https://www.aimspress.com/journal/mine

Mathematics in Engineering, 6(2): 221–237.
DOI:10.3934/mine.2024010
Received: 07 September 2023
Revised: 11 December 2023
Accepted: 28 February 2024
Published: 11 March 2024

Research article

Efficient implementation of the hybridized Raviart-Thomas mixed method by converting flux subspaces into stabilizations†

Sreevatsa Anantharamu[1] **and Bernardo Cockburn**[2,*]

[1] Senior Applications Engineer, X-ScaleSolutions, LLC, USA

[2] School of Mathematics, University of Minnesota, Minneapolis, MN 55455, USA

† **This contribution is part of the Special Issue:** Advancements in Polytopal Element Methods
Guest Editors: Michele Botti; Franco Dassi; Lorenzo Mascotto; Ilario Mazzieri
Link: www.aimspress.com/mine/article/6538/special-articles

* **Correspondence:** Email: bcockbur@umn.edu; Tel: +16126252587.

Abstract: We show how to reduce the computational time of the practical implementation of the Raviart-Thomas mixed method for second-order elliptic problems. The implementation takes advantage of a recent result which states that certain local subspaces of the vector unknown can be eliminated from the equations by transforming them into stabilization functions; see the paper published online in JJIAM on August 10, 2023. We describe in detail the new implementation (in MATLAB and a laptop with Intel(R) Core (TM) i7-8700 processor which has six cores and hyperthreading) and present numerical results showing 10 to 20% reduction in the computational time for the Raviart-Thomas method of index k, with k ranging from 1 to 20, applied to a model problem.

Keywords: mixed methods; hybridization; static condensation; mixed methods; hybridizable discontinuous Galerkin methods

1. Introduction

Let us begin by noting that the title of the paper contains both words: 'mixed methods' and 'stabilization'. At first glance, this might appear to be a technical error since mixed methods do not need stabilization since stability is directly ensured by the choice of their function spaces. However, this is *not* an error. In a recent paper, [5], one of the authors showed how a portion of the flux space of any hybridized mixed method can be recast as a stabilization of a equivalent hybridizable discontinuous Galerkin (HDG) methods. By *equivalent*, we mean that the original hybridized mixed method and the new HDG methods result in exactly the same solution. In this paper, we show that the implementation

of the *equivalent* HDG method is faster than that of original hybridized mixed method. We carry out this for *two* equivalent HDG methods.

We apply this implementation procedure to a particular mixed method, namely, the Raviart-Thomas (RT) method [13] in symplexes for two reasons. The first is that its extension to any other mixed (or HDG) method defined on polytopal meshes is particularly straightforward* The second reason is that, if this procedure does not work on a simple mixed method, it is very unlikely it would have a chance to work on more sophisticated mixed methods on polytopal meshes. In other words, we consider the material presented here as a necessary stepping stone towards the treatment of the cases of mixed and HDG methods defined in general polytopes.

The paper is organized as follows. In Section 2, we describe how to obtain an HDG method from a hybridized mixed method, we follow [5]. In Section 3, we describe in full detail the implementation of the hybridized RT method. We then do the same for *two* equivalent HDG methods. In Section 4, we display our numerical results. We end with some concluding remarks. We use the standard notation used for HDG methods, see, for example, [4, 5].

2. Background

2.1. Subspace-to-stabilization.

For the sake of completeness, we begin by summarizing the subspace-to-stabilization result [5, Section 5]. Consider the Poisson problem:

$$\begin{aligned} c\boldsymbol{q} &= -\boldsymbol{\nabla} u, \text{ in } \Omega, \text{ and} \\ \boldsymbol{\nabla}\cdot\boldsymbol{q} &= f, \text{ in } \Omega, \end{aligned} \tag{2.1}$$

with the boundary condition that $u = u_D$ on the boundary $\partial\Omega$. Here, c is the coefficient, f is the source term and u_D is the Dirichlet boundary data. In this paper, we consider the case $c = Id$. The method can be easily generalized to arbitrary symmetric positive definite tensor fields c. The hybridized mixed method formulation for this problem is as follows: For each element K of the mesh, find $\boldsymbol{q}_h$ and u_h belonging to the local function spaces $\boldsymbol{V}(K)$ and $W(K)$, respectively, such that:

$$\begin{aligned} (\boldsymbol{q}_h, \boldsymbol{v})_K - (u_h, \boldsymbol{\nabla}\cdot\boldsymbol{v})_K &= -\langle \hat{u}_h, \boldsymbol{v}\cdot\boldsymbol{n}\rangle_{\partial K}, \\ (\boldsymbol{\nabla}\cdot\boldsymbol{q}_h, w)_K &= (f, w)_K, \end{aligned} \tag{2.2}$$

for all test functions $\boldsymbol{v}$ and w belonging to the local function spaces $\boldsymbol{V}(K)$ and $W(K)$, respectively. The above equations are referred to as the 'local problem'. Here, $\hat{u}_h$ is an approximation to u on the faces of the triangulation and is a data to the above problem. The additional equations for this face variable are:

$$\begin{aligned} \langle \boldsymbol{q}_h^+\cdot\boldsymbol{n}^+ + \boldsymbol{q}_h^-\cdot\boldsymbol{n}^-, \mu\rangle_{F_I} &= 0, \text{ for all } \mu \in M_h(F_I), \\ \langle \hat{u}_h, \mu\rangle_{F_D} &= \langle u_D, \mu\rangle_{F_D}, \text{ for all } \mu \in M_h(F_D), \end{aligned} \tag{2.3}$$

where F_I is each interior face of the mesh, F_D is each Dirichlet boundary face of the mesh and $M_h(F)$ is the local function space on face F. The above equations are referred to as the 'global problem'.

*How to define mixed methods for polyhedral elements has been shown in the series of papers on M-decompositions [6, 7, 9] and on new commuting diagrams [8]. See also the review [10] and the references therein.

Split $\boldsymbol{V}(K)$ into $\boldsymbol{V}_s(K) \oplus \boldsymbol{V}_a(K)$. Here, $\boldsymbol{V}_s(K)$ is the subspace of $\boldsymbol{V}(K)$ that will be converted into stabilization and $\boldsymbol{V}_a(K)$ is the subspace of $\boldsymbol{V}(K)$ that will be used to define the local problem of the equivalent HDG method. Their exact definition is deferred until later. This splitting converts the local problem (Eq (2.2)) into:

$$\begin{aligned}
\left((\boldsymbol{q}_a + \boldsymbol{q}_s), \boldsymbol{v}_a\right)_K - (u_h, \boldsymbol{\nabla} \cdot \boldsymbol{v}_a)_K &= -\langle \hat{u}_h, \boldsymbol{v}_a \cdot \boldsymbol{n} \rangle_{\partial K}, \\
\left((\boldsymbol{q}_a + \boldsymbol{q}_s), \boldsymbol{v}_s\right)_K - (u_h, \boldsymbol{\nabla} \cdot \boldsymbol{v}_s)_K &= -\langle \hat{u}_h, \boldsymbol{v}_s \cdot \boldsymbol{n} \rangle_{\partial K}, \\
\left(\boldsymbol{\nabla} \cdot (\boldsymbol{q}_a + \boldsymbol{q}_s), w\right)_K &= (f, w)_K,
\end{aligned}$$

for all test functions $\boldsymbol{v}_a$, $\boldsymbol{v}_s$, and w in the local function spaces $\boldsymbol{V}_a(K)$, $\boldsymbol{V}_s(K)$, and $W(K)$, respectively. Requiring the functions in $\boldsymbol{V}_a(K)$ and $\boldsymbol{V}_s(K)$ to be orthogonal to each other in the $(\cdot, \cdot)_K$ inner-product, we obtain:

$$\begin{aligned}
(\boldsymbol{q}_a, \boldsymbol{v}_a)_K - (u_h, \boldsymbol{\nabla} \cdot \boldsymbol{v}_a)_K &= -\langle \hat{u}_h, \boldsymbol{v}_a \cdot \boldsymbol{n} \rangle_{\partial K}, \\
(\boldsymbol{q}_s, \boldsymbol{v}_s)_K - (u_h, \boldsymbol{\nabla} \cdot \boldsymbol{v}_s)_K &= -\langle \hat{u}_h, \boldsymbol{v}_s \cdot \boldsymbol{n} \rangle_{\partial K}, \\
\left(\boldsymbol{\nabla} \cdot (\boldsymbol{q}_a + \boldsymbol{q}_s), w\right)_K &= (f, w)_K.
\end{aligned} \tag{2.4}$$

Integrating the second equation by parts and requiring $\boldsymbol{V}_s(K)$ to be any subspace of $\boldsymbol{V}(K)$ that is $L^2(K)$-orthogonal to $\boldsymbol{\nabla} W(K)$ yields

$$(\boldsymbol{q}_s, \boldsymbol{v}_s)_K = \langle u_h - \hat{u}_h, \boldsymbol{v}_s \cdot \boldsymbol{n} \rangle_{\partial K}. \tag{2.5}$$

Note that in the above equation, $\boldsymbol{q}_s$ (which is the portion of q in the subspace $\boldsymbol{V}_s(K)$) depends solely on the jump $u_h - \hat{u}_h$ on the faces of the element. Observe that the appearance of the term $u_h - \hat{u}_h$ has some similarity to the flux stabilization $\tau(u_h - \hat{u}_h)$ that is used in a HDG method (note that $(\tau(\cdot)$ here is a linear-mapping that satisfies certain requirements). This similarity is exploited to define a stabilization function $\tau(u_h - \hat{u}_h)$ for the HDG method that is equivalent to the above hybridized mixed method below.

Based on the form of Eq (2.5), let us define $\boldsymbol{L}_{\boldsymbol{V}_s}$ to be the local lifting operator that maps a function μ in the space $L^2(\partial K)$ to the function $\boldsymbol{L}_{\boldsymbol{V}_s}(\mu)$ in the space $\boldsymbol{V}_s(K)$ as:

$$(\boldsymbol{L}_{\boldsymbol{V}_s}(\mu), \boldsymbol{v}_s)_K = \langle \mu, \boldsymbol{v}_s \cdot \boldsymbol{n} \rangle_{\partial K}, \tag{2.6}$$

for all test functions $\boldsymbol{v}_s$ in the local space $\boldsymbol{V}_s(K)$. Then, $\boldsymbol{q}_s = \boldsymbol{L}_{\boldsymbol{V}_s}(u_h - \hat{u}_h)$, where by $\boldsymbol{L}_{\boldsymbol{V}_s}(u_h)$, we mean $\boldsymbol{L}_{\boldsymbol{V}_s}(u_h|_{\partial K})$. Moreover, the hybridized mixed method in Eq (2.4) can be manipulated to the following HDG method for the portion $\boldsymbol{q}_a$ of the flux approximation and the full scalar approximation u_h: Find $\boldsymbol{q}_a$ and u_h in the local function spaces $\boldsymbol{V}_a(K)$ and $W(K)$, respectively, such that:

$$\begin{aligned}
(\boldsymbol{q}_a, \boldsymbol{v}_a)_K - (u_h, \boldsymbol{\nabla} \cdot \boldsymbol{v}_a)_K &= -\langle \hat{u}_h, \boldsymbol{v}_a \cdot \boldsymbol{n} \rangle_{\partial K}, \\
-(\boldsymbol{q}_a, \boldsymbol{\nabla} w)_K + \langle \boldsymbol{q}_h \cdot \boldsymbol{n}, w \rangle_{\partial K} &= (f, w)_K, \text{ and} \\
\boldsymbol{q}_h \cdot \boldsymbol{n} &= \boldsymbol{q}_a \cdot \boldsymbol{n} + \tau(u_h - \hat{u}_h),
\end{aligned} \tag{2.7}$$

where the stabilization function $\tau(\cdot)$ is defined using the lifting operator as: $\tau(u_h - \hat{u}_h) = \boldsymbol{n} \cdot \boldsymbol{L}_{\boldsymbol{V}_s}(u_h - \hat{u}_h)$. The global problem of the HDG method is same as the one in Eq (2.3).

The flux approximation $\boldsymbol{q}_h$ of the hybridized mixed method in Eq (2.2) is then obtained from the above HDG method as $\boldsymbol{q}_h = \boldsymbol{q}_a + \boldsymbol{L}_{\boldsymbol{V}_s}(u_h - \hat{u}_h)$. Thus, the effect of the local space $\boldsymbol{V}_s(K)$ is fully encapsulated in the defined stabilization function τ via the lifting operator $\boldsymbol{L}_{\boldsymbol{V}_s}$. This is the crux of the 'spaces-to-stabilization' idea.

In summary, the conditions on the local spaces $\boldsymbol{V}_a(K)$ and $\boldsymbol{V}_s(K)$ are that:

$$\begin{aligned}&\boldsymbol{V}_s(K) \text{ should be any subspace of } \boldsymbol{V}(K) \text{ that is } L^2\text{-orthogonal to } \nabla W(K),\\ &\boldsymbol{V}_a(K) \text{ should be the remaining portion of } \boldsymbol{V}(K) \text{ that is orthogonal to } \boldsymbol{V}_s(K)\\ &\text{in the } (\cdot,\cdot)_K \text{ inner-product.}\end{aligned} \tag{2.8}$$

The former and latter conditions above were used to obtain the Eqs (2.5) and (2.4), respectively.

2.2. The equivalent hybridizable discontinuous Galerkin method after static condensation

Similar to other HDG methods, the degrees of freedom corresponding to $\boldsymbol{q}_a$ and u_h can be statically condensed to yield a globally-coupled problem just for the degrees of freedom corresponding to the face variable $\hat{u}_h$ as follows. The local problem for the equivalent HDG method given in Eq (2.7) can be shown to be equal to the following local problem:

Find $\boldsymbol{q}_a$ and u_h in the local function spaces $\boldsymbol{V}_a(K)$ and $W(K)$, respectively, such that

$$\begin{aligned}(\boldsymbol{q}_a, \boldsymbol{v}_a)_K - (u_h, \boldsymbol{\nabla}\cdot\boldsymbol{v}_a)_K &= -\langle \hat{u}_h, \boldsymbol{v}_a\cdot\boldsymbol{n}\rangle_{\partial K},\\ (\boldsymbol{\nabla}\cdot\boldsymbol{q}_a, w)_K + (\boldsymbol{L}_{\boldsymbol{V}_s}(u_h), \boldsymbol{L}_{\boldsymbol{V}_s}(w))_K &= (f,w)_K + (\boldsymbol{L}_{\boldsymbol{V}_s}(\hat{u}_h), \boldsymbol{L}_{\boldsymbol{V}_s}(w))_K,\end{aligned} \tag{2.9}$$

for all test functions $\boldsymbol{v}_a$ and w in the local function spaces $\boldsymbol{V}_a(K)$ and $W(K)$, respectively. The influence of $\hat{u}_h$ and f on $(\boldsymbol{q}_a, u_h)$ can be separated as $(\boldsymbol{q}_a, u_h) = (\boldsymbol{Q}_{\hat{u}_h}, U_{\hat{u}_h}) + (\boldsymbol{Q}_f, U_f)$. Here, $(\boldsymbol{Q}_{\hat{u}_h}, U_{\hat{u}_h})$ is the solution to the following problem with $\mu = \hat{u}_h$:

Find $\boldsymbol{Q}_\mu$ and U_μ in the local function spaces $\boldsymbol{V}_a(K)$ and $W(K)$, respectively, such that:

$$\begin{aligned}(\boldsymbol{Q}_\mu, \boldsymbol{v}_a)_K - (U_\mu, \boldsymbol{\nabla}\cdot\boldsymbol{v}_a)_K &= -\langle \mu, \boldsymbol{v}_a\cdot\boldsymbol{n}\rangle_{\partial K},\\ (\boldsymbol{\nabla}\cdot\boldsymbol{Q}_\mu, w)_K + (\boldsymbol{L}_{\boldsymbol{V}_s}\left(U_\mu\right), \boldsymbol{L}_{\boldsymbol{V}_s}(w))_K &= (\boldsymbol{L}_{\boldsymbol{V}_s}(\mu), \boldsymbol{L}_{\boldsymbol{V}_s}(w))_K,\end{aligned} \tag{2.10}$$

for all test functions $\boldsymbol{v}_a$ and w in the local function spaces $\boldsymbol{V}_a(K)$ and $W(K)$, respectively. $(\boldsymbol{Q}_f, U_f)$ is the solution to the problem:

Find $\boldsymbol{Q}_f$ and U_f in the local function spaces $\boldsymbol{V}_a(K)$ and $W(K)$, respectively, such that:

$$\begin{aligned}(\boldsymbol{Q}_f, \boldsymbol{v}_a)_K - (U_f, \boldsymbol{\nabla}\cdot\boldsymbol{v}_a)_K &= 0,\\ (\boldsymbol{\nabla}\cdot\boldsymbol{Q}_f, w)_K + (\boldsymbol{L}_{\boldsymbol{V}_s}\left(U_f\right), \boldsymbol{L}_{\boldsymbol{V}_s}(w))_K &= (f,w)_K,\end{aligned} \tag{2.11}$$

for all test functions $\boldsymbol{v}_a$ and w in the local function spaces $\boldsymbol{V}_a(K)$ and $W(K)$, respectively.

Using the above decomposition, we can show that the equation for $\hat{u}_h$ given in Eq (2.3) is equal to the following problem:

Find $\hat{u}_h$ belonging to the global function space M_h such that:

$$\begin{aligned}\sum_K (\boldsymbol{Q}_{\hat{u}_h}, \boldsymbol{Q}_\mu)_K + \sum_K (\boldsymbol{L}_{\boldsymbol{V}_s}(U_{\hat{u}_h} - \hat{u}_h), \boldsymbol{L}_{\boldsymbol{V}_s}(U_\mu - \mu))_K &= \sum_K (f, U_\mu)_K, \quad \forall \mu \in M_h(F_I),\\ \langle \hat{u}_h, \mu\rangle_{F_D} &= \langle u_D, \mu\rangle_{F_D}, \quad \forall \mu \in M_h(F_D),\end{aligned} \tag{2.12}$$

for each interior face F_I and Dirichlet boundary face F_D of the triangulation. Here, the global function space M_h is the set of functions in $L^2(\mathcal{F})$, where $\mathcal{F}$ is the union of all faces F in the mesh, and the restriction of M_h on face F is $M_h(F)$.

3. Implementations of the hybridized Raviart-Thomas mixed method

We use this subspace-to-stabilization idea to come up with two new implementations of the hybridized RT mixed method. Each implementation stems from a different choice of the subspace $\boldsymbol{V}_s(K)$. We note that for the usual hybridized RT method, the local spaces $\boldsymbol{V}(K)$, $W(K)$ and $M_h(F)$ are:

$$\boldsymbol{V}(K) = [P_k(K)]^d \oplus \boldsymbol{x}\widetilde{P}_k(K),\ W(K) = P_k(K) \text{ and } M_h(F) = P_k(F).$$

Here, $P_k(K)$ denotes the space polynomials of degree k defined on K, $\widetilde{P}_k(K)$ denotes the space of homogeneous polynomials of degree k defined on K and d is the spatial dimension of the problem. Table 1 shows the three choices of $\boldsymbol{V}_s(K)$ and the name of the implementation that stems from each of these choices.

Table 1. The different implementations of the hybridized RT method. Note that $\boldsymbol{V}_s^{(1)}(K)$ is the largest subspace of $\boldsymbol{V}$ that is $\boldsymbol{L}^2$-orthogonal to $[P_k(K)]^d$, and that $\boldsymbol{V}_s^{(2)}(K)$ is of the form $\boldsymbol{V}_s^{(1)}(K) \oplus \boldsymbol{V}^{(3)}(K)$ where $\boldsymbol{V}^{(3)}(K)$ is the subspace of polynomials in $[P_k(K)]^d$ that are $\boldsymbol{L}^2$-orthogonal to $[P_{k-1}(K)]^d$.

Implementation	$\boldsymbol{V}_s(K)$	Notes	$\boldsymbol{V}_a(K)$
Usual-HRT	$\varnothing$	Usual	$[P_k(K)]^d \oplus \boldsymbol{x}\widetilde{P}_k(K)$
Stab-1-HRT	$\boldsymbol{V}_s^{(1)}(K)$	New	$[P_k(K)]^d$
Stab-2-HRT	$\boldsymbol{V}_s^{(2)}(K)$	New	$[P_{k-1}(K)]^d$

The implementation Usual-HRT is the usual implementation of the hybridized RT method, see [1]. This stems from the choice $\boldsymbol{V}_s(K) = \varnothing$ (the empty set). The implementations Stab-1-HRT and Stab-2-HRT are the two new implementations proposed in this paper. The new implementation Stab-1-HRT stems from the choice $\boldsymbol{V}_s(K) = \boldsymbol{V}_s^{(1)}(K)$, where $\boldsymbol{V}_s^{(1)}(K)$ is the largest subspace of $\boldsymbol{V}(K)$ that is L^2-orthogonal to $[P_k(K)]^d$. The other new implementation $\boldsymbol{V}_s^{(2)}(K)$ stems from choosing $\boldsymbol{V}_s(K) = \boldsymbol{V}_s^{(2)}(K)$, where $\boldsymbol{V}_s^{(2)}(K)$ is the space $\boldsymbol{V}_s^{(2)}(K)$ plus the vector-valued polynomials in $[P_k(K)]^d$ that are L^2-orthogonal to $[P_{k-1}(K)]^d$.

For each of these implementations, the local space $\boldsymbol{V}_a(K)$ is the $(\cdot,\cdot)_K$-orthogonal complement of $\boldsymbol{V}_s(K)$ within $\boldsymbol{V}(K)$. The space $\boldsymbol{V}_a(K)$ for the Usual-HRT, Stab-1-HRT, and Stab-2-HRT implementation is $[P_k(K)]^d + \boldsymbol{x}P_k(K)$, $[P_k(K)]^d$, and $[P_{k-1}(K)]^d$, respectively. The details of each of these implementations are given next.

3.1. Usual-HRT (usual)

The details of the Usual-HRT implementation are given below.

3.1.1. Basis

In the Usual-HRT implementation, the space $\boldsymbol{V}_a(K)$ equals $[P_k(K)]^d \oplus \boldsymbol{x}\widetilde{P}_k(K)$. We use the following basis functions for this space in each element K:

$$\boldsymbol{\varphi}_1^{(K)}, \ldots, \boldsymbol{\varphi}_n^{(K)}.$$

Here, $n = dm+m'$, where $m = C_d^{k+d-1}$, $m' = C_{d-1}^{k+d-1}$ and C_a^b is the binomial coefficient $b!/(a!(b-a)!)$. The first dm basis functions $\boldsymbol{\varphi}_1^{(K)}, \ldots, \boldsymbol{\varphi}_{dm}^{(K)}$ correspond to the $[P_k(K)]^d$ portion of the space. The remaining m' functions $\boldsymbol{\varphi}_{dm+1}^{(K)}, \ldots, \boldsymbol{\varphi}_n^{(K)}$ correspond to the remaining portion of the space.

These basis functions are orthonormal to each other. They satisfy the orthonormality condition:

$$\int_K \boldsymbol{\varphi}_i^{(K)} \cdot \boldsymbol{\varphi}_j^{(K)} \, \mathrm{d}\Omega = |K|\delta_{ij}, \tag{3.1}$$

where δ_{ij} is the Kroenecker delta and $|K|$ is the measure (area in 2D and volume in 3D) of element K. The first dm basis functions $\boldsymbol{\varphi}_1, \ldots, \boldsymbol{\varphi}_{dm}$ are constructed using the (normalized) Dubiner polynomials [12] as:

$$\boldsymbol{\varphi}_{d(i'-1)+j'}^{(K)} = q_{i'}^{(K)} \boldsymbol{e}_{j'}, \text{ for } i' = 1, m \text{ and } j' = 1, \ldots, d. \tag{3.2}$$

Here, $\boldsymbol{e}_{j'}$ are the canonical basis functions of $\mathbb{R}^d$. $q_1, \ldots, q_m$ are the orthonormal Dubiner polynomials in the element K obtained by mapping the orthonormal polynomials in the reference simplex $\hat{K}$ to the element K as:

$$q_i^{(K)}(\boldsymbol{x}^{(K)}(\hat{\boldsymbol{x}})) = \hat{q}_i(\hat{\boldsymbol{x}}),$$

where $\boldsymbol{x}^{(K)}(\hat{\boldsymbol{x}})$ is the affine mapping from the reference element $\hat{K}$ to the element K. These polynomials satisfy the following orthonormality relation:

$$\int_K q_i^{(K)} q_j^{(K)} \, \mathrm{d}\Omega = |K|\delta_{ij}. \tag{3.3}$$

The remaining m' basis functions $\boldsymbol{\varphi}_{dm+1}^{(K)}, \ldots, \boldsymbol{\varphi}_n^{(K)}$ are constructed by multiplying the degree k Dubiner polynomial basis functions with $\boldsymbol{x}$ and orthonormalizing them with the rest of the basis functions using the modified Gram-Schmidt kernel. This is described in Algorithm 1.

Algorithm 1 Generating $\boldsymbol{\varphi}_{dm+1}^{(K)}, \ldots, \boldsymbol{\varphi}_n^{(K)}$.

for $i = dm + 1, \ldots, n$ **do**
  $\boldsymbol{\varphi}_i^{(K)} \leftarrow \boldsymbol{x} q_{i-(d-1)m-m'}^{(K)}$
  Orthonormalize $\boldsymbol{\varphi}_i^{(K)}$ against the previous $(i-1)$ basis functions using a modified Gram-Schmidt kernel
end for

For the space $W(K)$ (which equals $P_k(K)$), we use the same orthonormal Dubiner polynomial basis functions:

$$q_1^{(K)}, \ldots, q_m^{(K)}.$$

For the space $M_h(F)$, we also use orthonormal Dubiner polynomial basis functions but defined on the faces of the mesh. They are:

$$\psi_1^{(F)}, \ldots, \psi_{m'}^{(F)},$$

where F is a typical face of the mesh.

3.1.2. Local problem

Using the above basis functions converts the local problem that depends on μ (Eq (2.10)) to the following matrix problem:

$$\begin{bmatrix} I_{n\times n}|K| & -D^{(K)\ T}_{m\times n} \\ D^{(K)}_{m\times n} & 0_{m\times m} \end{bmatrix} \begin{bmatrix} Q^{\mu,(K)}_{n\times((d+1)m')} \\ U^{\mu,(K)}_{m\times((d+1)m')} \end{bmatrix} = \begin{bmatrix} -b^{Q^\mu,(K)}_{n\times((d+1)m')} \\ 0_{m\times((d+1)m')} \end{bmatrix}. \tag{3.4}$$

Here, $I_{n\times n}$ is the identity matrix, $0_{a\times b}$ is $a\times b$ matrix of zeros, $D^{(K)}_{m\times n}$ is the divergence matrix, $Q^{\mu,(K)}_{n\times(d+1)m'}$ is the degree of freedom matrix corresponding to the element-wise mapping $\boldsymbol{Q}_\mu$, $U^{\mu,(K)}_{m\times((d+1)m')}$ is the degree of freedom matrix corresponding to the element-wise mapping U_μ, and $-b^{Q^\mu,(K)}_{n\times((d+1)m')}$ is the right-hand side matrix corresponding to the degrees of freedom $Q^{\mu,(K)}_{n\times(d+1)m'}$. The size of all matrices is given in the subscript. The entries of the divergence and right-hand matrices are:

$$\left[D^{(K)}_{m\times n}\right]_{i,j} = \int_K q_i \boldsymbol{\nabla}\cdot\boldsymbol{\varphi}^{(K)}_j \,\mathrm{d}\Omega, \text{ and } \left[b^{Q^\mu,(K)}_{n\times((d+1)m')}\right]_{i,(j-1)m'+r} = \int_{F_j} \psi^{(F_j)}_r \boldsymbol{\varphi}^{(K)}_i \cdot \boldsymbol{n}\,\mathrm{d}\Gamma, \tag{3.5}$$

respectively, where F_j is the j^{th} face of element K.

For efficient solution of the above local problem, we perform two optimizations. The first optimization pertains to computing the divergence matrix $D^{(K)}_{m\times n}$. This matrix has the following form:

$$D^{(K)}_{m\times n} = \begin{bmatrix} D^{(K)^{(1)}}_{(m-m')\times dm} & D^{(K)^{(2)}}_{m\times m'} \\ 0_{m'\times dm} \end{bmatrix}. \tag{3.6}$$

Here, $D^{(K)^{(1)}}_{(m-m')\times dm}$ is the portion of the divergence-matrix from the first dm basis functions, $\boldsymbol{\varphi}^{(K)}_1,\ldots,\boldsymbol{\varphi}^{(K)}_{dm}$, which correspond to the $[P_k(K)]^d$ portion of the space. The remaining portion of the divergence matrix, $D^{(K)^{(2)}}_{m\times m'}$, is from the last m' basis functions, $\boldsymbol{\varphi}^{(K)}_{dm+1},\ldots,\boldsymbol{\varphi}^{(K)}_n$. This form is exploited for its efficient computation as follows. The portion $D^{(K)^{(1)}}_{(m-m')\times dm}$ is first computed in along the coordinates of the reference element $\hat{K}$. We will denote this reference divergence matrix as $\hat{D}_{(m-m')\times dm}$ and is given by:

$$\left[\hat{D}_{(m-m')\times dm}\right]_{i,j} = \int_{\hat{K}} \hat{q}_i \hat{\boldsymbol{\nabla}}\cdot\hat{\boldsymbol{\varphi}}_j \,\mathrm{d}\hat{\Omega}.$$

Then, to compute $D^{(K)^{(1)}}_{(m-m')\times dm}$ in each element K, we simply combine the columns of $\hat{D}_{(m-m')\times dm}$ using the entries of the Jacobian of $\boldsymbol{x}^{(K)}(\hat{\boldsymbol{x}})$. Hence, we do not differentiate the first dm basis functions separately for each element of the mesh but just once in the reference element. Elementwise differentiation is performed only for the last m' basis functions needed to compute the remaining of the divergence matrix, portion, $D^{(K)^{(2)}}_{m\times m'}$.

The second optimization pertains to the solution of the local matrix problem in Eq (3.4). Since the mass matrix is identity, the degrees of freedom matrix $Q^{\mu,(K)}_{n\times(d+1)m'}$ can be easily eliminated to obtain the following matrix problem just for the degrees of freedom $U^{\mu,(K)}_{m\times((d+1)m')}$:

$$L^{(K)}_{m\times m} U^{\mu,(K)}_{m\times((d+1)m')} = D^{(K)}_{m\times n} b^{Q^\mu,(K)}_{n\times((d+1)m')}. \tag{3.7}$$

Here, $L^{(K)}_{m\times m}$ is the Laplacian matrix and is equal to:

$$L^{(K)}_{m\times m} = D^{(K)}_{m\times n} {D^{(K)}_{m\times n}}^T.$$

To solve this problem, the matrix $L^{(K)}_{m\times m}$ is first factored using the (dense) Cholesky factorization method and the computed factor is used to obtain $U^{\mu,(K)}_{m\times((d+1)m')}$.

Similarly, the above basis functions convert the local problem that depends on f (Eq (2.11)) to the following matrix problem:

$$\begin{bmatrix} I_{n\times n}|K| & -{D^{(K)}_{m\times n}}^T \\ D^{(K)}_{m\times n} & 0_{m\times m} \end{bmatrix} \begin{bmatrix} Q^{f,(K)}_{n\times 1} \\ U^{f,(K)}_{m\times 1} \end{bmatrix} = \begin{bmatrix} 0_{n\times 1} \\ |K|P^{(K)} f_{m\times 1} \end{bmatrix}.$$

Here, $Q^{f,(K)}_{n\times 1}$ and $U^{f,(K)}_{m\times 1}$ are the degrees of freedom vector corresponding to the mapping $\boldsymbol{Q}_f$ and U_f, respectively.Here, $P^{(K)} f_{m\times 1}$ is the degree of freedom vector obtained by L^2-projection of f onto W using the above basis functions. Similar to the local problem that depended on μ, the above matrix problem is also solved by eliminating the degrees of freedom corresponding to $Q^{f,(K)}_{n\times 1}$ and then using the above computed Cholesky factor to obtain $U^{f,(K)}_{m\times 1}$.

3.1.3. Global problem

Using the above computed local problem solutions, the element matrix $A^{(K)}_{((d+1)m')\times((d+1)m')}$ and element vector $b^{(K)}_{((d+1)m')\times 1}$ are computed in each element K as:

$$\begin{aligned} A^{(K)}_{((d+1)m')\times((d+1)m')} &= {Q^{\mu,(K)}_{n\times(d+1)m'}}^T Q^{\mu,(K)}_{n\times(d+1)m'}|K|, \text{ and} \\ b^{(K)}_{((d+1)m')\times 1} &= {U^{\mu,(K)}_{m\times(d+1)m'}}^T P^{(K)} f_{m\times 1}|K|, \end{aligned} \tag{3.8}$$

respectively. These element matrices and vectors are assembled together and the degrees of freedom of $\hat{u}_h$ corresponding to the Dirichlet boundary faces are statically condensed to obtain the global matrix problem:

$$A_{m'n_F\times m'n_F} \hat{\mathrm{u}}_{m'n_F\times 1} = b_{m'n_F\times 1}. \tag{3.9}$$

Here, n_F is the total number of faces of the mesh minus the Dirichlet boundary faces. $A_{m'n_F\times m'n_F}$ is the global left-hand side matrix. $b_{m'n_F\times 1}$ is the global right-hand side vector. $\hat{\mathrm{u}}_{m'n_F\times 1}$ is the degree of freedom vector corresponding to $\hat{u}_h$. The above global matrix problem is solved using the sparse Cholesky factorization method.

3.2. Stab-1-HRT (new)

The details of the new Stab-1-HRT implementation are given below. For general HDG methods, see [3, 11].

3.2.1. Basis

In this implementation, for the space $\boldsymbol{V}_a(K)$ in each element K, we use the following orthonormal basis functions:

$$\boldsymbol{\varphi}^{(K)}_1, \ldots, \boldsymbol{\varphi}^{(K)}_{dm}.$$

The above functions $\boldsymbol{\varphi}_i^{(K)}$ are the same ones that were defined in Section 3.1.1. For the space $\boldsymbol{V}_s(K)$, we use:

$$\boldsymbol{\varphi}_{dm+1}^{(K)}, \ldots, \boldsymbol{\varphi}_n^{(K)}.$$

These are the remaining m' basis functions that were defined in Section 3.1.1. For the remaining spaces W and M_h, we use the same basis functions as that used in Section 3.1.1.

3.2.2. Local problem

In this implementation, we have to compute the stabilization mapping $\boldsymbol{L}_{\boldsymbol{V}_s}$ (given in Eq (2.6)). Using the above basis functions for $\boldsymbol{V}_s(K)$, the matrix form of this mapping becomes:

$$L^{s,(K)}_{m'\times(d+1)m'} = \frac{1}{|K|} b^{s,(K)}_{m'\times(d+1)m'}.$$

Here, $b^{s,(K)}_{m'\times(d+1)m'}$ is the right-hand side of the stabilization mapping given by:

$$\left[b^{s,(K)}_{m'\times((d+1)m')}\right]_{i,(j-1)m'+r} = \int_{F_j} \psi_r^{(F_j)} \boldsymbol{\varphi}^{(K)}_{dm+i} \cdot \boldsymbol{n}\, \mathrm{d}\Gamma.$$

Using the above basis functions for $\boldsymbol{V}_a$ and the above matrix form of the stabilization mapping, the local problem that depends on μ (Eq (2.10)) becomes the following local matrix problem:

$$\begin{bmatrix} I_{n\times n}|K| & -D^{(K)}_{m\times dm}{}^T \\ D^{(K)}_{m\times dm} & M^{s,(K)}_{m\times m} \end{bmatrix} \begin{bmatrix} Q^{\mu,(K)}_{dm\times(d+1)m'} \\ U^{\mu,(K)}_{m\times(d+1)m'} \end{bmatrix} = \begin{bmatrix} -b^{Q^\mu,(K)}_{dm\times((d+1)m')} \\ b^{U^\mu,(K)}_{m\times((d+1)m')} \end{bmatrix}. \tag{3.10}$$

Here, $M^{s,(K)}_{m\times m}$ is the mass matrix that arises from the equivalent stabilization. It relates to the stabilization mapping matrix as:

$$M^{s,(K)}_{m\times m} = |K| \left[L^{s,(K)}_{m'\times(d+1)m'} B^{(K)}_{(d+1)m'\times m}\right]^T \left[L^{s,(K)}_{m'\times(d+1)m'} B^{(K)}_{(d+1)m'\times m}\right],$$

where $B^{(K)}_{(d+1)m'\times m}$ is the linear mapping from the degrees of freedom of u to that of μ in element K. $D^{(K)}_{m\times n}$ is the same divergence matrix that was defined in Section 3.1.2. The right-hand side matrix $b^{U^\mu,(K)}_{m\times((d+1)m')}$ is given by:

$$b^{U^\mu,(K)}_{m\times((d+1)m')} = |K| \left[L^{s,(K)}_{m'\times(d+1)m'} B^{(K)}_{(d+1)m'\times m}\right]^T L^{s,(K)}_{m'\times(d+1)m'}.$$

Observe that the divergence matrix $D^{(K)}_{m\times dm}$ in Eq (3.10) has the following form:

$$D^{(K)}_{m\times dm} = \begin{bmatrix} D^{(K)^{(1)}}_{(m-m')\times dm} \\ 0_{m'\times dm} \end{bmatrix},$$

where the portion $D^{(K)^{(1)}}_{(m-m')\times dm}$ is the same as that defined in Eq (3.6). Observe that the portion $D^{(K)^{(2)}}_{n\times m'}$ that was present in Eq (3.6) is not present above. Hence, we only compute the portion $D^{(K)^{(1)}}_{(m-m')\times dm}$ using the optimized implementation described in Section 3.1.2. We never compute $D^{(K)^{(2)}}_{n\times m'}$ that required differentiation of the last m' basis functions separately in each element of the mesh. This yields in

considerable reduction in the time taken to compute the individual local problems and is demonstrated by the numerical experiments in Section 4.

Similar to the Usual-HRT implementation, the degree of freedom matrix $Q^{\mu,(K)}_{dm\times(d+1)m'}$ is eliminated to obtain the following equation for $U^{\mu,(K)}_{m\times(d+1)m'}$:

$$L^{(K),1}_{m\times m} U^{\mu,(K)}_{m\times(d+1)m'} = D^{(K)}_{m\times n} b^{Q^\mu,(K)}_{dm\times((d+1)m')} + b^{U^\mu,(K)}_{m\times((d+1)m')}. \tag{3.11}$$

Here, the Laplacian matrix $L^{(K),1}_{m\times m}$ is given by:

$$L^{(K),1}_{m\times m} = D^{(K)}_{m\times dm} {D^{(K)}_{m\times dm}}^T + M^{s,(K)}_{m\times m}.$$

The Laplacian is first factored using the Cholesky factorization method and the computed factors are used to compute $U^{\mu,(K)}_{m\times(d+1)m'}$.

Similarly, the chosen basis functions convert the local problem that depends on f to the following matrix problem:

$$\begin{bmatrix} I_{n\times n}|K| & -{D^{(K)}_{m\times dm}}^T \\ D^{(K)}_{m\times dm} & M^{s,(K)}_{m\times m} \end{bmatrix} \begin{bmatrix} Q^{f,(K)}_{dm\times(d+1)m'} \\ U^{f,(K)}_{m\times(d+1)m'} \end{bmatrix} = \begin{bmatrix} 0_{dm\times 1} \\ |K|P^{(K)} f_{m\times 1} \end{bmatrix}.$$

The above matrix problem is solved similar to the above local problem that depended only on μ. The same Cholesky factors are used for its solution.

3.2.3. Global problem

In this implementation, the element matrix and vector are computed as:

$$\begin{aligned} A^{(K)}_{((d+1)m')\times((d+1)m')} &= {Q^{\mu,(K)}_{dm\times(d+1)m'}}^T Q^{\mu,(K)}_{dm\times(d+1)m'}|K|, \text{ and} \\ b^{(K)}_{((d+1)m')\times 1} &= {U^{\mu,(K)}_{m\times(d+1)m'}}^T P^{(K)} f_{m\times 1}|K|. \end{aligned} \tag{3.12}$$

Then, similar to the Usual-HRT implementation, the above element matrices and element vectors are assembled to form the global problem:

$$A_{m'n_F\times m'n_F} \hat{\mathrm{u}}_{m'n_F\times 1} = b_{m'n_F\times 1}.$$

This global problem is solved using the sparse Cholesky factorization.

3.3. Stab-2-HRT (new)

The details of the new Stab-2-HRT implementation are given below.

3.3.1. Basis

In this implementation, for the space $\boldsymbol{V}_a(K)$ in each element K, we use the following orthonormal basis functions:

$$\boldsymbol{\varphi}^{(K)}_1, \ldots, \boldsymbol{\varphi}^{(K)}_{d(m-m'))},$$

where the above functions $\boldsymbol{\varphi}^{(K)}_i$ are the same functions defined in Section 3.1.1. For the space $\boldsymbol{V}_s(K)$, we use the remaining $(d+1)m'$ functions as the basis:

$$\boldsymbol{\varphi}^{(K)}_{d(m-m')+1}, \ldots, \boldsymbol{\varphi}_n.$$

For the spaces W and M_h, we use the same basis functions that were used in Section 3.1.1.

3.3.2. Local problem

The local problem solution for this implementation is very similar to that of the Stab-1-HRT implementation. The matrix form of the stabilization mapping $\boldsymbol{L}_{\boldsymbol{V}_s}$ is:

$$L^{s,(K)}_{(d+1)m'\times(d+1)m'} = \frac{1}{|K|} b^{s,(K)}_{(d+1)m'\times(d+1)m'},$$

where

$$\left[b^{s,(K)}_{(d+1)m'\times((d+1)m')}\right]_{i,(j-1)m'+r} = \int_{F_j} \psi_r^{(F_j)} \boldsymbol{\varphi}^{(K)}_{d(m-m')+i} \cdot \boldsymbol{n}\, \mathrm{d}\Gamma.$$

The local problem that depends on μ (Eq (2.10)) becomes the following matrix problem:

$$\begin{bmatrix} I_{d(m-m')\times d(m-m')}|K| & -D^{(K)}_{m\times d(m-m')}{}^T \\ D^{(K)}_{m\times d(m-m')} & M^{(K),(L)}_{m\times m} \end{bmatrix} \begin{bmatrix} Q^{\mu,(K)}_{d(m-m')\times(d+1)m'} \\ U^{\mu,(K)}_{m\times(d+1)m'} \end{bmatrix} = \begin{bmatrix} -b^{Q^\mu,(K)}_{dm\times((d+1)m')} \\ b^{U^\mu,(K)}_{m\times((d+1)m')} \end{bmatrix}, \tag{3.13}$$

where the mass-matrix from the stabilization term is:

$$M^{s,(K)}_{m\times m} = |K| \left[L^{s,(K)}_{(d+1)m'\times(d+1)m'} B^{(K)}_{(d+1)m'\times m}\right]^T \left[L^{s,(K)}_{(d+1)m'\times(d+1)m'} B^{(K)}_{(d+1)m'\times m}\right].$$

Then, similar to the Stab-1-HRT implementation, the above local matrix problem is solved using the Cholesky factorization methodology after eliminating the matrix $Q^{\mu,(K)}_{d(m-m')\times(d+1)m'}$.

The local problem that depends on f becomes:

$$\begin{bmatrix} I_{d(m-m')\times d(m-m')}|K| & -D^{(K)}_{m\times d(m-m')}{}^T \\ D^{(K)}_{m\times d(m-m')} & M^{(K),(L)}_{m\times m} \end{bmatrix} \begin{bmatrix} Q^{f,(K)}_{d(m-m')\times 1} \\ U^{f,(K)}_{m\times 1} \end{bmatrix} = \begin{bmatrix} -0_{dm\times 1} \\ |K| P^{(K)} f_{m\times 1} \end{bmatrix}. \tag{3.14}$$

The above matrix problem is also solved using the same Cholesky factors that were computed while solving the above local problem that depended on μ alone.

3.3.3. Global problem

In this implementation, the element matrix and vector are computed as:

$$\begin{aligned} A^{(K)}_{((d+1)m')\times((d+1)m')} =& |K| \Bigg(Q^{\mu,(K)}_{dm\times(d+1)m'}{}^T Q^{\mu,(K)}_{dm\times(d+1)m'} + \left[L^{s,(K)}_{(d+1)m'\times(d+1)m'} \left(B^{(K)}_{(d+1)m'\times m} U^{\mu,(K)}_{m\times(d+1)m'} - I_{(d+1)m'\times(d+1)m'}\right)\right]^T \\ & \left[L^{s,(K)}_{(d+1)m'\times(d+1)m'} \left(B^{(K)}_{(d+1)m'\times m} U^{\mu,(K)}_{m\times(d+1)m'} - I_{(d+1)m'\times(d+1)m'}\right)\right] \Bigg), \\ b^{(K)}_{((d+1)m')\times 1} =& U^{\mu,(K)}_{m\times(d+1)m'}{}^T P^{(K)} f_{m\times 1} |K|. \end{aligned}$$

Then, similar to the Stab-1-HRT implementation, the above element matrices and element vectors are assembled to form the global problem

$$A_{m'n_F\times m'n_F} \hat{\mathrm{u}}_{m'n_F\times 1} = b_{m'n_F\times 1}$$

and this global problem is solved using the sparse Cholesky factorization.

4. Numerical results

We present results comparing the three implementations. We consider the Poisson problem with $f = 8\pi^2 \sin(2\pi x_1)\sin(2\pi x_2)$ in the domain $(0,1)^2$. The domain is first split into 16 uniform quadrilateral elements along each direction. Each quadrilateral element is further split into two triangular elements. This leads to a uniform mesh of 512 triangular elements. Polynomial degrees 1 to 20 are considered. Zero-Dirichlet boundary condition is imposed on all the four sides of the domain. All these implementations were first validated to make sure they yield identical solutions.

All numerical experiments were performed in MATLAB. We use a workstation with Intel(R) Core(TM) i7-8700 processor. The processor has six cores and hyperthreading. We also note that MATLAB uses the multi-threaded MKL BLAS backend for certain matrix and vector manipulations. Hence, there is some inherent parallelism in our implementation. For the global problem solution, we use the sparse direct solver available in MATLAB. MATLAB uses the sparse multi-threaded Cholesky solver CHOLMOD [2] for our symmteric positive definite problem when performing `x=A\b`.

Tables 2 and 3 show the time consumed by the one-time operations (that are performed just once in the reference element), the local problem solution in all the elements, global problem solution and the total solution time. The breakdown of the time taken by the different components of the local problem solution are shown in Tables 4 and 5. The percentage benefit of the new implementations compared to the Usual-HRT implementation is shown in Table 6.

Table 2. Comparison of the time (in seconds) for one-time operations and local problem solutions.

k	One-time operations			Local problem solutions		
	Usual-HRT	Stab-1-HRT	Stab-2-HRT	Usual-HRT	Stab-1-HRT	Stab-2-HRT
1	8.54E-03	4.66E-03	6.14E-03	1.07E-01	8.29E-02	9.56E-02
2	3.04E-03	2.11E-03	2.19E-03	1.65E-01	1.17E-01	1.26E-01
3	3.46E-03	8.79E-04	8.62E-04	2.36E-01	1.78E-01	1.89E-01
4	1.20E-03	1.20E-03	1.25E-03	3.59E-01	2.76E-01	2.90E-01
5	1.77E-03	1.70E-03	1.83E-03	5.31E-01	4.23E-01	4.40E-01
6	2.43E-03	2.52E-03	2.45E-03	8.82E-01	7.23E-01	6.70E-01
7	3.37E-03	3.36E-03	3.39E-03	1.25E+00	1.05E+00	1.07E+00
8	4.93E-03	5.01E-03	4.92E-03	1.74E+00	1.45E+00	1.47E+00
9	6.85E-03	6.85E-03	7.16E-03	2.28E+00	1.99E+00	2.02E+00
10	9.15E-03	9.17E-03	9.00E-03	3.07E+00	2.62E+00	2.68E+00
11	1.21E-02	1.26E-02	1.25E-02	3.98E+00	3.43E+00	3.47E+00
12	1.70E-02	1.72E-02	1.71E-02	5.13E+00	4.41E+00	4.59E+00
13	2.33E-02	2.35E-02	2.37E-02	6.56E+00	5.83E+00	5.84E+00
14	3.01E-02	2.99E-02	3.02E-02	8.27E+00	7.26E+00	7.39E+00
15	4.01E-02	3.99E-02	4.04E-02	1.06E+01	9.26E+00	9.34E+00
16	5.18E-02	5.12E-02	5.46E-02	1.30E+01	1.14E+01	1.16E+01
17	7.16E-02	7.28E-02	7.09E-02	1.60E+01	1.40E+01	1.41E+01
18	9.03E-02	8.97E-02	9.03E-02	1.95E+01	1.71E+01	1.73E+01
19	1.25E-01	1.25E-01	1.26E-01	2.51E+01	2.07E+01	2.09E+01
20	1.68E-01	1.63E-01	1.66E-01	2.99E+01	2.48E+01	2.51E+01

Table 3. Comparison of time (in seconds) for global problem solution and total solution.

k	Global problem solution			Total solution		
	Usual-HRT	Stab-1-HRT	Stab-2-HRT	Usual-HRT	Stab-1-HRT	Stab-2-HRT
1	3.02E-02	3.00E-02	3.12E-02	1.46E-01	1.18E-01	1.33E-01
2	2.76E-02	2.66E-02	3.06E-02	1.96E-01	1.46E-01	1.59E-01
3	2.70E-02	2.65E-02	3.04E-02	2.66E-01	2.06E-01	2.20E-01
4	3.03E-02	3.04E-02	3.41E-02	3.90E-01	3.08E-01	3.26E-01
5	3.44E-02	3.35E-02	3.77E-02	5.67E-01	4.58E-01	4.79E-01
6	3.96E-02	3.92E-02	4.19E-02	9.24E-01	7.65E-01	7.14E-01
7	4.66E-02	4.65E-02	5.08E-02	1.30E+00	1.10E+00	1.13E+00
8	5.24E-02	5.07E-02	5.54E-02	1.80E+00	1.51E+00	1.53E+00
9	5.81E-02	5.84E-02	6.35E-02	2.35E+00	2.06E+00	2.09E+00
10	6.54E-02	6.52E-02	6.99E-02	3.15E+00	2.69E+00	2.76E+00
11	7.22E-02	7.10E-02	7.73E-02	4.06E+00	3.51E+00	3.56E+00
12	8.00E-02	7.92E-02	8.51E-02	5.22E+00	4.51E+00	4.69E+00
13	9.06E-02	8.95E-02	9.55E-02	6.68E+00	5.94E+00	5.96E+00
14	1.00E-01	1.01E-01	1.05E-01	8.40E+00	7.39E+00	7.53E+00
15	1.12E-01	1.10E-01	1.16E-01	1.07E+01	9.41E+00	9.50E+00
16	1.25E-01	1.23E-01	1.29E-01	1.32E+01	1.16E+01	1.17E+01
17	1.39E-01	1.37E-01	1.42E-01	1.62E+01	1.42E+01	1.43E+01
18	1.52E-01	1.50E-01	1.55E-01	1.97E+01	1.73E+01	1.75E+01
19	1.62E-01	1.60E-01	1.69E-01	2.54E+01	2.09E+01	2.12E+01
20	1.81E-01	1.78E-01	1.85E-01	3.03E+01	2.51E+01	2.55E+01

Table 4. Comparison of time (in seconds) for computation of the additional RT basis functions and their contribution to the divergence matrix for all the elements in the mesh.

k	Additional RT basis			Div. matrix of additional RT basis		
	Usual-HRT	Stab-1-HRT	Stab-2-HRT	Usual-HRT	Stab-1-HRT	Stab-2-HRT
1	2.50E-02	2.52E-02	2.59E-02	2.74E-02	-	-
2	5.57E-02	5.35E-02	5.35E-02	4.43E-02	-	-
3	1.05E-01	1.05E-01	1.04E-01	5.99E-02	-	-
4	1.88E-01	1.88E-01	1.88E-01	8.57E-02	-	-
5	3.16E-01	3.18E-01	3.17E-01	1.15E-01	-	-
6	5.49E-01	5.48E-01	5.23E-01	1.61E-01	-	-
7	8.37E-01	8.37E-01	8.36E-01	2.08E-01	-	-
8	1.23E+00	1.21E+00	1.21E+00	2.72E-01	-	-
9	1.68E+00	1.71E+00	1.70E+00	3.35E-01	-	-
10	2.30E+00	2.29E+00	2.30E+00	4.29E-01	-	-
11	3.05E+00	3.05E+00	3.05E+00	5.31E-01	-	-
12	3.98E+00	3.97E+00	4.03E+00	6.83E-01	-	-
13	5.12E+00	5.24E+00	5.16E+00	8.89E-01	-	-
14	6.52E+00	6.57E+00	6.62E+00	1.08E+00	-	-
15	8.42E+00	8.47E+00	8.46E+00	1.37E+00	-	-
16	1.04E+01	1.05E+01	1.05E+01	1.61E+00	-	-
17	1.28E+01	1.29E+01	1.29E+01	2.05E+00	-	-
18	1.58E+01	1.58E+01	1.58E+01	2.41E+00	-	-
19	1.93E+01	1.92E+01	1.92E+01	4.31E+00	-	-
20	2.32E+01	2.31E+01	2.32E+01	5.04E+00	-	-

Table 5. Comparison of time (in seconds) for the solution of the local matrix problems using the Cholesky decomposition for each element in the mesh.

k	Local matrix problem		
	Usual-HRT	Stab-1-HRT	Stab-2-HRT
1	3.47E-02	3.88E-02	4.98E-02
2	4.04E-02	4.20E-02	5.07E-02
3	4.27E-02	4.78E-02	5.86E-02
4	4.93E-02	5.42E-02	6.88E-02
5	5.69E-02	6.34E-02	8.10E-02
6	1.15E-01	1.21E-01	9.54E-02
7	1.37E-01	1.45E-01	1.68E-01
8	1.58E-01	1.58E-01	1.85E-01
9	1.65E-01	1.82E-01	2.21E-01
10	2.06E-01	2.03E-01	2.54E-01
11	2.36E-01	2.34E-01	2.82E-01
12	2.74E-01	2.69E-01	3.40E-01
13	3.27E-01	3.70E-01	4.62E-01
14	3.95E-01	4.26E-01	5.03E-01
15	4.55E-01	4.87E-01	5.74E-01
16	5.54E-01	5.44E-01	6.88E-01
17	6.07E-01	6.41E-01	7.81E-01
18	7.19E-01	7.51E-01	9.29E-01
19	8.61E-01	9.04E-01	1.12E+00
20	1.02E+00	1.05E+00	1.31E+00

Table 6. Percentage performance benefit of the Stab-1-HRT and Stab-2-HRT implementations over the Usual-HRT implementation.

k	Local problem solution		Total solution	
	Stab-1-HRT	Stab-2-HRT	Stab-1-HRT	Stab-2-HRT
1	22.65	10.90	19.40	8.93
2	28.84	23.40	25.32	18.63
3	24.35	20.03	22.72	17.46
4	23.04	19.06	21.16	16.55
5	20.34	17.23	19.21	15.53
6	17.94	23.99	17.15	22.64
7	16.20	14.47	15.58	13.59
8	16.85	15.67	16.41	15.01
9	12.77	11.44	12.40	10.88
10	14.87	12.83	14.53	12.38
11	13.85	12.71	13.58	12.32
12	13.95	10.49	13.70	10.20
13	11.15	11.01	10.97	10.74
14	12.21	10.64	12.01	10.42
15	12.31	11.56	12.16	11.36
16	12.50	11.24	12.36	11.04
17	12.40	11.64	12.24	11.47
18	12.20	11.08	12.07	10.94
19	17.59	16.47	17.40	16.25
20	17.15	16.01	16.98	15.82

The new implementations Stab-1-HRT and Stab-2-HRT are faster than the Usual-HRT implementation for all the polynomial degrees. From Table 6, we observe that they are 10-20% faster depending on the polynomial degree. Stab-1-HRT is slightly faster than Stab-2-HRT for nearly all polynomial degrees (except degree six).

The local problem solutions consume the majority of the total solution time. These local problem solution solutions are faster for the new Stab-1-HRT and Stab-2-HRT implementations compared to the usual-HRT implementations. From Tables 4 and 5 (which show the breakdown of the local problem solutions), we observe that this performance benefit is essentially because in the new Stab-1-HRT and Stab-2-HRT implementations, we need not compute the derivative of the additional RT basis functions, i.e., $\boldsymbol{\varphi}_{dm+1}, \ldots, \boldsymbol{\varphi}_n$, and their contribution to the divergence matrix of the local problems. However, in the usual-HRT implementation, these derivatives and their contribution to the divergence matrix must be computed. In the Stab-1-HRT and Stab-2-HRT implementations, there is an overhead associated with the computation of the stabilization mappings. However, the benefits from not computing the derivative of the additional RT basis functions and their contribution to the divergence matrix significantly outweights this overhead. Hence, the new implementations of the hybridized RT method yield significant (10–20%) performance benefit compared to the usual implementation.

5. Conclusions and ongoing work

As pointed out in [5], although the choice of the local function space $\boldsymbol{V}_a(K)$ is not unique, as we have also seen here, the *smallest* of these local spaces, $\nabla W(K) = \nabla P_k(K)$, is actually unique. It remains to be seen it we still retain an advantage over the usual implementation of the hybridized RT method for this choice. This constitutes the subject of ongoing work.

The present paper is currently being extended along the following directions:

(1) As pointed out in the Introduction, the choice of the RT method as the mixed method is by no means restrictive, as the approach proposed here can also be applied to any other mixed method for polyhedral meshes, for e.g., those defined using M-decompositions in [6, 7, 9] and using new commuting diagrams [8] (see also the review [10]) and the references therein). The space $\boldsymbol{V}(K)$ in these methods also have the form $[P_k(K)]^d \oplus \boldsymbol{V}_{\text{fill}}$ and the $W(K)$ is still equal to $P_k(K)$. Similar to the implementation presented here, the effect of $\boldsymbol{V}_{\text{fill}}$ (and also a portion of $[P_k(K)]^d$) can be encapsulated within a mapping $\boldsymbol{L}_{\boldsymbol{V}_s}$ defined in a similar way.

In our new implementations, we have not defined the basis functions via the Piola Transform. On the other hand, the reference unit simplex is only used to define the orthonormal polynomial basis functions for $[P_k(K)]^d$ and to compute the divergence matrix via chain rule. In the case of polytopal elements, we note that the polytopal mixed methods in [6–10] do not make use of a reference element. Hence, it would be a different baseline to compare our implementations to.

Furthermore, the application of this approach to other general HDG methods can be carried out very easily, as we are going to show elsewhere.

(2) In $d = 3$ dimensions, we expect similar or better speedups. For e.g., for simplexes and Stab-2-HRT, the reduction in the computational effort for the local problem is approximately proportional to $dim(\boldsymbol{V}_s^{(2)}(K))$ which is $(d+1)\,C_{d-1}^{k+d-1}$. The factor

$$\frac{\dim(\boldsymbol{V}_s^{(2)}(K))}{\dim(\boldsymbol{V}(K))} = \frac{(d+1)\,C_{d-1}^{k+d-1}}{(dC_d^{k+d} + C_{d-1}^{k+d-1})} = \frac{d+1}{d+k}.$$

For large k, this factor is around $4/3$ times larger for $d = 3$ compared to $d = 2$. Hence, we expect similar or bigger speedups for three-dimensional problems. Extension to $d = 3$ is part of our ongoing work.

(3) Finally, note that, although we have exploited the fact that the tensor-valued function c is the identity for our model second-order elliptic problem, it is easy to extend what has been done to a general elliptic problem. This is also part of our ongoing work.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

Bernardo Cockburn's research was supported in part by the Advanced Computational Center for Entry Systems Simulation (ACCESS) through NASA grant 80NSSC21K1117.

Conflict of interest

The authors declare no conflicts of interest.

References

1. D. N. Arnold, F. Brezzi, Mixed and nonconforming finite element methods: implementation, postprocessing and error estimates, *RAIRO Modél. Math. Anal. Numér.*, **19** (1985), 7–32. https://doi.org/10.1051/m2an/1985190100071
2. Y. Chen, T. A. Davis, W. W. Hager, S. Rajamanickam, Algorithm 887: CHOLMOD, supernodal sparse Cholesky factorization and update/downdate, *ACM Trans. Math. Software*, **35** (2008), 1–14. https://doi.org/10.1145/1391989.1391995
3. B. Cockburn, Static condensation, hybridization, and the devising of the HDG methods, In: G. Barrenechea, F. Brezzi, A. Cangiani, E. Georgoulis, *Building bridges: connections and challenges in modern approaches to numerical partial differential equations*, Cham: Springer, **114** (2016), 129–177. https://doi.org/10.1007/978-3-319-41640-3_5
4. B. Cockburn, Discontinuous Galerkin methods for computational fluid dynamics, In: E. Stein, R. de Borst, T. J. R. Hughes, *Encyclopedia of computational mechanics*, 2 Eds., John Wiley & Sons, Ltd., **5** (2018), 141–203. https://doi.org/10.1002/9781119176817.ecm2053
5. B. Cockburn, Hybridizable discontinuous Galerkin methods for second-order elliptic problems: overview, a new result and open problems, *Japan J. Indust. Appl. Math.*, **40** (2023), 1637–1676. https://doi.org/10.1007/s13160-023-00603-9
6. B. Cockburn, G. Fu, Superconvergence by M-decompositions. Part II: Construction of two-dimensional finite elements, *ESAIM Math. Model. Numer. Anal.*, **51** (2017), 165–186. https://doi.org/10.1051/m2an/2016016

7. B. Cockburn, G. Fu, Superconvergence by M-decompositions. Part III: Construction of three-dimensional finite elements, *ESAIM Math. Model. Numer. Anal.*, **51** (2017), 365–398. https://doi.org/10.1051/m2an/2016023
8. B. Cockburn, G. Fu, A systematic construction of finite element commuting exact sequences, *SIAM J. Numer. Anal.*, **55** (2017), 1650–1688. https://doi.org/10.1137/16M1073352
9. B. Cockburn, G. Fu, F. J. Sayas, Superconvergence by M-decompositions. Part I: General theory for HDG methods for diffusion, *Math. Comp.*, **86** (2017), 1609–1641. https://doi.org/10.1090/mcom/3140
10. B. Cockburn, G. Fu, K. Shi, An introduction to the theory of M-decompositions, In: D. Di Pietro, A. Ern, L. Formaggia, *Numerical methods for PDEs*, Cham: Springer, **15** (2018), 5–29. https://doi.org/10.1007/978-3-319-94676-4_2
11. B. Cockburn, J. Gopalakrishnan, R. Lazarov, Unified hybridization of discontinuous Galerkin, mixed and continuous Galerkin methods for second order elliptic problems, *SIAM J. Numer. Anal.*, **47** (2009), 1319–1365. https://doi.org/10.1137/070706616
12. M. Dubiner, Spectral methods on triangles and other domains, *J. Sci. Comput.*, **6** (1991), 345–390. https://doi.org/10.1007/BF01060030
13. P. A. Raviart, J. M. Thomas, A mixed finite element method for second order elliptic problems, In: I. Galligani, E. Magenes, *Mathematical aspects of finite element method*, Lecture Notes in Mathematics, Springer, **606** (1977), 292–315. https://doi.org/10.1007/BFb0064470

AIMS Press