh (*right*). *The top figures* used a RCM-Cholesky algorithm whereas in *the bottom figures* a multi-level static condensation technique was applied. *The dotted line* indicates equivalent speed

reason. We note that rank of the systems is $N_\lambda \times (P+1)$ for HDG and $N_\lambda P$ for CG. In the case of a periodic domain with a regular pattern of elements we can approximate N_λ as $N_\lambda \approx 2N_{el}$ for quadrilaterals and $N_\lambda = 3N_{el}/2$ for triangles. Therefore we can argue that the rank is approximately $2N_{el}(P+1)$ or $2N_{el}P$ for the HDG and CG in a quadrilateral domain and the rank is approximately $3/2N_{el}(P+1)$ or $3/2N_{el}P$ for HDG or CG in a triangular domain.

The fifth and sixth columns of Tables 1 and 2 indicated the upper bandwidth of the boundary/trace matrix. In the subsequent two columns we also indicate the ratio of the bandwidth to rank for both the CG and HDG boundary/trace matrices. From this data it is clear that the weaker coupling of the HDG matrix leads to a boundary system which has an upper bandwidth which is approximately half the size of the CG matrix at the same polynomial order.

Table 1 Table of boundary matrix system for the 15 × 15 quadrilateral mesh

Poly. order	Rank			Bandwidth		Bandwidth/Rank		(HDG-CG)/CG matrix entries %
	CG	HDG	%	CG	HDG	CG	HDG	
2	616	1260	205	81	92	0.131	0.073	128
4	1456	2100	144	191	154	0.131	0.073	17
6	2296	2940	128	301	216	0.131	0.073	−7
8	3136	3780	121	411	278	0.131	0.074	−17
10	3976	4620	116	521	340	0.131	0.074	−23
12	4816	5460	113	631	402	0.131	0.074	−26
14	5656	6300	111	741	464	0.131	0.074	−29

Table 2 Table of boundary matrix system for the 15 × 15 triangular mesh

Poly. order	Rank			Bandwidth		Bandwidth/Rank		(HDG-CG)/CG matrix entries %
	CG	HDG	%	CG	HDG	CG	HDG	
2	753	1860	247	105	119	0.139	0.064	157
4	1993	3100	156	266	199	0.133	0.064	15
6	3233	4340	134	428	279	0.132	0.064	−12
8	4473	5580	125	590	359	0.132	0.064	−23
10	5713	6820	119	752	439	0.132	0.064	−29
12	6953	8060	116	914	519	0.131	0.064	−33
14	8193	9300	114	1076	599	0.131	0.064	−36

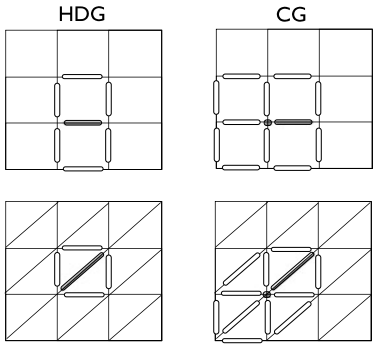

Fig. 8 Schematic of coupling of an edge in HDG and edge plus a vertex for CG with adjacent edges. The coupled degrees of freedom are identified by a *shaded long oval* and the edges that are connected to this edge are identified by *white ovals*

To appreciate how the factor arises we consider the coupling of an edge within the trace for the HDG scheme as schematically shown in Fig. 8. In the left hand plots of this figure we observe that an edge is coupled to either six or four other edges in the HDG quadrilateral and triangular meshes, respectively. This would imply a minimum upper bandwidth of $4(P+1)$ for the HDG discretization quadrilateral domains and $3(P+1)$ for triangular domains. The right hand plots of Fig. 8 show a schematic of the edge coupling of the CG schemes where we consider the expansions related to an edge to also include a single vertex. We observed that in the CG case the central data is coupled to 11 other edges in both the quadrilateral and triangular meshes. This implies that a minimum upper bandwidth is approximately $7(P+1)$

and $7P$ for HDG and CG methods, respectively. Using these approximation and the scaling of the rank for periodic meshes we obtain the following factor between the CG and HDG bandwidth over rank values on quadrilateral and triangular meshes:

$$\left[\frac{\text{Bwidth/Rank}_{CG}}{\text{Bwidth/Rank}_{HDG}}\right]_{quad} = \frac{7P/(2N_{el}P)}{4(P+1)/(2N_{el}(P+1))} = \frac{7}{4},$$

$$\left[\frac{\text{Bwidth/Rank}_{CG}}{\text{Bwidth/Rank}_{HDG}}\right]_{tri} = \frac{7P/(3/2N_{el}P)}{3(P+1)/(3/2N_{el}(P+1))} = \frac{7}{3}.$$

These estimates are consistent with the factor of two observed in the numerical results.

In the final column of these Tables 1 and 2 we show a theoretical estimate of the percentage additional cost of performing an HDG versus a CG solve as represented by the number of entries in the factored matrices. If we have a Cholesky factored matrix of rank R and upper bandwidth B the solve routine requires two back-substitutes of size $Slv = R.B - B(B+1)/2$. The data in the last column therefore contains the ratio $(Slv_{CG} - Slv_{HDG})/Slv_{CG}$ expressed as a percentage. This ratio highlights that the boundary/trace solve for first-order polynomials is more than twice as costly for the HDG scheme as compared to the CG scheme. However for polynomial orders $P > 12$ we observe that the HDG banded Cholesky solve requires up to 30% less operations than the CG solve. We also note that for both the quadrilateral and triangular meshes the cross over in efficiency arises between $4 < P < 6$ which is consistent with the earlier CPU tests shown in Fig. 7. We should recall however that the data in Fig. 7 contains the CPU cost to perform the solve for all degrees of freedom including the local (interior) elemental degrees of freedom rather than just the boundary/trace space. In addition there are other operations to perform inner products and enforce boundary conditions as well as potential influences from the processor architecture such as caching affects.

Finally we note that the implementation strategy of the multilevel static condensation [37] is not dependent upon the coupling between elemental domains but rather utilizes the compactness of the trace space degrees of freedom. The CG approach therefore benefits from the smaller trace space around a vertex for a fixed polynomial order as compared to the HDG technique. However in larger problems which are likely to require a parallel implementation the weaker coupling between the elements is likely to be advantageous from a communication point-of-view.

6.1 The Influence of τ and Post-Processing

To highlight the role of τ and the post-processing technique discussed in Sect. 2.7 we next compare the solutions using the quadrilateral and triangular discretizations with two different values of τ. Therefore we consider a value of $\tau = 1$ with the case when $\tau = 1000$ (which is close to the CG solution) for the Poisson equation when $\lambda = 0$ with the exact solution given by (35). We recall that a comparison of the $\tau = 1$ HDG with CG method was also considered in [21] for up to third-order polynomial expansions. In Fig. 9 we plot the L_2 error of the solution $u(x, y)$ as a function of the mesh size h for both the triangular (top) and quadrilateral (bottom) discretizations. The solid lines in these figures denote the HDG without any post-processing whilst the dotted lines show the HDG solution when elemental $(P+1)$ post-processing is applied.

We see for the triangular domain without the post-processing at $\tau = 1$ and $\tau = 1000$ the u-field converges at a rate of $O(P+1)$ however for the $\tau = 1$ case we observe that the error is typically larger by a constant factor of up to a factor 10 at higher polynomial orders. If we

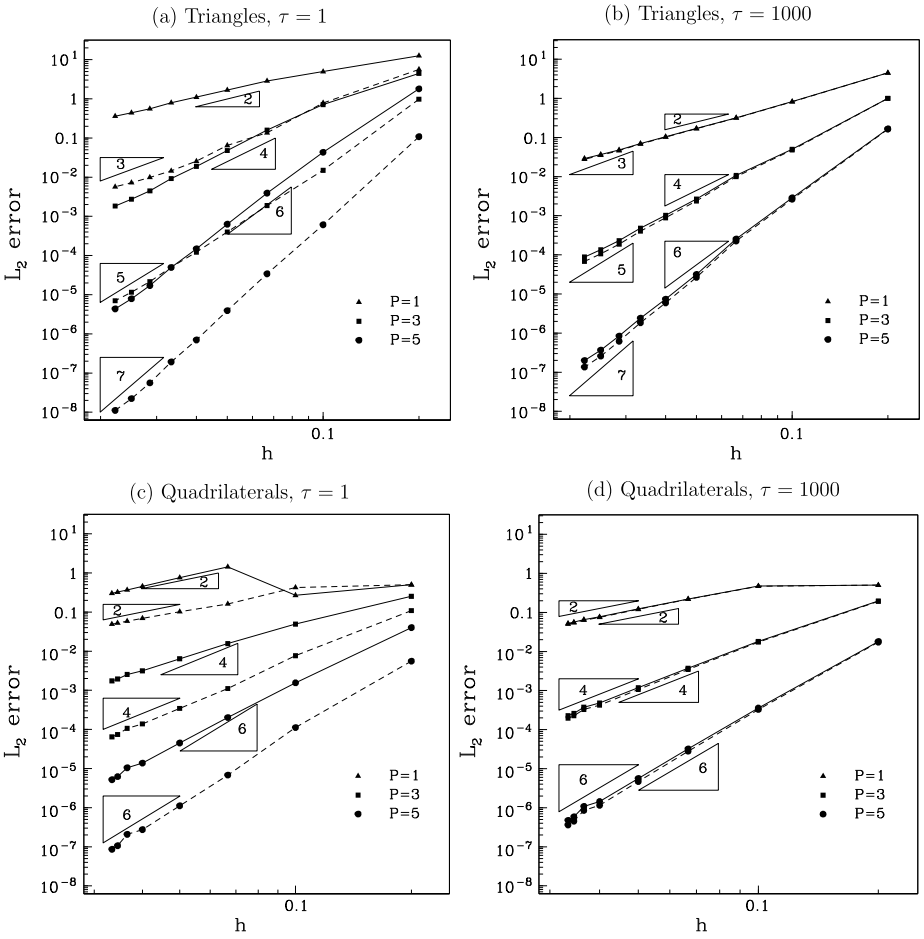

Fig. 9 L_2 error of the solution to the Poisson problem as a function of mesh size h for different polynomial expansions on triangular and quadrilateral meshes at $\tau = 1$ and $\tau = 1000$. *The solid lines* denote the HDG solution with no post-processing and *the dotted lines* represent the HDG solution with post-processing as discussed in Sect. 2.7

consider the $(P + 1)$ post-processed results we observe that the $\tau = 1$ case now has a similar constant factor as the $\tau = 1000$ case but the convergence order is increased to $(P + 2)$. In Fig. 9 some care must be taken in interpreting the data since the post-processed results for $P = 1$ overlap with no post-processing case at $P = 3$ for $\tau = 1$. The post-processing results for the $\tau = 1000$ case do not show an improvement as significant as for the $\tau = 1$ case. In part this is due to the observation that the $\tau = 1000$ HDG problem with no-processing is achieving a slightly higher convergence rate than $(P + 1)$ at small values of h for the test case we have considered.

For the u-field convergence of quadrilateral meshes is shown in Figs. 9(c) and (d). We observe that at $\tau = 1000$ we obtain the expected convergence rate of $(P + 1)$ both with and without post-processing. However for the $\tau = 1$ case, the error on again tends to have a higher constant at higher polynomial orders when no post-processing is applied. Also in this case the solution is not quite achieving the $(P + 1)$ rate of convergence. However when

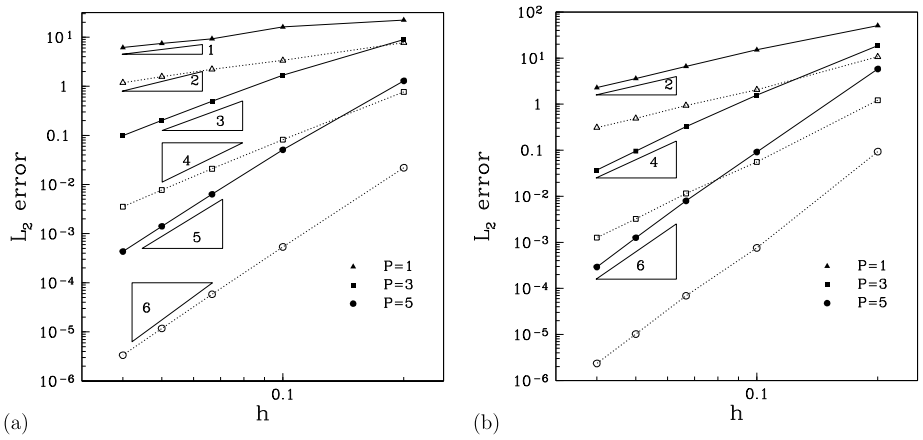

Fig. 10 L_2 error of the weak discontinuous Galerkin flux q to the solution to the Poisson problem as a function of mesh size h for different polynomial expansions on (**a**) quadrilateral and (**b**) triangular mesh. The *solid lines* denote the HDG error for the solution for $u = \sin(10\pi x)\cos(10\pi y)$ and the *dotted lines* denote the HDG error for the solution $u = \sin(5\pi x)\cos(5\pi y)$. Both solutions are for $\tau = 1$

we again elementally post-process the data, we observe the constant is significantly reduced and the expected $(P + 1)$ convergence is recovered.

It is interesting to see that for the relatively under resolved discretization of $P = 1$ in the quadrilateral mesh, the HDG solution with $\tau = 1$ does not converge monotonically in the L_2 norm. The solution does however converge more uniformly in the flux q as shown in Fig. 10. In processing this data we have evaluated the flux q_i using the DG solution solving (17c). This is in full agreement with the theoretical results in [23, 26] which show that for any constant τ, the asymptotic order of convergence is $O(P+1)$ and that when τ is of order $1/h$, the order is only $O(P)$. In contrast, for quadrilateral meshes, the flux convergences at a rate of $O(P)$ for a $\tau = 1$ HDG discretisation.

This sub-optimal behavior, however, can be explained by the fact that our DG approximation is still rather under-resolved for this particular numerical tests over the polynomial ranges provided considered. The less oscillatory problem of $u(x, y) = \sin(5\pi x)\cos(5\pi y)$ demonstrates $O(P + 1)$ convergence, particularly for the better resolved $P = 5$ problem, when using $\tau = 1$.

7 Discussion and Conclusions

We have directly compared an HDG and CG implementation of a two-dimensional elliptic scalar problem using triangular and quadrilateral polynomial expansions ranging in order from $1 \leq P \leq 14$. We have discussed how to efficiently formulate the HDG framework from an implementation point of view and draw direct comparison between the HDG and CG techniques. In this framework, the only differences between the methods are (i) the finite dimensional space in which λ is taken (discontinuous piecewise polynomials for the mixed methods and the HDG methods, and continuous piecewise polynomials for the CG method), and (ii) the discrete operators, called the *local solvers* relating λ on the boundary of the elements to the approximate solution inside them (obtained by discretizing the problem by a mixed, DG or CG method, on each element).

Moreover, the framework suggested that, roughly speaking, the HDG methods were *between* the mixed methods and the CG method, the HDG being closer to the mixed methods

when their stabilisation function was *small* and being closer to the CG method when is was *large*. In the former case, the normal component of the jump across inter-element boundaries of the approximation to the gradient is small whereas in the second, the jump of the approximation of the scalar variable is small, as expected. This fact was put in firm mathematical ground in [26] where it was shown that when the stabilisation function of the HDG method is of order one, the method has all the optimal convergence and super-convergence of the mixed methods (see also [23]), and that when the stabilisation function is of order $1/h$, h being the maximum diameter of the elements, the HDG method has the same convergence properties of the CG method (in full agreement with the results in [12]). In considering the error of a simple test case we observed that the $\tau = 1$ case leads to faster convergence in the flux at lower polynomial orders but a higher constant term. This constant offset may well be offset by the use of the post-processing techniques discussed in Sect. 2.7 [19, 23, 26].

In our numerical tests we have demonstrated that for polynomial expansion of fifth-order and higher that the HDG solver can be as efficient (within 20%) as the CG technique when comparing solution time, rather than setup costs, which is typically relevant to time dependent PDE solutions where linear components are treated implicitly. Despite the higher rank of the HDG approach the weaker elemental coupling of the technique leads to a reduced matrix bandwidth and faster compute time when using banded LU factorisation. Nevertheless other matrix solution strategies, such as the multi-level static condensation [37], are known to be more efficient and do not rely on bandwidth structure for their efficiency. In this case we observe the HDG to be 20% slow than the CG approach at higher polynomial orders which is still relatively competitive.

There are obviously other advantages of the HDG approach such as local conservation properties and the greater potential for efficient parallel communication. Adaptivity is also often considered another advantage of the HDG approach. In this setting we would have to balance the higher setup cost of the HDG approach when compared to the CG method against the more flexible manner of handling refinement and derefinement allowed by the HDG methods, as well as against the fact that coarser meshes might be needed to reach the same accuracy.

Let us end by pointing out that one could surmise that a combination of the main assets of the above-mentioned methods would be optimal. We could see the introduction of the so-called multiscale discontinuous Galerkin (MDG) [10, 36] in this light. In our setting, this can be done by taking the LDG method to define the discrete local solvers and the hybrid unknown λ in the space of continuous piecewise continuous functions. However, in [25] it was proven that the resulting method, which was called the embedded discontinuous Galerkin method (EDG) [45], loses the convergence properties of the associated HDG method and behaves closely to that of the CG method.

Acknowledgements The authors would like to thank Dr. Cuoug Nguyen for his comparative runs which highlighted the asymptotic convergence of the problem examined in Fig. 10(a). The first author gratefully acknowledges the support provided under NSF Career Award NSF-CCF0347791 (R. Kirby), NIH grant 5R01HL067646 (A. Cheung), and the Leverhulme Trust. The second author would like to acknowledge support from Advanced Research Fellowship from EPSRC. The third author would like to acknowledge support from the National Science Foundation (Grant DMS-0712955).

References

1. Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Sorensen, D.: LAPACK Users' Guide, 3rd edn. Society for Industrial and Applied Mathematics, Philadelphia (1999)

2. Arnold, D.N., Brezzi, F.: Mixed and nonconforming finite element methods: implementation, postprocessing and error estimates. RAIRO Modél. Math. Anal. Numér. **19**, 7–32 (1985)
3. Arnold, D.N., Brezzi, F., Cockburn, B., Marini, D.: Discontinuous Galerkin methods for elliptic problems. In: Discontinuous Galerkin Methods: Theory, Computation and Applications, pp. 89–101. Springer, Berlin (2000)
4. Arnold, D.N., Brezzi, F., Cockburn, B., Marini, D.: Unified analysis of discontinuous Galerkin methods for elliptic problems. SIAM J. Numer. Anal. **39**, 1749–1779 (2002)
5. Blackburn, H.M., Sherwin, S.J.: Instability modes and transition of pulsatile stenotic flow: pulse-period dependence. J. Fluid Mech. **573**, 57 (2007)
6. Blackburn, H.M., Barkley, D., Sherwin, S.J.: Convective instability and transient growth in flow over a backwards-facing step. J. Fluid Mech. **603**, 271–304 (2008)
7. Boost c++ libraries
8. Bramble, J.H., Xu, J.: A local post-processing technique for improving the accuracy in mixed finite-element approximations. SIAM J. Numer. Anal. **26**(6), 1267–1275 (1989)
9. Brezzi, F., Douglas, J., Jr., Marini, L.D.: Two families of mixed finite elements for second order elliptic problems. Numer. Math. **47**, 217–235 (1985)
10. Buffa, A., Hughes, T.J.R., Sangalli, G.: Analysis of a multiscale discontinuous Galerkin method for convection-diffusion problems. SIAM J. Numer. Anal. **44**, 1420–1440 (2006). (electronic)
11. Carmo, B.S., Sherwin, S.J., Bearman, P.W., Willden, R.H.J.: Wake transition in the flow around two circular cylinders in staggered arrangements. J. Fluid Mech. **597**, 1–29 (2008)
12. Castillo, P., Cockburn, B., Perugia, I., Schötzau, D.: An a priori error analysis of the local discontinuous Galerkin method for elliptic problems. SIAM J. Numer. Anal. **38**, 1676–1706 (2000)
13. Chen, Z.: Equivalence between and multigrid algorithms for nonconforming and mixed methods for second-order elliptic problems. Math. Comput. **4**, 1–33 (1996)
14. Ciarlet, P.: The Finite Element Method for Elliptic Problems. North-Holland, Amsterdam (1978)
15. Cockburn, B.: Discontinuous Galerkin methods. Z. Angew. Math. Mech. **83**, 731–754 (2003)
16. Cockburn, B.: Discontinuous Galerkin methods for computational fluid dynamics. In: Borst, R., Stein, E., Hughes, T.J.R. (eds.) Encyclopedia of Computational Mechanics, vol. 3, pp. 91–123. Wiley, New York (2004)
17. Cockburn, B., Gopalakrishnan, J.: A characterization of hybridized mixed methods for second order elliptic problems. SIAM J. Numer. Anal. **42**, 283–301 (2004)
18. Cockburn, B., Gopalakrishnan, J.: Error analysis of variable degree mixed methods for elliptic problems via hybridization. Math. Comput. **74**, 1653–1677 (2005)
19. Cockburn, B., Dong, B., Guzmán, J.: A superconvergent LDG-hybridizable Galerkin method for second-order elliptic problems. Math. Comput. **77**, 1887–1916 (2008)
20. Cockburn, B., Shu, C.-W.: Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. J. Sci. Comput. **16**, 173–261 (2001)
21. Cockburn, B., Dong, B., Guzmán, J., Restelli, M., Sacco, R.: A hybridizable discontinuous Galerkin method for steady-state convection-diffusion-reaction problems. SIAM J. Sci. Comput. **31**, 3827–3846 (2009)
22. 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**, 1319–1365 (2009)
23. Cockburn, B., Gopalakrishnan, J., Sayas, F.-J.: A projection-based error analysis of HDG methods. Math. Comput. **79**, 1351–1367 (2010)
24. Cockburn, B., Gopalakrishnan, J., Wang, H.: Locally conservative fluxes for the continuous Galerkin method. SIAM J. Numer. Anal. **45**, 1742–1776 (2007)
25. Cockburn, B., Guzmán, J., Soon, S.-C., Stolarski, H.K.: An analysis of the embedded discontinuous Galerkin method for second-order elliptic problems. SIAM J. Numer. Anal. **47**, 2686–2707 (2009)
26. Cockburn, B., Guzmán, J., Wang, H.: Superconvergent discontinuous Galerkin methods for second-order elliptic problems. Math. Comput. **78**, 1–24 (2009)
27. Cockburn, B., Karniadakis, G.E., Shu, C.-W.: Discontinuous Galerkin Methods: Theory, Computation and Applications. Springer, Berlin (2000)
28. Comodi, M.I.: The Hellan-Herrmann-Johnson method: some new error estimates and postprocessing. Math. Comput. **52**, 17–29 (1989)
29. Darekar, R., Sherwin, S.J.: Flow past a square-section cylinder with a wavy stagnation face. J. Fluid Mech. **426**, 263 (2001)
30. Deville, M.O., Mund, E.H., Fischer, P.F.: High Order Methods for Incompressible Fluid Flow. Cambridge University Press, Cambridge (2002)
31. Dubiner, M.: Spectral methods on triangles and other domains. J. Sci. Comput. **6**, 345–390 (1991)

32. Fraejis de Veubeke, B.M.: Displacement and equilibrium models in the finite element method. In: Zienkiewicz, O.C., Holister, G. (eds.) Stress Analysis, pp. 145–197. Wiley, New York (1977)
33. Gastaldi, L., Nochetto, R.H.: Sharp maximum norm error estimates for general mixed finite element approximations to second order elliptic equations. RAIRO Modél. Math. Anal. Numér. **23**, 103–128 (1989)
34. Guermond, J.L., Shen, J.: Velocity-correction projection methods for incompressible flows. SIAM J. Numer. Anal. **41**, 112–134 (2003)
35. Hughes, T.J.R.: The Finite Element Method: Linear Static and Dynamic Finite Element Analysis. Prentice-Hall, New York (1987)
36. Hughes, J.T.R., Scovazzi, G., Bochev, P.B., Buffa, A.: A multiscale discontinuous Galerkin method with the computational structure of a continuous Galerkin method. Comput. Methods Appl. Mech. Eng. **195**, 2761–2787 (2006)
37. Karniadakis, G.E., Sherwin, S.J.: Spectral/*hp* Element Methods for CFD, 2nd edn. Oxford University Press, London (2005)
38. Karniadakis, G.E., Israeli, M., Orszag, S.A.: High-order splitting methods for the incompressible Navier–Stokes equations. J. Comput. Phys. **97**(2), 414–443 (1991)
39. Koornwinder, T.: Two-variable analogues of the classical orthogonal polynomials. In: Theory and Applications of Special Functions. Academic Press, San Diego (1975)
40. Proriol, J.: Sur une famille de polynomes á deux variables orthogonaux dans un triangle. C.R. Acad. Sci Paris **257**, 2459–2461 (1957)
41. Raviart, P.A., Thomas, J.M.: A mixed finite element method for second order elliptic problems. In: Galligani, I., Magenes, E. (eds.) Mathematical Aspects of Finite Element Method. Lecture Notes in Math., vol. 606, pp. 292–315. Springer, New York (1977)
42. Schwab, Ch.: *p*- and *hp*- Finite Element Methods: Theory and Applications to Solid and Fluid Mechanics. Oxford University Press, London (1999)
43. Sherwin, S.J.: Hierarchical hp finite elements in hybrid domains. Finite Elem. Anal. Des. **27**, 109 (1997)
44. Sherwin, S.J., Karniadakis, G.E.: A triangular spectral element method; applications to the incompressible Navier-stokes equations. Comput. Methods Appl. Mech. Eng. **123**, 189–229 (1995)
45. Soon, S.-C., Cockburn, B., Stolarski, H.K.: A hybridizable discontinuous Galerkin method for linear elasticity. Int. J. Numer. Methods Eng. **80**, 1058–1092 (2009)
46. Stenberg, R.: A family of mixed finite elements for the elasticity problem. Numer. Math. **53**, 513–538 (1988)
47. Stenberg, R.: Postprocessing schemes for some mixed finite elements. RAIRO Modél. Math. Anal. Numér. **25**, 151–167 (1991)
48. Szabó, B.A., Babuška, I.: Finite Element Analysis. Wiley, New York (1991)
49. Vos, P.E.J., Sherwin, S.J., Kirby, R.M.: From *h* to *p* efficiently: Implementing finite and spectral/*hp* element methods to achieve optimal performance for low and high order discretisations. J. Comput. Phys. **29**(13), 5161–5181 (2010)
50. Zienkiewicz, O.C., Taylor, R.L.: The Finite Element Method: Fluid Mechanics, vol. 3, 5th edn. Butterworth-Heinemann, Oxford (2000)
51. Zienkiewicz, O.C., Taylor, R.L.: The Finite Element Method: Solid Mechanics, vol. 2, 5th edn. Butterworth-Heinemann, Oxford (2000)
52. Zienkiewicz, O.C., Taylor, R.L.: The Finite Element Method: The Basis, vol. 1, 5th edn. Butterworth-Heinemann, Oxford (2000)