Journal of Computational Physics 231 (2012) 5955–5988

Contents lists available at SciVerse ScienceDirect

Journal of Computational Physics

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

Hybridizable discontinuous Galerkin methods for partial differential equations in continuum mechanics

N. C. Nguyen *, J. Peraire

Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

ARTICLE INFO

Article history:
Received 16 January 2011
Received in revised form 26 January 2012
Accepted 20 February 2012
Available online 25 April 2012

Keywords:
Finite element methods
Discontinuous Galerkin methods
Hybrid/mixed methods
Augmented lagrangian
Partial differential equations
Stokes equations
Euler equations
Navier–Stokes equations
Elasticity
Hyperelasticity
Elastodynamics
Fluid mechanics
Solid mechanics
Continuum mechanics

ABSTRACT

We present hybridizable discontinuous Galerkin methods for solving steady and time-dependent partial differential equations (PDEs) in continuum mechanics. The essential ingredients are a local Galerkin projection of the underlying PDEs at the element level onto spaces of polynomials of degree k to parametrize the numerical solution in terms of the numerical trace; a judicious choice of the numerical flux to provide stability and consistency; and a global jump condition that enforces the continuity of the numerical flux to arrive at a global weak formulation in terms of the numerical trace. The HDG methods are fully implicit, high-order accurate and endowed with several unique features which distinguish themselves from other discontinuous Galerkin methods. First, they reduce the globally coupled unknowns to the approximate trace of the solution on element boundaries, thereby leading to a significant reduction in the degrees of freedom. Second, they provide, for smooth viscous-dominated problems, approximations of all the variables which converge with the optimal order of $k+1$ in the L^2-norm. Third, they possess some superconvergence properties that allow us to define inexpensive element-by-element postprocessing procedures to compute a new approximate solution which may converge with higher order than the original solution. And fourth, they allow for a novel and systematic way for imposing boundary conditions for the total stress, viscous stress, vorticity and pressure which are not naturally associated with the weak formulation of the methods. In addition, they possess other interesting properties for specific problems. Their approximate solution can be postprocessed to yield an exactly divergence-free and $\boldsymbol{H}(\mathrm{div})$-conforming velocity field for incompressible flows. They do not exhibit volumetric locking for nearly incompressible solids. We provide extensive numerical results to illustrate their distinct characteristics and compare their performance with that of continuous Galerkin methods.

1. Introduction

Partial differential equations (PDEs) in continuum mechanics have attracted considerable interest from physicists and mathematicians over centuries. This interest stems from the curiosity of real-world phenomena and the attempt to model them by using physical balance laws and mathematical equations. Except for very few cases in which they admit analytical solutions, PDEs are numerically solved in most physical problems of interest. The development of numerical methods for PDEs and their applications have given birth to two new emerging fields in engineering and science, namely, computational mechanics and computational mathematics. Indeed, numerical methods for PDE have played and will continue to play an important role in aiding our understanding of complex physical phenomena without the need of performing experiments.

* Corresponding author.
E-mail address: cuongng@mit.edu (N. C. Nguyen).

0021-9991/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jcp.2012.02.033

They have also helped and will continue to help us to design airplanes, cars, transportation systems, telecommunication systems, biomedical devices, among other important components or systems.

Over the past decades, the finite element method has been among the most popular techniques for numerically solving PDEs. There are several discretization strategies within the finite element method. They include continuous Galerkin methods, SUPG methods [31], least-squares finite element methods [32], spectral element methods [52], mixed finite element methods [8], extended finite element methods [40], discontinuous Galerkin methods [58], discontinuous Petrov–Galerkin methods [20], and other finite element methods. Each method has its own strengths and weaknesses that make it ideal for some applications, but not the best choice for others. As a result, finite element practitioners typically make a choice of methods based on the particular problem they want to solve. For instance, discontinuous Galerkin methods mainly find their domain of applications in computational fluid dynamics, while continuous Galerkin methods are widely used to solve problems in solid and fluid mechanics.

In recent years, considerable attention has been turned to discontinuous Galerkin (DG) methods for the numerical solution of PDEs [3,5,6,18,19,23,28,29,33,38,41,54,55,58,61]. DG methods possess several attractive features. In particular, they are flexible for complicated geometry, locally conservative, high-order accurate, highly parallelizable, low dissipative, and more stable than continuous Galerkin (CG) methods for convection-dominated problems. However, in spite of all these advantages, DG methods have not yet made a more significant impact for practical applications as we would hope. This is largely due to the main criticism that DG methods are computationally extensive since they have too many degrees of freedom due to nodal duplication at the element boundary interfaces. More specifically, assuming about six linear tetrahedral elements per node, the number of unknowns in a DG system would approximately 24 times the number of unknowns in the corresponding CG system. For cubic tetrahedral elements, the degrees-of-freedom ratio between DG and CG reduces to approximately 5. The high computational cost and memory storage are a major impediment to the widespread application of DG methods for real-world problems. Therefore, it would be highly desirable to develop new DG methods that have all the advantages of DG methods (analyzed in [3]), and yet are competitive with continuous Galerkin methods.

Recently, researchers in computational mechanics and mathematics have developed more efficient DG methods—the multiscale discontinuous Galerkin (MDG) method [30] and the embedded discontinuous Galerkin (EDG) method [25]—as a timely response to the above criticism of DG methods. Further development of the MDG method for the incompressible Navier–Stokes equations leads to the so-called Galerkin interface stabilisation (GIS) method [35]. The EDG methods for the compressible Euler and Navier–Stokes equations have been proposed in [53]. Both the MDG and EDG methods are devised to solve for the numerical trace of the solution as globally coupled unknowns. Therefore, the matrix system of these DG methods is similar to that of the statically condensed version of CG methods. However, both the MDG and EDG methods are not locally conservative and have the same convergence rates as CG methods.

More recently, a newer class of DG methods—the so-called hybridizable discontinuous Galerkin (HDG) method—is first introduced in [13] for elliptic problems. The HDG method for elliptic problems is then analyzed in [10,15,16] where it is shown that the HDG method has many common features with the Raviart–Thosmas (RT) mixed method [57] and the Brezzi–Douglas–Marini (BDM) mixed method [7]. In particular, the HDG method provides optimal convergence rates of order $k+1$ for all the approximate variables. Furthermore, element-by-element postprocessing is developed to increase the convergence rate of the approximate scalar variable to $k+2$ for $k \geqslant 1$. Shortly later, HDG methods have been further developed and analyzed for linear convection–diffusion problems [47], nonlinear convection–diffusion problems [11,48,65], Stokes problems [12,14,17,49], and incompressible Navier–Stokes equations [45,46,50]. A unique feature of these HDG methods is that the approximate velocity, pressure and velocity gradient converge with the optimal order $k+1$ in the L^2-norm for smooth diffusion-dominated incompressible flows. In such case, a local postprocessing scheme proposed in [14,45] can be used to obtain an exactly divergence-free, $\boldsymbol{H}(\mathrm{div})$-conforming velocity which converges with order $k+2$ for $k \geqslant 1$.

Although computational fluid dynamics has been their main domain of applications, DG methods have recently been introduced to computational solid mechanics [4,26,27,51,64]. However, most current DG methods are far from competitive with CG methods for solving problems in solid mechanics because they provide the same accuracy often at a significantly greater cost than CG methods. After all, most of problems in solid mechanics are not convection-dominated, as such the efficiency and applicability of many existing DG methods are limited to quite few problems. However, owing to their excellent convergence properties for diffusion-dominated problems and reduced degrees of freedom, HDG methods can be competitive with CG methods and thus can be relevant for solving a large number of problems in solid mechanics. In particular, HDG methods have been introduced for solving linear elasticity and shell problems [62] as well as linear acoustic and elastic wave equations [44]. It has been demonstrated for linear acoustic and elastic wave equations that HDG methods produce significantly more accurate solution than the standard CG method even for the same global degrees of freedom (see [44]).

In this paper, we further develop hybridizable discontinuous Galerkin methods for solving steady and time-dependent PDEs in continuum mechanics, with the objective of broadening the application of HDG methods in computational fluid dynamics and computational solid mechanics. The essential ingredients are a local Galerkin projection of the underlying PDEs at the element level onto spaces of polynomials of degree k to parametrize the numerical solution in terms of the numerical trace; a judicious choice of the numerical flux to provide stability and consistency; and a global jump condition that enforces the continuity of the numerical flux to arrive at a global weak formulation in terms of the numerical trace. The HDG methods are fully implicit, high-order accurate and endowed with several unique features which distinguish themselves from other discontinuous Galerkin methods. First, it reduces the globally coupled unknowns to the approximate trace of the solution on element boundaries, thereby leading to a significant reduction in the degrees of freedom. Second, it

provides, for smooth viscous-dominated problems, approximations of all the variables which converge with the optimal order of $k+1$ in the L^2-norm. Third, they possess some superconvergence properties that allow us to define element-by-element postprocessing procedures to compute a new approximate solution which may converge with higher order than the original solution. And fourth, they allow for a novel and systematic way for imposing boundary conditions for the total stress, viscous stress, vorticity and pressure which are not naturally associated with the weak formulation of the methods. In addition, they possess other interesting properties for specific problems. Their approximate solution can be postprocessed to yield an exactly divergence-free and $\boldsymbol{H}(\mathrm{div})$-conforming velocity field for incompressible flows. They do not exhibit volumetric locking for nearly incompressible solids.

However, it should be mentioned that the HDG method is not as efficient as other explicit DG methods [29,38] for explicit time discretization of time-dependent problems. This is because, unlike explicit DG schemes, the HDG method does not result in a block-diagonal matrix system even when explicit time-stepping schemes are used to discretize the time derivative. Hence, the HDG method should be used with implicit time-stepping methods to deal with time-dependent problems.

The paper is organized as follows. In Section 2, we introduce the notation used throughout the paper. In Section 3, we introduce HDG methods for solving PDEs in fluid mechanics including the Stokes equations, incompressible Navier–Stokes equations, compressible Euler and Navier–Stokes equations. In particular, we briefly review the recent development of HDG methods for the incompressible Stokes and Navier–Stokes equations. Then we focus on devising HDG methods for the compressible Euler and Navier–Stokes equations, and systems of conservation laws in moving domains. Throughout this section we present numerical results to demonstrate the convergence and accuracy of the methods for a wide range of flow regimes. In Section 4, we focus on devising HDG methods for linear elasticity, nonlinear elasticity, and nonlinear elastodynamics. We then provide numerical experiments to show the competitiveness of the HDG methods. Finally, in Section 5, we end the paper with some concluding remarks.

2. Notation

Throughout this paper we shall denote scalar variables by italic letters with no boldface (a, A, b, B, etc.), vector variables by italic boldface lowercase letters ($\boldsymbol{a}, \boldsymbol{b}$, etc.), and second-order tensor variables by italic boldface uppercase letters ($\boldsymbol{A}, \boldsymbol{B}$, etc.). The identity tensor shall be denoted by $\boldsymbol{I}$. The components of $\boldsymbol{a}$ and $\boldsymbol{A}$ shall be denoted as a_i and A_{ij}, respectively. The symbols, $\cdot, \times, \otimes$, shall denote the usual scalar product, vector product, and tensor product, respectively. We shall use boldface roman uppercase letters ($\mathbf{A}, \mathbf{B}$, etc.) to denote matrices with entries (A_{ij}, B_{ij}, etc.) and boldface roman lowercase letters ($\mathbf{a}, \mathbf{b}$, etc.) to denote column vectors with elements (a_i, b_i, etc.). We shall also denote sets and spaces by calligraphic letters ($\mathcal{A}, \mathcal{B}$, etc.). In this paper, the tensor product notation and matrix product notation are interchanged, that is, $\boldsymbol{a} \cdot \boldsymbol{b} = \boldsymbol{a}^T\boldsymbol{b}$, $\boldsymbol{a} \otimes \boldsymbol{b} = \boldsymbol{a}\boldsymbol{b}^T$, $\boldsymbol{A} \cdot \boldsymbol{b} = \boldsymbol{A}\boldsymbol{b}$ and $\boldsymbol{A} \cdot \boldsymbol{B} = \boldsymbol{A}\boldsymbol{B}$.

2.1. Finite element mesh

Let Ω be a physical domain in $\mathbb{R}^d$ with Lipschitz boundary $\partial\Omega$ in $\mathbb{R}^{d-1}$. We denote by $\mathcal{T}_h$ a collection of disjoint elements (triangles and tetrahedrons) that partition Ω. We also denote by $\partial\mathcal{T}_h$ the set $\{\partial K : K \in \mathcal{T}_h\}$, that is, a collection of the boundaries of all elements in $\mathcal{T}_h$. We shall denote by $\boldsymbol{n}$ the outward unit normal of ∂K. For an element K of the collection $\mathcal{T}_h$, $F = \partial K \cap \partial\Omega$ is the boundary face if the $d-1$ Lebesgue measure of F is nonzero. For two elements K^+ and K^- of the collection $\mathcal{T}_h$, $F = \partial K^+ \cap \partial K^-$ is the interior face between K^+ and K^- if the $d-1$ Lebesgue measure of F is nonzero. Let $\mathcal{E}_h^o$ and $\mathcal{E}_h^\partial$ denote the set of interior and boundary faces, respectively. We denote by $\mathcal{E}_h$ the union of $\mathcal{E}_h^o$ and $\mathcal{E}_h^\partial$. Note that by definition $\partial\mathcal{T}_h$ and $\mathcal{E}_h$ are different. More precisely, an interior face is counted twice in $\partial\mathcal{T}_h$ but once in $\mathcal{E}_h$ and a boundary face is counted once in both $\partial\mathcal{T}_h$ and $\mathcal{E}_h$.

2.2. Approximation spaces

Let $\mathcal{P}^k(D)$ denote the set of polynomials of degree at most k on a domain D. We introduce discontinuous finite element spaces

$$\mathcal{W}_h^k = \{a \in L^2(\mathcal{T}_h) : a|_K \in \mathcal{P}^k(K),\ \forall K \in \mathcal{T}_h\},$$
$$\boldsymbol{\mathcal{V}}_h^k = \{\boldsymbol{a} \in (L^2(\mathcal{T}_h))^m : \boldsymbol{a}|_K \in (\mathcal{P}^k(K))^m,\ \forall K \in \mathcal{T}_h\},$$
$$\boldsymbol{\mathcal{Q}}_h^k = \{\boldsymbol{A} \in (L^2(\mathcal{T}_h))^{m\times d} : \boldsymbol{A}|_K \in (\mathcal{P}^k(K))^{m\times d},\ \forall K \in \mathcal{T}_h\},$$

for $\boldsymbol{a} = (a_i)$, $1 \leqslant i \leqslant m$, and $\boldsymbol{A} = (A_{ij}), 1 \leqslant i \leqslant m, 1 \leqslant j \leqslant d$. Here $L^2(D)$ is the space of square integrable functions on D. In addition, we introduce a traced finite element space

$$\boldsymbol{\mathcal{M}}_h^k = \{\boldsymbol{\mu} \in (L^2(\mathcal{E}_h))^m : \boldsymbol{\mu}|_F \in (\mathcal{P}^k(F))^m,\ \forall F \in \mathcal{E}_h\},$$

for $\boldsymbol{\mu} = (\mu_i)$, $1 \leqslant i \leqslant m$. Note that $\boldsymbol{\mathcal{M}}_h^k$ consists of functions which are continuous inside the faces $F \in \mathcal{E}_h$ and discontinuous at their borders.

For functions a and b in $L^2(D)$, we denote $(a,b)_D = \int_D ab$ if D is a domain in $\mathbb{R}^d$ and $\langle a,b\rangle_D = \int_D ab$ if D is a domain in $\mathbb{R}^{d-1}$. Likewise, for functions $\boldsymbol{a}$ and $\boldsymbol{b}$ in $(L^2(D))^m$, we denote $(\boldsymbol{a},\boldsymbol{b})_D = \int_D \boldsymbol{a}\cdot\boldsymbol{b}$ if D is a domain in $\mathbb{R}^d$ and $\langle \boldsymbol{a},\boldsymbol{b}\rangle_D = \int_D \boldsymbol{a}\cdot\boldsymbol{b}$ if D is a domain in $\mathbb{R}^{d-1}$. For functions $\boldsymbol{A}$ and $\boldsymbol{B}$ in $(L^2(D))^{m\times d}$, we denote $(\boldsymbol{A},\boldsymbol{B})_D = \int_D \mathrm{tr}(\boldsymbol{A}^T\boldsymbol{B})$ if D is a domain in $\mathbb{R}^d$ and $\langle \boldsymbol{A},\boldsymbol{B}\rangle_D = \int_D \mathrm{tr}(\boldsymbol{A}^T\boldsymbol{B})$ if D is a domain in $\mathbb{R}^{d-1}$, where tr is the trace operator of a square matrix. We finally introduce the following volume inner products

$$(a,b)_{\mathcal{T}_h} = \sum_{K\in\mathcal{T}_h}(a,b)_K, \quad (\boldsymbol{a},\boldsymbol{b})_{\mathcal{T}_h} = \sum_{K\in\mathcal{T}_h}(\boldsymbol{a},\boldsymbol{b})_K, \quad (\boldsymbol{A},\boldsymbol{B})_{\mathcal{T}_h} = \sum_{K\in\mathcal{T}_h}(\boldsymbol{A},\boldsymbol{B})_K,$$

and boundary inner products

$$\langle a,b\rangle_{\partial\mathcal{T}_h} = \sum_{K\in\mathcal{T}_h}\langle a,b\rangle_{\partial K}, \quad \langle \boldsymbol{a},\boldsymbol{b}\rangle_{\partial\mathcal{T}_h} = \sum_{K\in\mathcal{T}_h}\langle \boldsymbol{a},\boldsymbol{b}\rangle_{\partial K}, \quad \langle \boldsymbol{A},\boldsymbol{B}\rangle_{\partial\mathcal{T}_h} = \sum_{K\in\mathcal{T}_h}\langle \boldsymbol{A},\boldsymbol{B}\rangle_{\partial K}.$$

All of the above notations and definitions are necessary for the description of the ideas in this paper.

3. HDG methods for PDEs in fluid mechanics

In this section, we present HDG methods for solving steady and time-dependent PDEs in fluid mechanics. We begin by reviewing the recent development of HDG methods for Stokes problems [14,17,49] and incompressible Navier–Stokes problems [45,46,50]. We then focus on devising HDG methods for the compressible Euler and Navier–Stokes equations in both stationary and moving domains. We provide numerical results to illustrate the main characteristics of the HDG methods. We will aim to maintain a balance between generality and specificity, so that the essential ideas can easily be captured and extended to many other PDEs in fluid mechanics.

3.1. Stokes equations

We review here the HDG method introduced in [49] for solving the Stokes system

$$\begin{aligned} &-\nabla\cdot\nu\nabla\boldsymbol{u}+\nabla p=\boldsymbol{f}, \quad \text{in } \Omega,\\ &\nabla\cdot\boldsymbol{u}=0, \quad \text{in } \Omega,\\ &\boldsymbol{u}=\boldsymbol{g}, \quad \text{on } \partial\Omega. \end{aligned} \tag{1}$$

As usual we rewrite the above equation as the following first order system of equations

$$\begin{aligned} &\boldsymbol{L}-\nabla\boldsymbol{u}=0, \quad \text{in } \Omega\\ &-\nabla\cdot\nu\boldsymbol{L}+\nabla p=\boldsymbol{f}, \quad \text{in } \Omega,\\ &\nabla\cdot\boldsymbol{u}=0, \quad \text{in } \Omega,\\ &\boldsymbol{u}=\boldsymbol{g} \quad \text{on } \partial\Omega. \end{aligned} \tag{2}$$

We assume that $\boldsymbol{g}$ satisfies the compatibility condition $\int_{\partial\Omega}\boldsymbol{g}\cdot\boldsymbol{n}=0$.

The HDG method for the above Stokes problem seeks an approximation $(\boldsymbol{L}_h,\boldsymbol{u}_h,p_h,\hat{\boldsymbol{u}}_h)\in\mathcal{Q}_h^k\times\mathcal{V}_h^k\times\mathcal{W}_h^k\times\mathcal{M}_h^k$ such that

$$(\boldsymbol{L}_h,\boldsymbol{E})_{\mathcal{T}_h}+(\boldsymbol{u}_h,\nabla\cdot\boldsymbol{E})_{\mathcal{T}_h}-\langle\hat{\boldsymbol{u}}_h,\boldsymbol{E}\cdot\boldsymbol{n}\rangle_{\partial\mathcal{T}_h}=0, \tag{3a}$$

$$(\nu\boldsymbol{L}_h-p_h\boldsymbol{I},\nabla\boldsymbol{w})_{\mathcal{T}_h}+\langle\hat{\boldsymbol{h}}_h,\boldsymbol{w}\rangle_{\partial\mathcal{T}_h}=(\boldsymbol{f},\boldsymbol{w})_{\mathcal{T}_h}, \tag{3b}$$

$$-(\boldsymbol{u}_h,\nabla q)_{\mathcal{T}_h}+\langle\hat{\boldsymbol{u}}_h\cdot\boldsymbol{n},q\rangle_{\partial\mathcal{T}_h}=0, \tag{3c}$$

$$\langle\hat{\boldsymbol{h}}_h,\boldsymbol{\mu}\rangle_{\partial\mathcal{T}_h\setminus\partial\Omega}+\langle\hat{\boldsymbol{u}}_h-\boldsymbol{g},\boldsymbol{\mu}\rangle_{\partial\Omega}=0, \tag{3d}$$

$$(p_h,1)_{\mathcal{T}_h}=0, \tag{3e}$$

for all $(\boldsymbol{E},\boldsymbol{w},q,\boldsymbol{\mu})\in\mathcal{Q}_h^k\times\mathcal{V}_h^k\times\mathcal{W}_h^k\times\mathcal{M}_h^k$, where the so-called *interior numerical flux* $\hat{\boldsymbol{h}}_h$ is defined as

$$\hat{\boldsymbol{h}}_h=(-\nu\boldsymbol{L}_h+p_h\boldsymbol{I})\cdot\boldsymbol{n}+\boldsymbol{S}(\boldsymbol{u}_h-\hat{\boldsymbol{u}}_h). \tag{3f}$$

Here $\boldsymbol{S}$ is the second-order *stabilization tensor* which is typically chosen as

$$\boldsymbol{S}=\frac{\nu}{\ell}\boldsymbol{I}, \tag{4}$$

where ℓ is a characteristic length scale. In (3), the first three Eqs. (3a)–(3c) define the local solver; the forth Eq. (3d) enforces the continuity of the numerical across interior faces and imposes the Dirichlet boundary condition; the fifth Eq. (3e) is the average pressure condition needed for well-posedness of the HDG system (3). This completes the definition of the HDG method for solving the Stokes system (2).

The HDG method can be implemented by using an augmented Lagrangian approach [24]. In this approach, the Eq. (3c) is augmented with a pseudo time-derivative of the approximate pressure and the resulting system is advanced in time using

the backward-Euler method to the steady state. For every pseudo timestep, we eliminate the unknowns $(\boldsymbol{L}_h, \boldsymbol{u}_h, p_h)$ to obtain a matrix system involving only the degrees of freedom of $\hat{\boldsymbol{u}}_h$. Since the approximate trace $\hat{\boldsymbol{u}}_h$ is defined on element faces and singled-value on each face, the HDG method has less global degrees of freedom than many other DG methods. It is theoretically shown and numerically confirmed in [14,17,49] that the numerical approximations $(\boldsymbol{L}_h, \boldsymbol{u}_h, p_h)$ converge with the optimal order $k+1$. In addition, we can use an element-by-element postprocessing procedure proposed in [14] to obtain an exactly divergence-free, $\boldsymbol{H}(\mathrm{div})$-conforming approximate velocity which converges with the order $k+2$. Hence, the HDG method has higher order of convergence than other DG methods and CG methods.

3.2. Incompressible Navier–Stokes equations

3.2.1. Formulation

We next review the HDG method introduced in [45,46,50] for the steady incompressible Navier–Stokes equations

$$\begin{aligned} &\boldsymbol{L} - \nabla \boldsymbol{u} = 0, \quad \text{in } \Omega \\ &-\nabla \cdot \nu \boldsymbol{L} + \nabla p + \nabla \cdot (\boldsymbol{u} \otimes \boldsymbol{u}) = \boldsymbol{f}, \quad \text{in } \Omega, \\ &\nabla \cdot \boldsymbol{u} = 0, \quad \text{in } \Omega, \\ &\boldsymbol{u} = \boldsymbol{g}, \quad \text{on } \partial\Omega, \end{aligned} \tag{5}$$

with the compatibility condition $\int_{\partial\Omega} \boldsymbol{g} \cdot \boldsymbol{n} = 0$. We shall discuss the treatment of Neumann boundary conditions shortly later.

The HDG method for the above system seeks an approximation $(\boldsymbol{L}_h, \boldsymbol{u}_h, p_h, \hat{\boldsymbol{u}}_h) \in \mathcal{Q}_h^k \times \mathcal{V}_h^k \times \mathcal{W}_h^k \times \mathcal{M}_h^k$ such that

$$(\boldsymbol{L}_h, \boldsymbol{E})_{\mathcal{T}_h} + (\boldsymbol{u}_h, \nabla \cdot \boldsymbol{E})_{\mathcal{T}_h} - \langle \hat{\boldsymbol{u}}_h, \boldsymbol{E} \cdot \boldsymbol{n} \rangle_{\partial\mathcal{T}_h} = 0, \tag{6a}$$

$$(\nu \boldsymbol{L}_h - p_h \boldsymbol{I} - \boldsymbol{u}_h \otimes \boldsymbol{u}_h, \nabla \boldsymbol{w})_{\mathcal{T}_h} + \langle \hat{\boldsymbol{h}}_h, \boldsymbol{w} \rangle_{\partial\mathcal{T}_h} = (\boldsymbol{f}, \boldsymbol{w})_{\mathcal{T}_h}, \tag{6b}$$

$$-(\boldsymbol{u}_h, \nabla q)_{\mathcal{T}_h} + \langle \hat{\boldsymbol{u}}_h \cdot \boldsymbol{n}, q \rangle_{\partial\mathcal{T}_h} = 0, \tag{6c}$$

$$\langle \hat{\boldsymbol{h}}_h, \boldsymbol{\mu} \rangle_{\partial\mathcal{T}_h \setminus \partial\Omega} + \langle \hat{\boldsymbol{u}}_h - \boldsymbol{g}, \boldsymbol{\mu} \rangle_{\partial\Omega} = 0, \tag{6d}$$

$$(p_h, 1)_{\mathcal{T}_h} = 0, \tag{6e}$$

for all $(\boldsymbol{E}, \boldsymbol{w}, q, \boldsymbol{\mu}) \in \mathcal{Q}_h^k \times \mathcal{V}_h^k \times \mathcal{W}_h^k \times \mathcal{M}_h^k$, where the interior numerical flux $\hat{\boldsymbol{h}}_h$ is defined as

$$\hat{\boldsymbol{h}}_h = (-\nu \boldsymbol{L}_h + p_h \boldsymbol{I} + \hat{\boldsymbol{u}}_h \otimes \hat{\boldsymbol{u}}_h) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h). \tag{6f}$$

The stabilization matrix is typically chosen as

$$\boldsymbol{S} = \left(\frac{\nu}{\ell} + \tau_c\right)\boldsymbol{I}, \tag{7}$$

where τ_c is the stabilization parameter due to the nonlinear convective term. It can be selected as $|\boldsymbol{u}_h \cdot \boldsymbol{n}|$ or simply as an upper bound of $|\boldsymbol{u} \cdot \boldsymbol{n}|$ over the entire domain.

The HDG method can treat other boundary conditions effectively. In particular, let us consider the following boundary conditions

$$\begin{aligned} &\boldsymbol{u} = \boldsymbol{g}_D, \quad \text{on } \partial\Omega_D, \\ &\boldsymbol{B}\boldsymbol{n} = \boldsymbol{g}_N, \quad \text{on } \partial\Omega_N, \end{aligned} \tag{8}$$

where $\partial\Omega_N$ is a part of the boundary $\partial\Omega$ such that $\partial\Omega_N \cup \partial\Omega_D = \partial\Omega$ and $\partial\Omega_N \cap \partial\Omega_D = \emptyset$. Here $\boldsymbol{B}$ is a linear operator that depends on $(\boldsymbol{L}, \boldsymbol{u}, p)$. Examples of the form of $\boldsymbol{B}$ are given in Table 1. Note that the third and fourth rows in the Table 1 are boundary conditions on the vorticity since $(\boldsymbol{L} - \boldsymbol{L}^T)\boldsymbol{n} = \omega \times \boldsymbol{n}$, where ω is the vorticity vector. In order to incorporate the above boundary conditions, we only need to replace the fourth Eq. (6d) in (6) with the following equation

Table 1
Examples of other boundary conditions for the incompressible Navier–Stokes equations. Note that the asterisk symbol * indicates that the average pressure condition $(p_h, 1)_\Omega = 0$ is also imposed. The dagger symbol † indicates that a Dirichlet boundary condition for the normal component of the velocity has also to be provided on $\partial\Omega_N$.

Condition type	$\boldsymbol{B}$	$\hat{\boldsymbol{b}}_h$
Stress	$-\nu(\boldsymbol{L} + \boldsymbol{L}^T) + p\boldsymbol{I}$	$(-\nu(\boldsymbol{L}_h + \boldsymbol{L}_h^T) + p_h\boldsymbol{I}) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h)$
Viscous stress*	$-\nu(\boldsymbol{L} + \boldsymbol{L}^T)$	$-\nu(\boldsymbol{L}_h + \boldsymbol{L}_h^T) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h)$
Vorticity + pressure	$-\nu(\boldsymbol{L} - \boldsymbol{L}^T) + p\boldsymbol{I}$	$(-\nu(\boldsymbol{L}_h - \boldsymbol{L}_h^T) + p_h\boldsymbol{I}) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h)$
Vorticity*,†	$-\nu(\boldsymbol{L} - \boldsymbol{L}^T)$	$-\nu(\boldsymbol{L}_h - \boldsymbol{L}_h^T) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h)$
Gradient + pressure	$-\nu\boldsymbol{L} + p\boldsymbol{I}$	$(-\nu\boldsymbol{L}_h + p_h\boldsymbol{I}) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h)$
Gradient*	$-\nu\boldsymbol{L}$	$-\nu\boldsymbol{L}_h \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h)$

$$\langle \hat{\boldsymbol{h}}_h, \boldsymbol{\mu} \rangle_{\partial \mathcal{T}_h \setminus \partial\Omega} + \langle \hat{\boldsymbol{u}}_h - \boldsymbol{g}_D, \boldsymbol{\mu} \rangle_{\partial\Omega_D} + \langle \hat{\boldsymbol{b}}_h - \boldsymbol{g}_N, \boldsymbol{\mu} \rangle_{\partial\Omega_N} = 0. \tag{9}$$

Here $\hat{\boldsymbol{b}}_h$ is the *boundary numerical flux* on $\partial\Omega_N$ and is tabulated in Table 1. Multiple Dirichlet and Neumann boundary conditions can be treated in a similar manner.

The implementation of the HDG method for the incompressible Navier–Stokes equations is more involved than that for the Stokes equations due to the nonlinear convective term. We briefly summarize the key implementation steps as the detailed discussion can be found in [45]. We apply the Newton–Raphson method to linearize (6) to obtain a linearized system. There are two different strategies proposed in [45] for solving the linearized system. In the first strategy, we hybridize the linearized system to obtain a reduced linear system involving the degrees of freedom of the approximate velocity and average pressure. The reduced linear system has a structure of the saddle point problem. In the second strategy, we use the augmented Lagrangian method developed for the Stokes equations [17,49] to solve the linearized system. Within each iteration of the augmented Lagrangian method, we solve a linear system involving the degrees of freedom of the approximate velocity only.

In addition, the local postprocessing procedure proposed in [14,45] can be used to obtain an exactly divergence-free, $\boldsymbol{H}$(div)-conforming approximate velocity $\boldsymbol{u}_h^*$. Numerical results presented in [45] show that the approximate pressure, velocity and velocity gradient converge with the optimal order $k+1$ for diffusion-dominated problems with smooth solutions. In such case, the postprocessed velocity $\boldsymbol{u}_h^*$ converges with the order $k+2$ for $k \geqslant 1$. Moreover, the postprocessed velocity is less expensive to compute than the original velocity $\boldsymbol{u}_h$ since the postprocessing is performed at the element level.

3.2.2. The Kovasznay flow example

We now compare the HDG method with the standard Taylor–Hood finite element (FE) method for a test case. In particular, we consider the Kovasznay flow [34] with the analytical solution

$$u_1 = 1 - \exp(\lambda x_1)\cos(2\pi x_2), \quad u_2 = \frac{\lambda}{2\pi}\exp(\lambda x_1)\sin(2\pi x_2), \quad p = -\frac{1}{2}\exp(2\lambda x_1) + c,$$

where $\lambda = \frac{Re}{2} - \sqrt{\frac{Re^2}{4} + 4\pi^2}$ and $Re = \frac{1}{\nu}$ is the Reynolds number. We take Dirichlet boundary conditions for the velocity as the restriction of the exact solution to the domain boundary and the Reynolds number $Re = 20$ so that $\nu = 0.05$. The computational domain is $\Omega = (0,2) \times (-0.5, 1.5)$.

We consider triangular meshes that are obtained by splitting a regular $n \times n$ Cartesian grid into a total of $2n^2$ triangles, giving uniform element sizes of $h = 2/n$. On these meshes, we use polynomials of degree k to represent all the approximate variables using a nodal basis within each element for the HDG method. However, we use the P2/P1 Taylor–Hood element ($k = 2$ for the approximate velocity and $k = 1$ for the approximate pressure) and P3/P2 Taylor–Hood element ($k = 3$ for the approximate velocity and $k = 2$ for the approximate pressure) for the CG method.

We present in Table 2 the error and order of convergence in L^2-norm for both the HDG method and the Taylor–Hood FE method. Here we take the stabilization matrix to be the identity matrix. We see that the HDG method provides more accurate solutions than the Taylor–Hood FE method for the same mesh size and polynomial degree (of the approximate velocity). In particular, the approximate pressure, velocity gradient and postprocessed velocity of the HDG method converge with one order higher than those of the Taylor–Hood FE method. Furthermore, we emphasize that the postprocessed velocity of the HDG method is exactly divergence-free and $\boldsymbol{H}$(div)-conforming.

Table 2
Comparison of the convergence of the L^2 errors in the pressure, velocity gradient, and velocity between the HDG method and the continuous Taylor–Hood FE method for the Kovasznay example. Note that the postprocessed velocity $\boldsymbol{u}_h^*$ is taken as the approximation for the HDG method and that the polynomial degree of the approximate velocity is taken for the Taylor–Hood FE method.

Degree	Mesh	HDG						Taylor–Hood FEM					
		$\|p - p_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{L} - \boldsymbol{L}_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{u} - \boldsymbol{u}_h^*\|_{\mathcal{T}_h}$		$\|\boldsymbol{u} - \boldsymbol{u}_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{L} - \boldsymbol{L}_h\|_{\mathcal{T}_h}$		$\|p - p_h\|_{\mathcal{T}_h}$	
k	n	Error	Order	Error	Order	Error	Order	Error	Order	Error	Order	Error	Order
2	4	1.48e−1	–	2.16e−0	–	1.38e−1	–	1.31e−1	–	1.58e−0	–	8.71e−2	–
	8	9.02e−3	4.03	2.37e−1	3.19	8.28e−3	4.05	1.34e−2	3.29	3.73e−1	2.08	1.47e−2	2.56
	16	9.32e−4	3.27	3.37e−2	2.81	5.47e−4	3.92	1.52e−3	3.15	9.26e−2	2.01	2.83e−3	2.38
	32	1.12e−4	3.06	4.62e−3	2.87	3.75e−5	3.87	1.82e−4	3.06	2.31e−2	2.01	6.56e−4	2.11
	64	1.38e−5	3.02	6.08e−4	2.93	2.46e−6	3.93	2.25e−5	3.02	5.76e−3	2.00	1.62e−4	2.02
3	4	1.57e−2	–	3.00e−1	–	1.42e−2	–	1.62e−2	–	3.66e−1	–	2.13e−2	–
	8	7.93e−4	4.31	2.47e−2	3.60	5.68e−4	4.64	1.20e−3	3.76	5.02e−2	2.87	1.30e−3	4.03
	16	5.01e−5	3.98	1.73e−3	3.83	1.89e−5	4.91	7.78e−5	3.94	6.37e−3	2.98	1.30e−4	3.32
	32	3.18e−6	3.98	1.16e−4	3.90	6.37e−7	4.89	5.00e−6	3.96	8.06e−4	2.98	1.54e−5	3.08
	64	2.00e−7	3.99	7.52e−6	3.95	2.07e−8	4.94	3.15e−7	3.99	1.00e−4	3.01	1.89e−6	3.02

3.3. Compressible Euler equations

We consider the steady-state Euler equations of gas dynamics written in nondimensional conservation form as

$$\nabla \cdot \boldsymbol{F}(\boldsymbol{u}) = \boldsymbol{f}, \quad \text{in } \Omega, \tag{10}$$

where $\boldsymbol{u} = (u_i)$, $1 \leqslant i \leqslant m = d + 2$, is a vector of conserved dimensionless quantities (namely, density, momentum and energy), $\boldsymbol{F}(\boldsymbol{u})$ are inviscid fluxes of dimension $m \times d$, and $\boldsymbol{f}$ is a source term. The Euler Eq. (10) must be supplemented with appropriate boundary conditions at the inflow and outflow boundaries and at the solid wall. We shall discuss these boundary conditions later.

3.3.1. Formulation

We now devise a HDG method for solving the compressible Euler system (10). The steps toward this goal are proceeded as follows. First, we consider the governing equations on any element $K \in \mathcal{T}_h$, multiply them with a test function $\boldsymbol{w} \in (\mathcal{P}^k(K))^m$, and integrate the resulting equations by part. This results in a so-called *local problem*: $\boldsymbol{u}_h \in \boldsymbol{\mathcal{V}}_h^k$ satisfies

$$-(\boldsymbol{F}(\boldsymbol{u}_h), \nabla \boldsymbol{w})_K + \langle \hat{\boldsymbol{h}}_h, \boldsymbol{w} \rangle_{\partial K} = (\boldsymbol{f}, \boldsymbol{w})_K, \quad \forall\, \boldsymbol{w} \in (\mathcal{P}^k(K))^m, \tag{11}$$

where the numerical flux $\hat{\boldsymbol{h}}_h$ is an approximation to $\boldsymbol{F}(\boldsymbol{u}) \cdot \boldsymbol{n}$ over ∂K. Note that (11) is a Galerkin projection of the governing equations at the element level onto $(\mathcal{P}^k(K))^m$. Second, we take the numerical flux to be

$$\hat{\boldsymbol{h}}_h = \boldsymbol{F}(\hat{\boldsymbol{u}}_h) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h), \quad \text{on } \partial K, \tag{12}$$

where $\hat{\boldsymbol{u}}_h \in \boldsymbol{\mathcal{M}}_h^k$ is an approximation to the trace of the solution $\boldsymbol{u}$ on ∂K, and $\boldsymbol{S}$ is a stabilization matrix which has an important effect on both the stability and accuracy of the resulting scheme. Third, we require that

$$\langle \hat{\boldsymbol{h}}_h, \boldsymbol{\mu} \rangle_{\partial \mathcal{T}_h \setminus \partial \Omega} + \langle \hat{\boldsymbol{b}}_h, \boldsymbol{\mu} \rangle_{\partial \Omega} = 0, \quad \forall \boldsymbol{\mu} \in \boldsymbol{\mathcal{M}}_h^k, \tag{13}$$

where $\hat{\boldsymbol{b}}_h$ is the boundary numerical flux whose definition depends on the boundary conditions and will be given below. The first term in (13) enforces the continuity of the L^2 projection of $\hat{\boldsymbol{h}}_h$ across interior faces, while the second term imposes the boundary conditions.

Finally, by inserting (12) into (11) and (13) and summing the resulting equations over all elements we obtain the HDG formulation: $(\boldsymbol{u}_h, \hat{\boldsymbol{u}}_h) \in \boldsymbol{\mathcal{V}}_h^k \times \boldsymbol{\mathcal{M}}_h^k$ satisfies

$$-(\boldsymbol{F}(\boldsymbol{u}_h), \nabla \boldsymbol{w})_{\mathcal{T}_h} + \langle \boldsymbol{F}(\hat{\boldsymbol{u}}_h) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h), \boldsymbol{w} \rangle_{\partial \mathcal{T}_h} = (\boldsymbol{f}, \boldsymbol{w})_{\mathcal{T}_h}, \quad \forall \boldsymbol{w} \in \boldsymbol{\mathcal{V}}_h^k, \tag{14a}$$

$$\langle \boldsymbol{F}(\hat{\boldsymbol{u}}_h) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h), \boldsymbol{\mu} \rangle_{\partial \mathcal{T}_h \setminus \partial \Omega} + \langle \hat{\boldsymbol{b}}_h, \boldsymbol{\mu} \rangle_{\partial \Omega} = 0, \quad \forall \boldsymbol{\mu} \in \boldsymbol{\mathcal{M}}_h^k. \tag{14b}$$

This system is the HDG method for the Euler Eq. (10). It remains to define the boundary numerical flux and the stabilization matrix.

3.3.2. Boundary conditions

At the inlet section or outlet section of the flow, we need to either set the state variable $\boldsymbol{u}$ to the freestream condition $\boldsymbol{u}_\infty$ or extrapolate it depending on the eigenvalues of the system. To this end, we define the boundary flux vector $\hat{\boldsymbol{b}}_h$ as

$$\hat{\boldsymbol{b}}_h = \boldsymbol{A}_n^+(\hat{\boldsymbol{u}}_h)(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h) - \boldsymbol{A}_n^-(\hat{\boldsymbol{u}}_h)(\boldsymbol{u}_\infty - \hat{\boldsymbol{u}}_h), \tag{15}$$

where $\boldsymbol{A}_n^\pm = (\boldsymbol{A}_n \pm |\boldsymbol{A}_n|)/2$ and $\boldsymbol{A}_n = [\partial \boldsymbol{F}(\boldsymbol{u})/\partial \boldsymbol{u}] \cdot \boldsymbol{n}$ denotes the Jacobian of the normal component of the inviscid fluxes.

At the solid surface with slip condition, we must impose zero normal velocity and extrapolate the density, the tangential velocity, and the energy. To this end we set

$$\hat{\boldsymbol{b}}_h = \boldsymbol{g}_h - \hat{\boldsymbol{u}}_h, \tag{16}$$

where the vector $\boldsymbol{g}_h$ is defined in terms of $\boldsymbol{u}_h$ as follows

$$g_{h1} = u_{h1}, \quad (g_{h2}, \ldots, g_{hm-1}) = \boldsymbol{v}_h - (\boldsymbol{v}_h \cdot \boldsymbol{n})\boldsymbol{n}, \quad g_{hm} = u_{hm}. \tag{17}$$

Here $\boldsymbol{v}_h = (u_{h2}, \ldots, u_{hm-1})$ are the velocity components of $\boldsymbol{u}_h$. Note that since $(\boldsymbol{v}_h - (\boldsymbol{v}_h \cdot \boldsymbol{n})\boldsymbol{n}) \cdot \boldsymbol{n} = 0$ we have $\hat{\boldsymbol{v}}_h \cdot \boldsymbol{n} = 0$ on the solid wall, where $\hat{\boldsymbol{v}}_h = (\hat{u}_{h2}, \ldots, \hat{u}_{hm-1})$ are the velocity components of $\hat{\boldsymbol{u}}_h$.

3.3.3. Stabilization matrix

There are several possible choices for the stabilization matrix including the Roe scheme [60] and Lax–Friedrich scheme [19]. For the Roe scheme, we choose

$$\boldsymbol{S} = \boldsymbol{L}|\boldsymbol{\Lambda}|\boldsymbol{R}, \tag{18}$$

where $\boldsymbol{L}$, $\boldsymbol{R}$, and $\boldsymbol{\Lambda}$ are the matrices of the left and right eigenvectors, and eigenvalues of the Jacobian matrix $[\partial \boldsymbol{F}(\hat{\boldsymbol{u}}_h)/\partial \hat{\boldsymbol{u}}_h] \cdot \boldsymbol{n}$, respectively. The second choice is the local Lax–Friedrich scheme

$$\boldsymbol{S} = \tau^a_{\max}\boldsymbol{I}, \tag{19}$$

where $\tau^a_{\max}$ is the local maximum speed of the system, and $\boldsymbol{I}$ is the identity matrix. The third choice is the global Lax–Friedrich scheme

$$\boldsymbol{S} = \tau^g_{\max}\boldsymbol{I}, \tag{20}$$

where $\tau^g_{\max}$ is the global maximum speed of the system. In general, the choice of the stabilization matrix becomes less critical for high k since numerical dissipation in the order of $O(h^{k+1})$ vanishes rapidly with increasing k.

3.3.4. Implementation

By applying the Newton–Raphson method to linearize the nonlinear system (14), we obtain the following linear system at every ℓth Newton iteration

$$\begin{bmatrix} \mathbf{A}^\ell & \mathbf{B}^\ell \\ \mathbf{C}^\ell & \mathbf{D}^\ell \end{bmatrix} \begin{pmatrix} \delta\mathbf{u}^\ell \\ \delta\hat{\mathbf{u}}^\ell \end{pmatrix} = \begin{pmatrix} \mathbf{f}^\ell \\ \mathbf{g}^\ell \end{pmatrix}, \tag{21}$$

where $\delta\mathbf{u}^\ell$ and $\delta\hat{\mathbf{u}}^\ell$ are the vectors of degrees of freedom of $\delta\boldsymbol{u}_h^\ell$ and $\delta\hat{\boldsymbol{u}}_h^\ell$, respectively, which are in that order the Newton increments of the ℓth iterates $\boldsymbol{u}_h^\ell$ and $\hat{\boldsymbol{u}}_h^\ell$. It is important to note that the matrix $\mathbf{A}^\ell$ has a block-diagonal structure due to the discontinuous nature of the approximation spaces. Therefore, it can be inverted at the element level to yield a block-diagonal matrix $(\mathbf{A}^\ell)^{-1}$. We can thus eliminate $\delta\mathbf{u}^\ell$ to obtain a reduced system in terms of $\delta\hat{\mathbf{u}}^\ell$ as

$$\mathbf{K}^\ell\delta\hat{\mathbf{u}}^\ell = \mathbf{r}^\ell, \tag{22}$$

where

$$\mathbf{K}^\ell = -\mathbf{C}^\ell(\mathbf{A}^\ell)^{-1}\mathbf{B}^\ell + \mathbf{D}^\ell, \quad \mathbf{r}^\ell = \mathbf{g}^\ell - \mathbf{C}^\ell(\mathbf{A}^\ell)^{-1}\mathbf{f}^\ell. \tag{23}$$

This is the global system to be solved at every Newton iteration. Since $\delta\hat{\boldsymbol{u}}_h^\ell$ is single-valued over faces of the elements, the final matrix system of the HDG method is smaller than that of many other DG methods. Moreover, the matrix $\mathbf{K}^\ell$ is compact in the sense that only the degrees of freedom between neighboring faces that share the same element are connected. To form $\mathbf{K}^\ell$ we does not need to explicitly compute the matrices $\mathbf{A}^\ell$, $\mathbf{B}^\ell$, $\mathbf{C}^\ell$, and $\mathbf{D}^\ell$. Instead we compute the elemental matrices and elemental vectors, and perform the standard finite element assembly to form the system (22).

For large problems iterative solution methods are unavoidable. One of the key indicators of the cost in an iterative method is the cost of the matrix vector multiplication which is proportional to the number of non-zeros in the problem matrix. For HDG one not only obtains a smaller matrix with fewer globally coupled degrees of freedom, but for a given size matrix the number of nonzeros is smaller. This is because in standard DG methods the number of nonzero elements scales like $O(k^d)$, whereas in the HDG method the number of nonzeros scales like $O(k^{d-1})$. This has the potential for significantly smaller matrices and hence more efficient solution techniques.

3.3.5. Numerical examples

We first consider the Ringleb flow to demonstrate the optimal accuracy of the HDG method. The Ringleb flow is an exact smooth solution of the Euler equations obtained using the hodograph method [9]. For any given (x,y), we first obtain the radial velocity V by solving the following nonlinear equation

$$(x - 0.5L^2) + y^2 = \frac{1}{4\rho^2V^4},$$

where

$$c = \sqrt{1 - \frac{V^2}{5}} \quad \rho = c^5 \quad L = \frac{1}{c} + \frac{1}{3c^3} + \frac{1}{5c^5} - \frac{1}{2}\ln\frac{1+c}{1-c}.$$

We then compute the exact solution as

$$\rho = c^5 \quad p = c^7/\gamma \quad v_1 = V\cos(\theta) \quad v_2 = V\sin(\theta),$$

where

$$\psi = \sqrt{\frac{1}{2V^2} - (x - 0.5L)\rho} \quad \theta = \arcsin(\psi V).$$

Since the exact solution can be determined for any spatial point, we take the domain Ω to be $(-2,-1)\times(1,2)$. The boundary condition is prescribed by setting the freestream value $\boldsymbol{u}_\infty$ to the exact solution on the boundary of the domain. We consider triangular meshes that are obtained by splitting a regular $n \times n$ Cartesian grid into $2n^2$ triangles. On these meshes, we use polynomials of degree k to represent all the approximate variables with a nodal basis [29]. We present in Table 3 the L^2 error

Table 3
History of convergence of the HDG method for the Ringleb flow.

Mesh $1/h$	$k=1$		$k=2$		$k=3$		$k=4$	
	Error	Order	Error	Order	Error	Order	Error	Order
2	4.35e−3	–	3.24e−4	–	2.35e−5	–	2.08e−6	–
4	1.10e−3	1.98	4.85e−5	2.74	1.43e−6	4.04	7.90e−8	4.72
8	2.80e−4	1.98	6.92e−6	2.81	8.63e−8	4.05	2.80e−9	4.82
16	7.06e−5	1.99	9.37e−7	2.88	5.18e−9	4.06	9.36e−11	4.90

and convergence rate of the numerical solution $\boldsymbol{u}_h$ as a function of h and k. We observe that the approximate solution converges with the optimal order $k+1$. Hence, the convergence of the HDG method is optimal.

The second example we consider is an inviscid flow over an airfoil. The geometry of the airfoil is obtained by transforming a circle in the complex ζ-plane into the complex z-plane using a Kármán–Trefftz transformation

$$\frac{z-n}{z+n}=\left(\frac{\zeta-1}{\zeta+1}\right)^n. \tag{24}$$

For the particular airfoil considered here the circle on the ζ-plane is centered at point $(-0.05, 0.05)$ and the radius of the circle is such that it passes through the point $(1,0)$. The value of n determines the trailing edge angle β_t, and for our example it is chosen as $n = 2 - \beta_t/\pi = 1.98$. The results reported below have been obtained with the Roe stabilization matrix (18), but we have found no appreciable differences when using the Lax stabilization form (20). The airfoil geometry as well as a detail of the mesh utilized in all our computations is shown in Fig. 1. The figure also shows the Mach number contours obtained for a free stream Mach number $M_\infty = 0.1$ and an angle of attach $\alpha = 0$ using $k = 4$. In order to validate this solution, we show in Fig. 2 a comparison of the pressure coefficient computed with $k = 4$ and the analytical potential solution for the incompressible case. The very minor differences between the two solutions can be attributed to compressibility effects as well as to the proximity of the far field boundary in the Euler computations. In the computations, the far field boundary has only been placed at a distance of five chords away from the airfoil and no vortex correction has been applied. We expect the errors to be small since the angle of attack is zero and hence the lift is small which in turn implies that the first-order vortex correction would also be small.

This last example serves to demonstrate that high-order polynomials produce much more accurate solutions at lower cost than low-order polynomials. In particular, we consider an inviscid subsonic flow past a circular bump in a channel at $M_\infty = 0.5$. The length and height of the channel are 3 and 1.5, respectively; and the length of the circular bump is 0.2. Inlet/outlet conditions are prescribed at the left/right boundaries, while inviscid wall boundary condition is used on the top and bottom sides. We use two different regular meshes to compute the numerical solutions: one mesh has 6400 linear ($k = 1$) elements and the other mesh has 400 fourth-order ($k = 4$) elements. Hence, the two meshes have the same resolution h/k. Fig. 3 shows the Mach number contours obtained with these two meshes. It is clear that the fifth-order solution is superior to the second-order solution, although the former has only 12,600 global degrees of freedom while the latter has 77,760 global degrees of freedom.

3.4. Compressible Navier–Stokes equations

We extend the HDG method developed above to the steady-state compressible Navier–Stokes equations in conservation form as

$$\nabla\cdot(\boldsymbol{F}(\boldsymbol{u})+\boldsymbol{G}(\boldsymbol{u},\nabla\boldsymbol{u}))=\boldsymbol{f},\quad \text{in } \Omega, \tag{25}$$

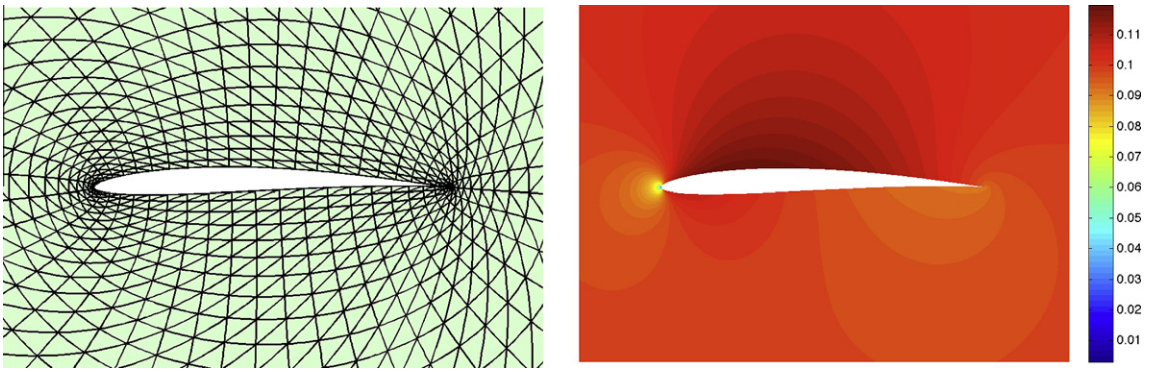

Fig. 1. Inviscid flow over a Kármán–Trefftz airfoil at $M_\infty = 0.1$ and $\alpha = 0$: detail of the mesh (left) and Mach number contours for $k = 4$ (right).

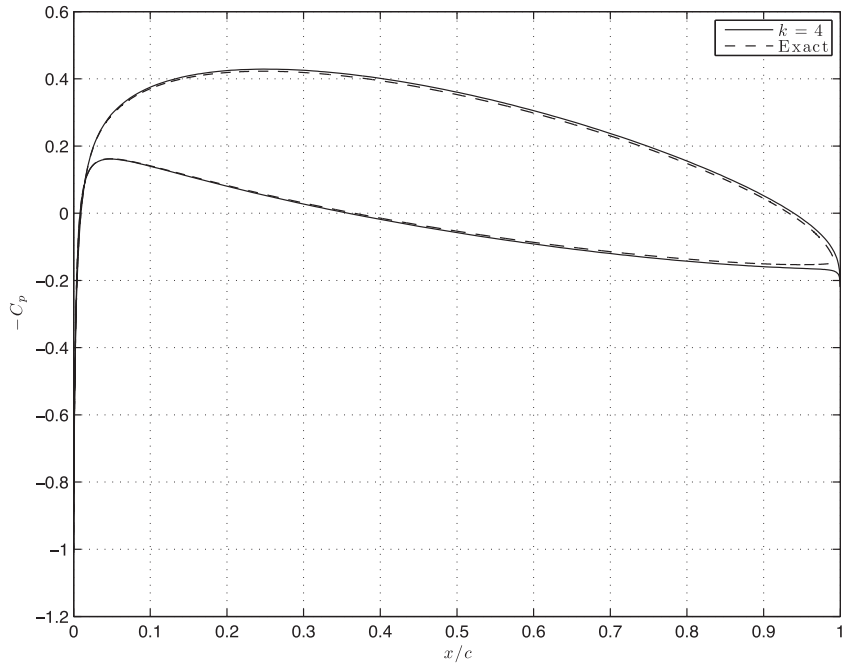

Fig. 2. Inviscid flow over a Kármán–Trefftz airfoil at $M_\infty = 0.1$ and $\alpha = 0$: comparison of the pressure coefficient between the HDG solution for $M_\infty = 0.1$ using $k = 4$ and the exact incompressible potential solution.

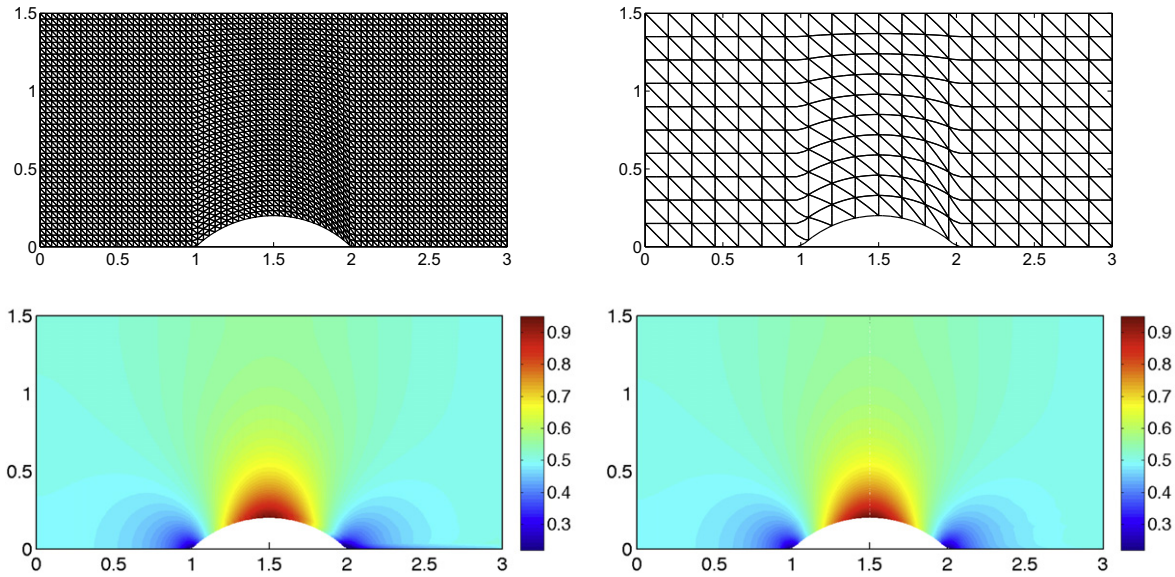

Fig. 3. Inviscid flow past a circular bump in a channel at $M_\infty = 0.5$: comparison of the Mach number contour between a fine regular mesh of 6400 linear elements (left) and a coarse regular mesh of 400 fourth-order elements (right).

where the viscous fluxes $\boldsymbol{G}(\boldsymbol{u}, \nabla \boldsymbol{u})$ of dimension $m \times d$ depends on the state vector $\boldsymbol{u}$ and its gradient $\nabla \boldsymbol{u}$. The nondimensional form of the Navier–Stokes equations as well as the definition of the inviscid and viscous fluxes can be found in [2]. Of course, the Navier–Stokes equations (25) should be supplemented with appropriate boundary conditions at the inflow, outflow and solid wall boundaries. The treatment of these boundary conditions will be discussed later.

3.4.1. Formulation

To begin we consider the Navier–Stokes system (25) on any element $K \in \mathcal{T}_h$ and rewrite it as

$$\begin{aligned} &\boldsymbol{Q} - \nabla \boldsymbol{u} = 0, \quad \text{in } K, \\ &\nabla \cdot (\boldsymbol{F}(\boldsymbol{u}) + \boldsymbol{G}(\boldsymbol{u}, \boldsymbol{Q})) = \boldsymbol{f}, \quad \text{in } K. \end{aligned} \tag{26}$$

Multiplying this system with test functions $(\boldsymbol{E}, \boldsymbol{w}) \in (\mathcal{P}^k(K))^{m\times d} \times (\mathcal{P}^k(K))^m$ and carrying out integration by parts we obtain $(\boldsymbol{Q}_h, \boldsymbol{u}_h) \in \mathcal{Q}_h^k \times \mathcal{V}_h^k$ such that

$$\begin{aligned} &(\boldsymbol{Q}_h, \boldsymbol{E})_K + (\boldsymbol{u}_h, \nabla \cdot \boldsymbol{E})_K - \langle \hat{\boldsymbol{u}}_h, \boldsymbol{E} \cdot \boldsymbol{n} \rangle_{\partial K} = 0, \quad \forall \boldsymbol{E} \in (\mathcal{P}^k(K))^{m\times d}, \\ &-(\boldsymbol{F}(\boldsymbol{u}_h) + \boldsymbol{G}(\boldsymbol{u}_h, \boldsymbol{Q}_h), \nabla \boldsymbol{w})_K + \langle \hat{\boldsymbol{h}}_h, \boldsymbol{w} \rangle_{\partial K} = (\boldsymbol{f}, \boldsymbol{w})_K, \quad \forall \boldsymbol{w} \in (\mathcal{P}^k(K))^m. \end{aligned} \tag{27}$$

This is a Galerkin projection of the governing equations at the element level onto $(\mathcal{P}^k(K))^{m\times d} \times (\mathcal{P}^k(K))^m$. The interior numerical flux is defined as

$$\hat{\boldsymbol{h}}_h = (\boldsymbol{F}(\hat{\boldsymbol{u}}_h) + \boldsymbol{G}(\hat{\boldsymbol{u}}_h, \boldsymbol{Q}_h)) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h), \quad \text{on } \partial K. \tag{28}$$

Then we require that

$$\langle \hat{\boldsymbol{h}}_h, \boldsymbol{\mu} \rangle_{\partial \mathcal{T}_h \setminus \partial\Omega} + \langle \hat{\boldsymbol{b}}_h, \boldsymbol{\mu} \rangle_{\partial\Omega} = 0, \quad \forall \boldsymbol{\mu} \in \boldsymbol{\mathcal{M}}_h^k, \tag{29}$$

where the boundary numerical flux $\hat{\boldsymbol{b}}_h$ will be defined later. The Eq. (27) defines a local problem in which $(\boldsymbol{Q}_h, \boldsymbol{u}_h)$ can be solved in an element-by-element fashion] whenever $\hat{\boldsymbol{u}}_h$ is known. The Eq. (29) is thus a global weak formulation in terms of $\hat{\boldsymbol{u}}_h$ since, through the local problem, both $\hat{\boldsymbol{h}}_h$ and $\hat{\boldsymbol{b}}_h$ can be interpreted as functions of $\hat{\boldsymbol{u}}_h$.

Finally, by summing (27) over all elements we obtain the HDG formulation: $(\boldsymbol{Q}_h, \boldsymbol{u}_h, \hat{\boldsymbol{u}}_h) \in \mathcal{Q}_h^k \times \mathcal{V}_h^k \times \boldsymbol{\mathcal{M}}_h^k$ satisfies

$$(\boldsymbol{Q}_h, \boldsymbol{E})_{\mathcal{T}_h} + (\boldsymbol{u}_h, \nabla \cdot \boldsymbol{E})_{\mathcal{T}_h} - \langle \hat{\boldsymbol{u}}_h, \boldsymbol{E} \cdot \boldsymbol{n} \rangle_{\partial \mathcal{T}_h} = 0, \tag{30a}$$

$$-(\boldsymbol{F}(\boldsymbol{u}_h) + \boldsymbol{G}(\boldsymbol{u}_h, \boldsymbol{Q}_h), \nabla \boldsymbol{w})_{\mathcal{T}_h} + \langle (\boldsymbol{F}(\hat{\boldsymbol{u}}_h) + \boldsymbol{G}(\hat{\boldsymbol{u}}_h, \boldsymbol{Q}_h)) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h), \boldsymbol{w} \rangle_{\partial \mathcal{T}_h} = (\boldsymbol{f}, \boldsymbol{w})_{\mathcal{T}_h}, \tag{30b}$$

$$\langle (\boldsymbol{F}(\hat{\boldsymbol{u}}_h) + \boldsymbol{G}(\hat{\boldsymbol{u}}_h, \boldsymbol{Q}_h)) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h), \boldsymbol{\mu} \rangle_{\partial \mathcal{T}_h \setminus \partial\Omega} + \langle \hat{\boldsymbol{b}}_h, \boldsymbol{\mu} \rangle_{\partial\Omega} = 0, \tag{30c}$$

for all $(\boldsymbol{E}, \boldsymbol{w}, \boldsymbol{\mu}) \in \mathcal{Q}_h^k \times \mathcal{V}_h^k \times \boldsymbol{\mathcal{M}}_h^k$. This completes the definition of the HDG method for solving the Navier–Stokes Eq. (25).

We shall not discuss the implementation of the HDG method described here because it is similar to that of the HDG method for the Euler Eq. (10). However, we would like to emphasize that in the Navier–Stokes case we also solve a matrix system involving the degrees of freedom of $\hat{\boldsymbol{u}}_h$ at every Newton iteration as we did for the Euler case. Moreover, the HDG matrix for the Navier–Stokes case has the same structure as the matrix for the Euler case.

3.4.2. Wall boundary conditions

The inlet and outlet boundary conditions are specified in the same manner as those of the Euler case. However, at the solid surface with no slip condition, we need to extrapolate the density, impose zero velocity, and set either a fixed temperature $T = T_w$ (isothermal wall) or zero heat flux $\partial T/\partial \boldsymbol{n} = 0$ (adiabatic wall). The strategy to impose these conditions is described as follows. First, we set

$$\hat{b}_{h1} = u_{h1} - \hat{u}_{h1}, \tag{31}$$

which means that we extrapolate the density. We then set

$$\hat{b}_{hi} = -\hat{u}_{hi}, \quad 2 \leqslant i \leqslant m-1, \tag{32}$$

which means that we impose $\hat{u}_{hi} = 0,\ 2 \leqslant i \leqslant m-1$, at the viscous solid wall. For the last component of $\hat{\boldsymbol{b}}_h$, we need to distinguish between the isothermal wall and the adiabatic wall. For the isothermal wall, we set

$$\hat{b}_{hm} = T_w - \widehat{T}_h(\hat{\boldsymbol{u}}_h), \tag{33}$$

where the approximate trace of the temperature $\widehat{T}_h(\hat{\boldsymbol{u}}_h)$ is computed from $\hat{\boldsymbol{u}}_h$. For the adiabatic wall, we set

$$\hat{b}_{hm} = \hat{h}_{hm}, \tag{34}$$

which means that we impose zero heat flux at the solid wall.

3.4.3. Stabilization matrix

We add to the stabilization matrix of the HDG method for the Euler equations a viscous stabilization matrix which is equal to

$$\boldsymbol{S}_v = \frac{\gamma}{PrRe}\boldsymbol{I}, \tag{35}$$

where Re is the Reynolds number, Pr is the Prandtl number, and γ is the heat capacity ratio. For compressible viscous flows the Reynolds number is typically very large and hence the contribution of the viscous component may be very small.

However, in some cases where the Reynolds number is small, the viscous component plays an important role in the accuracy of the method.

3.4.4. Numerical examples

The first example is aimed at verifying the accuracy and convergence of the method for the compressible Navier–Stokes equations. We consider a compressible Couette flow with a source term on a square domain ($0 \leqslant x_1 \leqslant H$ and $0 \leqslant x_2 \leqslant H$). The exact solution is given by

$$v_1 = \bar{x}_2 \log(1+\bar{x}_2), \quad v_2 = 0, \quad p = p_\infty \quad \text{and} \quad T = T_0 + \bar{x}_2(T_1 - T_0) + \frac{\gamma-1}{2\gamma p_\infty} Pr \bar{x}_2(1-\bar{x}_2),$$

where $\bar{x}_2 = x_2/H$. The density is then given by $\rho = 1/T$. The source term is determined from the exact solution as

$$\boldsymbol{f} = \left(0, -\frac{\bar{x}_2+2}{(\bar{x}_2+1)^2}, 0, \frac{3\log(\bar{x}_2+1) + \bar{x}_2(2\log(\bar{x}_2+1)+2)+1}{(\bar{x}_2+1)^2} - 3\log(\bar{x}_2+1) - \log(\bar{x}_2+1)^2\right).$$

For our numerical experiments we take $H = 1$, $T_0 = 0.8$ and $T_1 = 0.85$, $Pr = 0.72$ and $p_\infty = 1/\left(\gamma M_\infty^2\right)$ with $M_\infty = 0.15$. In order to asses the accuracy of the computed solution we calculate the L^2-norm of the error in the density ρ, the linear momentum vector $\boldsymbol{p} = (\rho v_1, \rho v_2)$, the energy ρE, the heat flux $\boldsymbol{q} = \nabla T$, and the viscous stress tensor $\boldsymbol{\tau}$ for different mesh sizes and different polynomial orders. We display the computed errors and orders of convergence for the approximations to these quantities in Table 4 for the Roe stabilization matrix (18) added with a viscous stabilization (35). We observe that all the approximate quantities converge with the optimal order of accuracy of $k+1$ in the L^2 norm. To the best of our knowledge, the HDG method is the first known method that provides the optimal order of convergence for the viscous stress and the heat flux.

In the second example we consider a laminar flow over a Kármán–Trefftz airfoil. The geometry of the airfoil and the mesh are the same as those used for the Euler example described above. The characteristics of the flow are $M_\infty = 0.1$, $Re = 4000$ and $Pr = 0.72$. Since the Reynolds number is large, the viscous stabilization in (35) is relatively small. Fig. 4 shows the Mach contours of the solution computed using a fourth-order polynomial approximation. We observe that a high-order element is actually sufficient to capture the boundary layer. In order to verify that this is a grid converged solution we show in Fig. 5 the distribution of pressure and skin friction coefficient using polynomial approximations of $k = 2$, 3 and 4. It can be readily observed that differences in the surface quantities computed for the different orders of approximation are very small.

In the last example we consider a steady laminar flow past a NACA0012 foil at $M_\infty = 0.85$, $Re = 1000$, and $\alpha = 5$ degrees. Fig. 6 shows the finite element mesh and Mach number contour. The results are computed using 160 elements and polynomial degree $k = 6$. We see that accurate solution can be obtained using very few high-order elements.

Table 4
History of convergence of the HDG method for the compressible Couette flow.

Degree	Mesh	$\|\rho - \rho_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{p} - \boldsymbol{p}_h\|_{\mathcal{T}_h}$		$\|\rho E - \rho E_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{\tau} - \boldsymbol{\tau}_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{q} - \boldsymbol{q}_h\|_{\mathcal{T}_h}$	
k	$1/h$	Error	Order	Error	Order	Error	Order	Error	Order	Error	Order
1	2	9.50e−4	–	1.21e−2	–	1.65e−2	–	3.59e−2	–	4.87e−3	–
	4	1.56e−4	2.60	3.15e−3	1.95	4.32e−3	1.93	1.07e−2	1.74	1.3e−3	1.91
	8	2.62e−5	2.58	8.05e−4	1.97	9.66e−4	2.16	2.96e−3	1.86	2.95e−4	2.14
	16	4.71e−6	2.48	2.04e−4	1.98	2.06e−4	2.23	7.95e−4	1.89	6.16e−5	2.26
	32	9.55e−7	2.30	5.13e−5	1.99	4.56e−5	2.17	2.13e−4	1.90	1.30e−5	2.25
	64	2.21e−7	2.11	1.29e−5	1.99	1.09e−5	2.06	5.72e−5	1.89	2.91e−6	2.16
2	2	4.69e−5	–	6.84e−4	–	1.97e−3	–	2.60e−3	–	2.77e−4	–
	4	6.59e−6	2.83	9.22e−5	2.89	3.55e−4	2.47	4.44e−4	2.55	3.50e−5	2.98
	8	9.79e−7	2.75	1.18e−5	2.97	5.92e−5	2.58	7.05e−5	2.66	3.92e−6	3.16
	16	1.60e−7	2.62	1.49e−6	2.99	1.01e−5	2.55	1.10e−5	2.69	4.10e−7	3.26
	32	2.69e−8	2.57	1.88e−7	2.99	1.72e−6	2.55	1.70e−6	2.69	4.36e−8	3.23
	64	4.45e−9	2.60	2.37e−8	2.99	2.85e−7	2.59	2.61e−7	2.70	4.93e−9	3.14
3	2	4.06e−6	–	4.61e−5	–	1.40e−4	–	2.10e−4	–	2.66e−5	–
	4	2.68e−7	3.92	3.30e−6	3.81	1.21e−5	3.53	1.95e−5	3.43	2.51e−6	3.41
	8	1.60e−8	4.06	2.11e−7	3.97	8.92e−7	3.76	1.59e−6	3.61	1.68e−7	3.90
	16	9.98e−10	4.01	1.30e−8	4.02	6.16e−8	3.86	1.21e−7	3.72	9.02e−9	4.22
	32	6.42e−11	3.96	8.03e−10	4.02	4.09e−9	3.91	8.74e−9	3.79	4.22e−10	4.42
	64	4.12e−12	3.96	4.96e−11	4.02	2.64e−10	3.95	6.01e−10	3.86	1.92e−11	4.46
4	2	3.45e−7	–	3.50e−6	–	1.53e−5	–	1.99e−5	–	2.36e−6	–
	4	1.66e−8	4.38	1.33e−7	4.71	9.43e−7	4.02	1.08e−6	4.20	9.4e−8	4.65
	8	7.51e−10	4.46	4.33e−9	4.94	4.67e−8	4.34	4.82e−8	4.49	2.6e−9	5.17
	16	3.07e−11	4.61	1.35e−10	5.01	1.95e−9	4.58	1.88e−9	4.68	6.64e−11	5.29
	32	1.15e−12	4.74	4.16e−12	5.01	7.35e−11	4.73	6.75e−11	4.80	1.87e−12	5.15

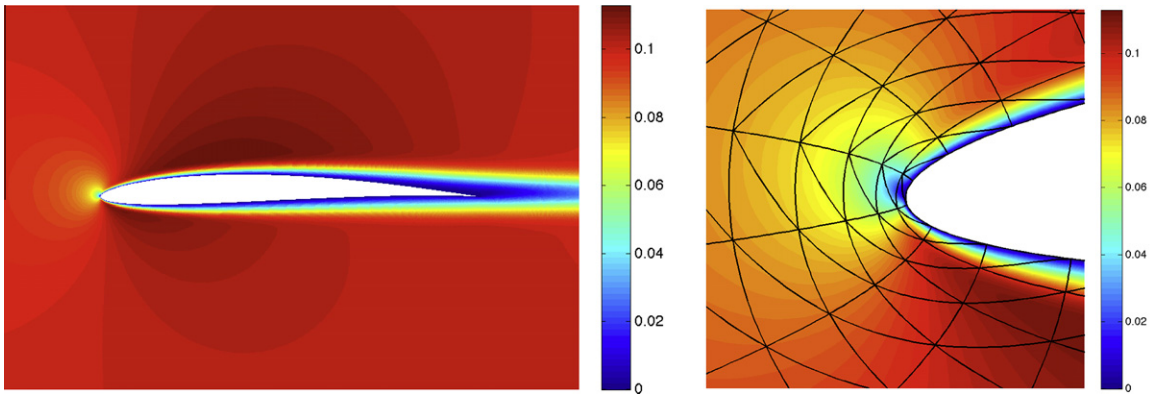

Fig. 4. Laminar flow over a Kármán-Trefftz airfoil at $M_\infty = 0.1$, $Re = 4000$ and $\alpha = 0$: Mach number distribution (left) and detail of the mesh and Mach number solution near the leading edge region (right) using fourth order polynomial approximations.

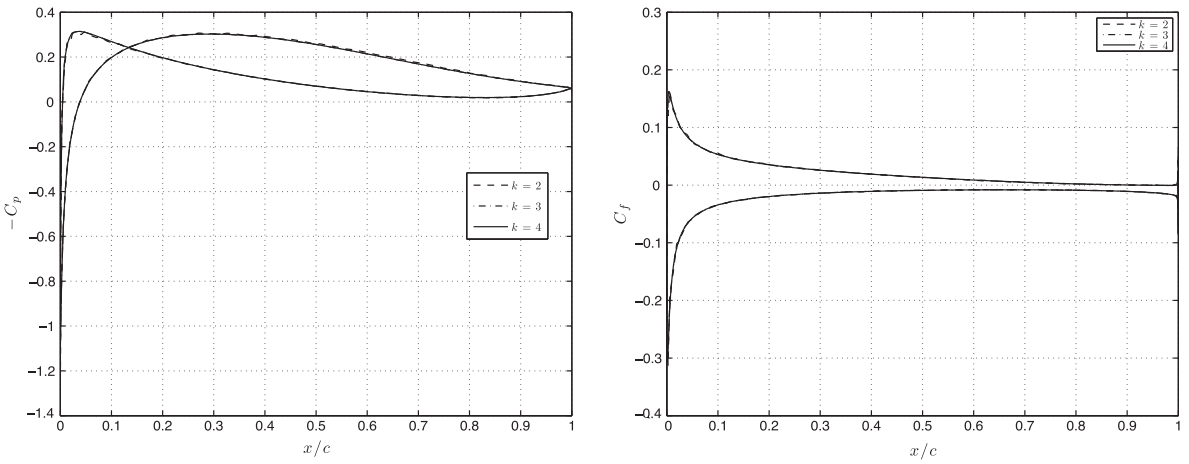

Fig. 5. Steady laminar flow over a Kármán–Trefftz airfoil at $M_\infty = 0.1$, $Re = 4000$ and $\alpha = 0$: pressure coefficient distribution (left) and skin friction coefficient (right) over the airfoil surface.

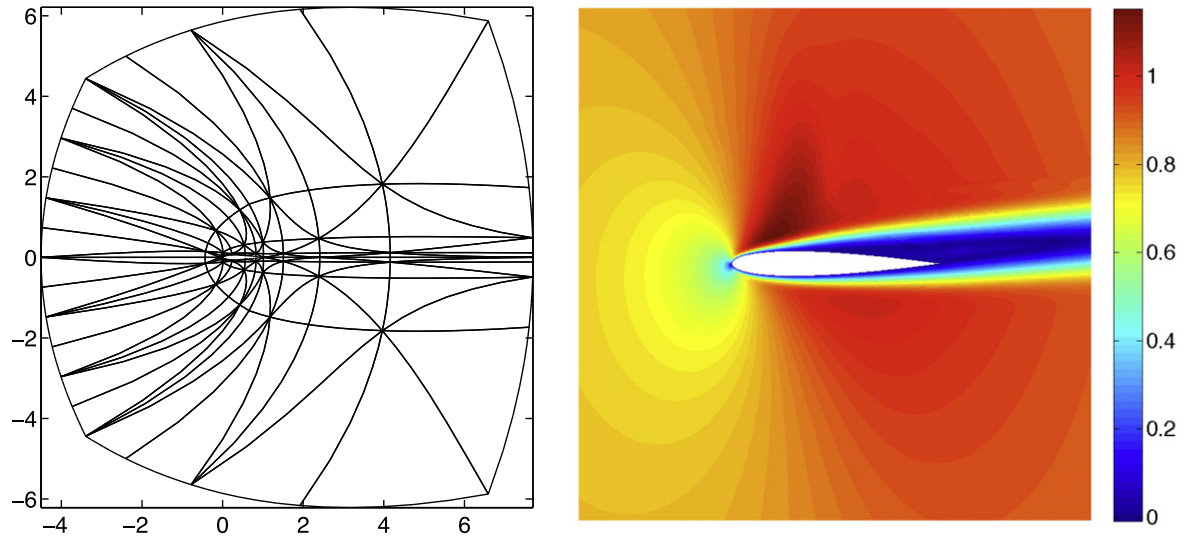

Fig. 6. Steady laminar flow over a NACA 0012 airfoil at $M_\infty = 0.85$, $Re = 1000$ and $\alpha = 5$: finite element mesh (160 elements of order $k = 6$) (left) and Mach number contour (right).

3.5. Time-dependent PDEs

In this subsection we extend the HDG method described earlier to time-dependent PDEs in fluid mechanics. We shall focus solely on the unsteady compressible Navier–Stokes equations as the reader will see that the same approach can be applied to other time-dependent PDEs.

3.5.1. The unsteady compressible Navier–Stokes equations

We consider the unsteady compressible Navier–Stokes equations written as a first-order system of equations

$$\begin{aligned} &\boldsymbol{Q} - \nabla \boldsymbol{u} = 0, \quad \text{in } \Omega \times (0,T], \\ &\frac{\partial \boldsymbol{u}}{\partial t} + \nabla \cdot (\boldsymbol{F}(\boldsymbol{u}) + \boldsymbol{G}(\boldsymbol{u},\boldsymbol{Q})) = \boldsymbol{f}, \quad \text{in } \Omega \times (0,T], \end{aligned} \tag{36}$$

with an initial condition $\boldsymbol{u}(\boldsymbol{x}, t=0) = \boldsymbol{u}_0(\boldsymbol{x})$, $\forall \boldsymbol{x} \in \Omega$, and appropriate boundary conditions as stated in the steady case.

3.5.2. Time integration using the backward difference formulas

For simplicity of exposition, we shall consider the backward Euler method for time integration as higher-order backward difference formula (BDF) schemes admit a similar implementation. We denote by $(\boldsymbol{Q}_h^n, \boldsymbol{u}_h^n, \hat{\boldsymbol{u}}_h^n)$ the numerical approximations to $(\boldsymbol{Q}(t^n), \boldsymbol{u}(t^n), \hat{\boldsymbol{u}}(t^n))$ at time $t^n = n\Delta t^n$, where Δt^n is a timestep size at level n. Using the HDG method to discretize (36) in space and the backward Euler method to discretize the time derivative, we obtain that $(\boldsymbol{Q}_h^n, \boldsymbol{u}_h^n, \hat{\boldsymbol{u}}_h^n) \in \mathcal{Q}_h^k \times \mathcal{V}_h^k \times \mathcal{M}_h^k$ satisfies

$$(\boldsymbol{Q}_h^n, \boldsymbol{E})_{\mathcal{T}_h} + (\boldsymbol{u}_h^n, \nabla \cdot \boldsymbol{E})_{\mathcal{T}_h} - \langle \hat{\boldsymbol{u}}_h^n, \boldsymbol{E} \cdot \boldsymbol{n} \rangle_{\partial \mathcal{T}_h} = 0, \tag{37a}$$

$$\left(\frac{\boldsymbol{u}_h^n}{\Delta t^n}, \boldsymbol{w}\right)_{\mathcal{T}_h} - \left(\boldsymbol{F}(\boldsymbol{u}_h^n) + \boldsymbol{G}(\boldsymbol{u}_h^n, \boldsymbol{Q}_h^n), \nabla \boldsymbol{w}\right)_{\mathcal{T}_h} + \left\langle \left(\boldsymbol{F}(\hat{\boldsymbol{u}}_h^n) + \boldsymbol{G}(\hat{\boldsymbol{u}}_h^n, \boldsymbol{Q}_h^n)\right) \cdot \boldsymbol{n} + \boldsymbol{S}^n(\boldsymbol{u}_h^n - \hat{\boldsymbol{u}}_h^n), \boldsymbol{w} \right\rangle_{\partial \mathcal{T}_h} = \left(\boldsymbol{f}^n + \frac{\boldsymbol{u}_h^{n-1}}{\Delta t^n}, \boldsymbol{w}\right)_{\mathcal{T}_h}, \tag{37b}$$

$$\left\langle \left(\boldsymbol{F}(\hat{\boldsymbol{u}}_h^n) + \boldsymbol{G}(\hat{\boldsymbol{u}}_h^n, \boldsymbol{Q}_h^n)\right) \cdot \boldsymbol{n} + \boldsymbol{S}^n(\boldsymbol{u}_h^n - \hat{\boldsymbol{u}}_h^n), \boldsymbol{\mu} \right\rangle_{\partial \mathcal{T}_h \setminus \partial \Omega} + \left\langle \hat{\boldsymbol{b}}_h^n, \boldsymbol{\mu} \right\rangle_{\partial \Omega} = 0 \tag{37c}$$

for all $(\boldsymbol{E}, \boldsymbol{w}, \boldsymbol{\mu}) \in \mathcal{Q}_h^k \times \mathcal{V}_h^k \times \mathcal{M}_h^k$ and $n \geqslant 1$, where $\left(\boldsymbol{Q}_h^0, \boldsymbol{u}_h^0, \hat{\boldsymbol{u}}_h^0\right)$ is the L^2 projection of the initial data $(\boldsymbol{Q}_0, \boldsymbol{u}_0, \boldsymbol{u}_0)$ onto $\mathcal{Q}_h^k \times \mathcal{V}_h^k \times \mathcal{M}_h^k$.

The stabilization matrix and the boundary numerical flux are specified in the same way as for the steady case. Hence, the system (37) is almost identical to the HDG system (30) of the steady case except for the two additional terms due to the temporal discretization of the time derivative. This implies that it can be solved in the same way as for the steady Euler case. We further note that the use of higher-order backward difference formula (BDF) schemes such as the second-order accurate BDF2 scheme and the third-order accurate BDF3 scheme gives rise to a HDG formulation as similar as (37). The HDG method can also work with other implicit time-stepping methods such as the diagonally implicit Runge–Kutta (DIRK) methods as discussed below.

3.5.3. Time integration using the diagonally implicit Rung-Kutta methods

Let us consider the following DIRK (q,p) formulas [1] written in the form of Butcher's table for time integration

$$\begin{array}{cccccc} a_{11} & 0 & & \cdots & 0 & c_1 \\ a_{21} & a_{22} & 0 & \cdots & 0 & c_2 \\ \cdots & & & \cdots & & \cdots \\ a_{q1} & a_{q2} & & \cdots & a_{qq} & c_q \\ \hline b_1 & b_2 & & \cdots & b_q & \end{array} \tag{38}$$

where q denotes the number of stages and p denotes the order of the method. It is clear from the above table that every stage of the DIRK schemes is nothing but a backward-Euler solve by using the Gauss elimination to factorize the lower triangular matrix. We are now ready to describe the HDG method for spatial discretization and the DIRK(q,p) method for time integration.

To simplify notation we denote by $t^{n,i} \equiv t^n + c_i \Delta t^n$, $1 \leqslant i \leqslant q$, and $\left(\boldsymbol{Q}_h^{n,i}, \boldsymbol{u}_h^{n,i}, \hat{\boldsymbol{u}}_h^{n,i}\right) \equiv (\boldsymbol{Q}_h(t^{n,i}), \boldsymbol{u}_h(t^{n,i}), \hat{\boldsymbol{u}}_h(t^{n,i})), 1 \leqslant i \leqslant q$, which are the intermediate solutions. The numerical solution $\boldsymbol{u}_h^{n+1}$ at time level $n+1$ given by the DIRK(q,p) method is computed as follows:

$$\boldsymbol{u}_h^{n+1} = \boldsymbol{u}_h^n + \Delta t^n \sum_{i=1}^{q} b_i \boldsymbol{z}_h^{n,i} \tag{39}$$

where

$$\begin{aligned}
\boldsymbol{z}_h^{n,1} &= \frac{\boldsymbol{u}_h^{n,1} - \boldsymbol{u}_h^n}{a_{11}\Delta t^n},\\
\boldsymbol{z}_h^{n,2} &= \frac{\boldsymbol{u}_h^{n,2} - \boldsymbol{u}_h^n}{a_{22}\Delta t^n} - \frac{a_{21}}{a_{22}}\boldsymbol{z}_h^{n,1},\\
&\cdots\\
\boldsymbol{z}_h^{n,q} &= \frac{\boldsymbol{u}_h^{n,q} - \boldsymbol{u}_h^n}{a_{qq}\Delta t^n} - \sum_{j=1}^{q-1}\frac{a_{qj}}{a_{qq}}\boldsymbol{z}_h^{n,j}.
\end{aligned} \tag{40}$$

The intermediate solutions are determined as follows: $\left(\boldsymbol{Q}_h^{n,i}, \boldsymbol{u}_h^{n,i}, \hat{\boldsymbol{u}}_h^{n,i}\right) \in \mathcal{Q}_h^k \times \mathcal{V}_h^k \times \mathcal{M}_h^k$ satisfies

$$\left(\boldsymbol{Q}_h^{n,i}, \boldsymbol{E}\right)_{\mathcal{T}_h} + \left(\boldsymbol{u}_h^{n,i}, \nabla\cdot\boldsymbol{E}\right)_{\mathcal{T}_h} - \left\langle \hat{\boldsymbol{u}}_h^{n,i}, \boldsymbol{E}\cdot\boldsymbol{n}\right\rangle_{\partial\mathcal{T}_h} = 0, \tag{41a}$$

$$\begin{aligned}
&\left(\frac{\boldsymbol{u}_h^{n,i}}{a_{ii}\Delta t^n}, \boldsymbol{w}\right)_{\mathcal{T}_h} - \left(\boldsymbol{F}\left(\boldsymbol{u}_h^{n,i}\right) + \boldsymbol{G}\left(\boldsymbol{u}_h^{n,i}, \boldsymbol{Q}_h^{n,i}\right), \nabla\boldsymbol{w}\right)_{\mathcal{T}_h}\\
&\quad + \left\langle \left(\boldsymbol{F}\left(\hat{\boldsymbol{u}}_h^{n,i}\right) + \boldsymbol{G}\left(\hat{\boldsymbol{u}}_h^{n,i}, \boldsymbol{Q}_h^{n,i}\right)\right)\cdot\boldsymbol{n} + \boldsymbol{S}^{n,i}\left(\boldsymbol{u}_h^{n,i} - \hat{\boldsymbol{u}}_h^{n,i}\right), \boldsymbol{w}\right\rangle_{\partial\mathcal{T}_h} = \left(\boldsymbol{f}^{n,i} + \boldsymbol{s}_h^{n,i}, \boldsymbol{w}\right)_{\mathcal{T}_h},
\end{aligned} \tag{41b}$$

$$\left\langle \left(\boldsymbol{F}\left(\hat{\boldsymbol{u}}_h^{n,i}\right) + \boldsymbol{G}\left(\hat{\boldsymbol{u}}_h^{n,i}, \boldsymbol{Q}_h^{n,i}\right)\right)\cdot\boldsymbol{n} + \boldsymbol{S}^{n,i}\left(\boldsymbol{u}_h^{n,i} - \hat{\boldsymbol{u}}_h^{n,i}\right), \boldsymbol{\mu}\right\rangle_{\partial\mathcal{T}_h\setminus\partial\Omega} + \left\langle \hat{\boldsymbol{b}}_h^{n,i}, \boldsymbol{\mu}\right\rangle_{\partial\Omega} = 0, \tag{41c}$$

for all $(\boldsymbol{E}, \boldsymbol{w}, \boldsymbol{\mu}) \in \mathcal{Q}_h^k \times \mathcal{V}_h^k \times \mathcal{M}_h^k$, where the terms $\boldsymbol{s}_h^{n,i}, 1 \leqslant i \leqslant q$, on the right-hand side of (41b) are given by

$$\begin{aligned}
\boldsymbol{s}_h^{n,1} &= \frac{\boldsymbol{u}_h^n}{a_{11}\Delta t^n},\\
\boldsymbol{s}_h^{n,2} &= \frac{\boldsymbol{u}_h^n}{a_{22}\Delta t^n} + \frac{a_{21}}{a_{22}}\left(\frac{\boldsymbol{u}_h^{n,1}}{a_{11}\Delta t^n} - \boldsymbol{s}_h^{n,1}\right),\\
&\cdots\\
\boldsymbol{s}_h^{n,q} &= \frac{\boldsymbol{u}_h^n}{a_{qq}\Delta t^n} + \sum_{j=1}^{q-1}\frac{a_{qj}}{a_{qq}}\left(\frac{\boldsymbol{u}_h^{n,j}}{a_{jj}\Delta t^n} - \boldsymbol{s}_h^{n,j}\right).
\end{aligned} \tag{42}$$

We note that the resulting system (41) at each ith stage of the DIRK (q,p) method is very similar to the system (37) of the Backward-Euler method. Once $\boldsymbol{u}_h^{n+1}$ has been determined as above, we can compute $\left(\boldsymbol{Q}_h^{n+1}, \hat{\boldsymbol{u}}_h^{n+1}\right) \in \mathcal{Q}_h^k \times \mathcal{M}_h^k$ such that

$$\left(\boldsymbol{Q}_h^{n+1}, \boldsymbol{E}\right)_{\mathcal{T}_h} + \left(\boldsymbol{u}_h^{n+1}, \nabla\cdot\boldsymbol{E}\right)_{\mathcal{T}_h} - \left\langle \hat{\boldsymbol{u}}_h^{n+1}, \boldsymbol{E}\cdot\boldsymbol{n}\right\rangle_{\partial\mathcal{T}_h} = 0, \tag{43a}$$

$$\left\langle \left(\boldsymbol{F}\left(\hat{\boldsymbol{u}}_h^{n+1}\right) + \boldsymbol{G}\left(\hat{\boldsymbol{u}}_h^{n+1}, \boldsymbol{Q}_h^{n+1}\right)\right)\cdot\boldsymbol{n} + \boldsymbol{S}^{n+1}\left(\boldsymbol{u}_h^{n+1} - \hat{\boldsymbol{u}}_h^{n+1}\right), \boldsymbol{\mu}\right\rangle_{\partial\mathcal{T}_h\setminus\partial\Omega} + \left\langle \hat{\boldsymbol{b}}_h^{n+1}, \boldsymbol{\mu}\right\rangle_{\partial\Omega} = 0, \tag{43b}$$

for all $(\boldsymbol{E}, \boldsymbol{\mu}) \in \mathcal{Q}_h^k \times \mathcal{M}_h^k$. The system (43) can be solved by eliminating $\boldsymbol{Q}_h^{n+1}$ to obtain a global reduced system in terms of the degrees of freedom of $\hat{\boldsymbol{u}}_h^{n+1}$.

We see that the HDG-DIRK (q,p) method requires q backward-Euler solves (41) and one additional solve (43) for every timestep. This additional solve comes from the fact that the HDG method gives rise to a differential algebraic system of index 1. In practice, however, we do not need to solve (43) for every timestep because we can advance (39)–(41) in time without $\left(\boldsymbol{Q}_h^{n+1}, \hat{\boldsymbol{u}}_h^{n+1}\right)$. Rather we compute $\left(\boldsymbol{Q}_h^{n+1}, \hat{\boldsymbol{u}}_h^{n+1}\right)$ only when they are needed. Finally, we note that some special DIRK schemes such as the strongly S-stable DIRK (2,2) and DIRK (3,3) schemes [1] have $c_q = 1$, which means that the Eqs. (43a) and (43b) coincide with (41a) and (41c), respectively, at the last stage $i = q$. Hence, these DIRK schemes do not need to solve (43).

3.5.4. Numerical examples

The first test case for unsteady flow computations is the incompressible viscous flow passing a circular cylinder at the Reynolds number $Re = 200$ [37]. When the Reynolds number is above 50 and less than 1000, the flow will undergo separations but still maintain laminar. The induced vortices are then shed from upper and lower surfaces alternatively to form the famous Kármán vortex street. The problem setting is as follows. We consider the flow domain in a rectangle defined by $-10 \leqslant x_1 \leqslant 20$ and $-10 \leqslant x_2 \leqslant 10$. The center of the cylinder coincides with the origin of the coordinate system, and its diameter D is a unit value. The incoming free-stream flow U_0 is assumed to be uniform with a unit magnitude, so that the Dirichlet boundary condition $\boldsymbol{u} = (1,0)$ is imposed at the inlet $x_1 = -10$. No-slip boundary condition is imposed on the cylinder surface. The remaining boundary is prescribed with zero stress, that is, $(-\nu(\nabla\boldsymbol{u} + \nabla\boldsymbol{u}^T) + p\boldsymbol{I})\cdot\boldsymbol{n} = 0$. The Reynolds number is defined based on the free-stream velocity and the cylinder diameter $Re = U_0 D/\nu$.

Fig. 7 depicts the FE mesh used in the computation. The mesh is refined along the cylinder to resolve the boundary layer as well as in the wave region to capture the vortex street behind the cylinder. We use $k = 4$ for spatial discretization and the DIRK (3,3) scheme with a timestep size $\Delta t = 0.2$ for time integration. We present in Fig. 8 the horizontal velocity and vorticity at time $t = 100$. The Kármán vortex street is clearly observed. Fig. 9 shows the time history of drag and lift coefficients. The magnitude of the lift coefficient C_L is $\pm$ 0.69, while the magnitude of the drag coefficient C_D is 1.3564 $\pm$ 0.0462. The calculated

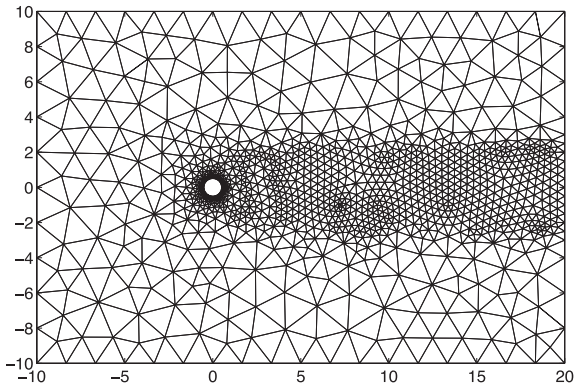

Fig. 7. Finite element mesh of 2494 elements for the circular cylinder example.

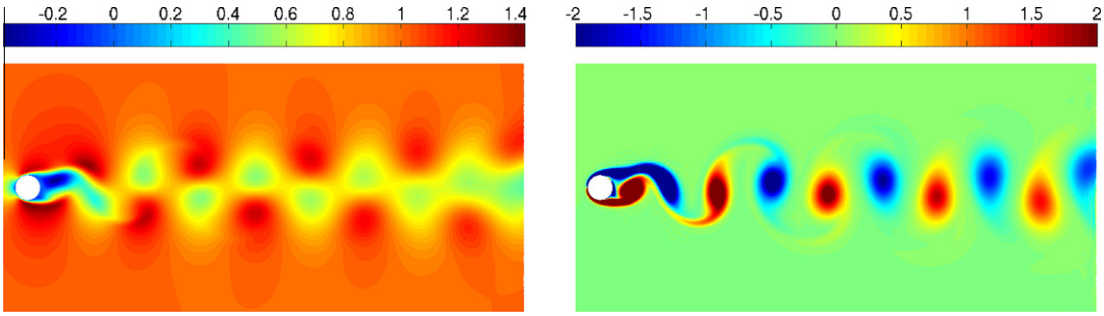

Fig. 8. Horizontal velocity and vorticity at $t = 100$ for incompressible viscous flow past a circular cylinder at $Re = 200$.

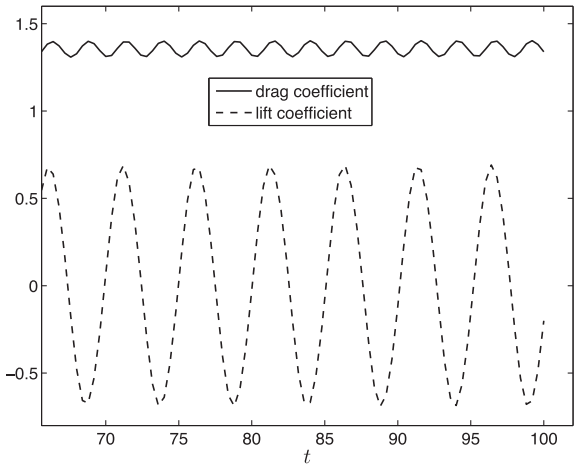

Fig. 9. Time history of lift and drag coefficients for incompressible viscous flow past a circular cylinder at $Re = 200$. The Strouhal number is 0.2.

Strouhal number is 0.2. These figures agree well with previous numerical results as well as with experimental measurements (see [37]).

The second test case is the compressible viscous flow passing a SD7003 foil [66] at the Reynolds number $Re = 10{,}000$, $M_\infty = 0.2$, and $\alpha = 4^o$. The FE mesh is shown in Fig. 10. We use $k = 4$ for spatial discretization and the DIRK (3,3) scheme with a timestep size $\Delta t = 0.025$ for time integration. We observe the vortex shedding behind the airfoil as depicted in Fig. 11. However, as it can be seen from the time history of drag and lift coefficients shown in Fig. 12, the close-to-periodic vortex shedding has much higher frequency than that of the previous test case. The average lift coefficient is 0.378 and the average drag coefficient is 0.0502. Our results also agree well with [66].

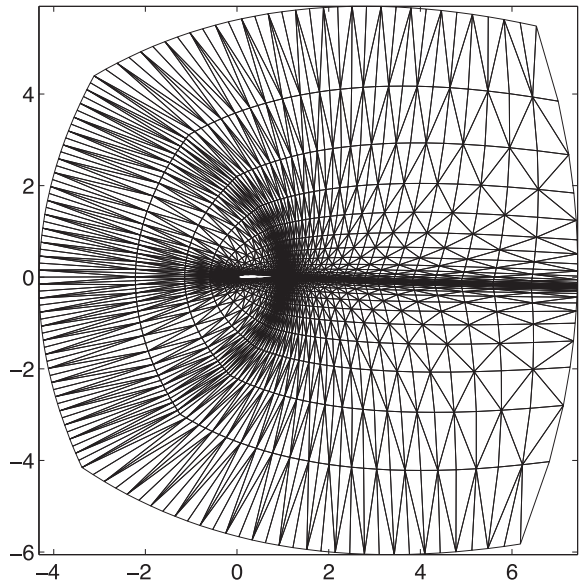

Fig. 10. Finite element mesh of 3360 elements for the SD7003 airfoil example.

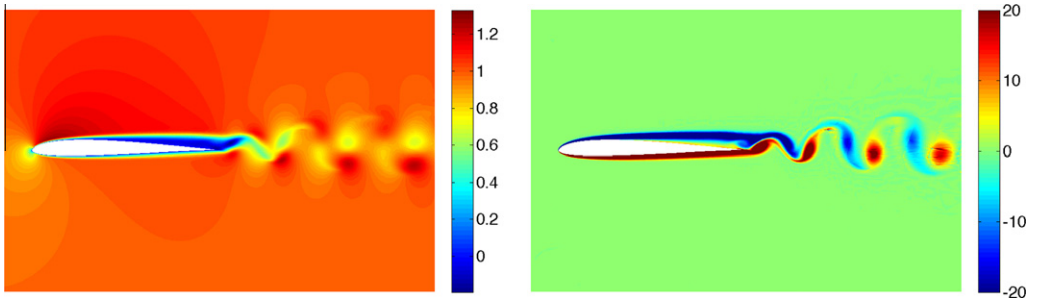

Fig. 11. Horizontal velocity and vorticity at $t = 15$ for compressible viscous flow past a SD7003 foil at $Re = 10^4$, $M_\infty = 0.2$, and $\alpha = 4^o$.

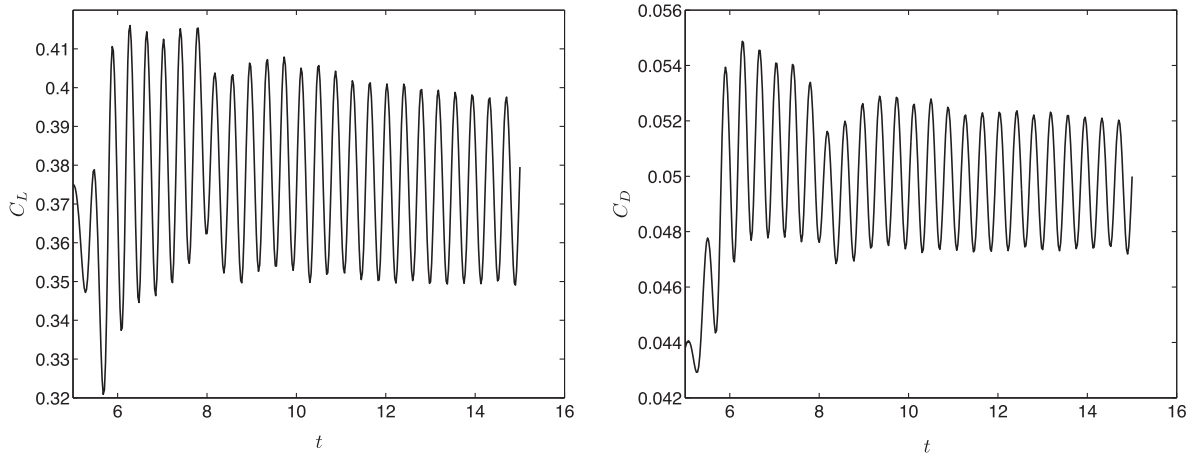

Fig. 12. Time history of lift and drag coefficients for compressible viscous flow past a SD7003 foil at $Re = 10^4$, $M_\infty = 0.2$, and $\alpha = 4^o$.

3.6. Conservation laws in moving domain

In this subsection we develop HDG methods for solving systems of conservation laws in moving domain. We begin by denoting the moving domain by $\Omega_{\boldsymbol{y}}(t) \in \mathbb{R}^d$ and its spatial coordinates by $\boldsymbol{y}$. In this domain we consider a system of conservation laws written in the Eulerian framework as

$$\left.\frac{\partial \boldsymbol{s}}{\partial t}\right|_{\boldsymbol{y}} + \nabla_{\boldsymbol{y}} \cdot \boldsymbol{F}(\boldsymbol{s}, \nabla_{\boldsymbol{y}} \boldsymbol{s}) = 0, \quad \text{in } \Omega_{\boldsymbol{y}}(t) \times (0,T], \tag{44}$$

where $\boldsymbol{s} \in \mathbb{R}^m$ is a vector of m conserved variables and $\boldsymbol{F} \in \mathbb{R}^{m\times d}$ contains d flux vectors of dimension m. The flux vectors are functions of $\boldsymbol{s}$ and $\nabla_{\boldsymbol{y}} s$, where $\nabla_{\boldsymbol{y}}$ is the gradient operator in the $\boldsymbol{y}$ coordinates. Examples of systems of conservation laws include the Euler equations, the Navier–Stokes equations, and the equations of solid motion.

The arbitrary Lagragian Eulerian (ALE) technique [21,22,39,56,67] is the most popular method to account for the time variation of the physical domain. This is done by introducing the velocity of the physical domain into the governing equations. Within the ALE framework one may choose to carry out the discretization of the resulting ALE equations either on the moving domain or a fixed reference domain. The discretization on a fixed reference domain is adopted here because this approach is shown to work well for high-order methods [56,67].

3.6.1. Arbitrary Lagrangian Eulerian formulation

Let us denote by Ω a *fixed* reference domain with spatial coordinate $\boldsymbol{x}$. The motion of the time-dependent domain $\Omega_{\boldsymbol{y}}(t)$ is defined by a deformation mapping $\boldsymbol{\phi}$ of the form $\boldsymbol{y} = \boldsymbol{\phi}(\boldsymbol{x})$. The deformation gradient and grid velocity are then given by

$$\boldsymbol{G} = \nabla_{\boldsymbol{x}} \boldsymbol{\phi}, \quad \boldsymbol{v}^g = \left.\frac{\partial \boldsymbol{\phi}}{\partial t}\right|_{\boldsymbol{x}}, \quad J = \det(\boldsymbol{G}). \tag{45}$$

We first apply the ALE method (see [56]) to the original system (44) in $\Omega_{\boldsymbol{y}}(t)$ to obtain a new system in Ω as

$$\left.\frac{\partial J\boldsymbol{s}}{\partial t}\right|_{\boldsymbol{x}} + \nabla_{\boldsymbol{x}} \cdot (J(\boldsymbol{F}(\boldsymbol{s}, \nabla_{\boldsymbol{y}} \boldsymbol{s}) - \boldsymbol{s} \otimes \boldsymbol{v}^g)\boldsymbol{G}^{-T}) = 0, \quad \text{in } \Omega \times (0,T]. \tag{46}$$

We next introduce a new vector $\boldsymbol{u} = J\boldsymbol{s}$ and thus get

$$\nabla_{\boldsymbol{y}} \boldsymbol{s} = (\nabla_{\boldsymbol{x}} \boldsymbol{s})\boldsymbol{G}^{-1} = (\nabla_{\boldsymbol{x}} (\boldsymbol{u}/J))\boldsymbol{G}^{-1} = (J^{-1}\nabla_{\boldsymbol{x}} \boldsymbol{u} + \boldsymbol{u} \otimes \nabla_{\boldsymbol{x}} J^{-1})\boldsymbol{G}^{-1}. \tag{47}$$

As a result, we can rewrite the above system in terms of the new vector $\boldsymbol{u}$ as

$$\left.\frac{\partial \boldsymbol{u}}{\partial t}\right|_{\boldsymbol{x}} + \nabla_{\boldsymbol{x}} \cdot \boldsymbol{F}_{\boldsymbol{x}}(\boldsymbol{u}, \nabla_{\boldsymbol{x}} \boldsymbol{u}) = 0, \quad \Omega \times [0,T], \tag{48}$$

where

$$\boldsymbol{F}_{\boldsymbol{x}}(\boldsymbol{u}, \nabla_{\boldsymbol{x}} \boldsymbol{u}) = J(\boldsymbol{F}(J^{-1}\boldsymbol{u}, (J^{-1}\nabla_{\boldsymbol{x}} \boldsymbol{u} + \boldsymbol{u} \otimes \nabla_{\boldsymbol{x}} J^{-1})\boldsymbol{G}^{-1}) - J^{-1}\boldsymbol{u} \otimes \boldsymbol{v}^g)\boldsymbol{G}^{-T}. \tag{49}$$

The system (48) and (49) represents the transformed conservation laws in terms of $\boldsymbol{u}$ in the reference domain Ω. Since this system can readily be discretized using the HDG method developed in the previous subsection, we omit the detailed steps to save space. We mention however that the initial and boundary conditions must also be transformed to the reference domain and hence they must be imposed on $\boldsymbol{u}$ and $\boldsymbol{F}_{\boldsymbol{x}}$. This step is straightforward and thus omitted here.

If the mapping $\boldsymbol{\phi}$ can be described by some analytical expression of the spatial coordinates $\boldsymbol{x}$ and time t, one can directly determine the grid velocity, the deformation gradient and its determinant from (45). If this is not the case then one needs to numerically compute $\boldsymbol{\phi}$. Although there exists a variety of methods for computing mesh deformation, we find such discussion beyond the scope of this paper. Below we will consider an example in which that the mapping is known analytically to illustrate the HDG method.

3.6.2. Heaving and pitching NACA 0012 foil

We consider the compressible viscous flow around a heaving and pitching NACA0012 airfoil. The motion of the airfoil is described by the following mapping

$$\begin{bmatrix} \phi_1(\boldsymbol{x},t) \\ \phi_2(\boldsymbol{x},t) \end{bmatrix} = \begin{bmatrix} \cos(\theta(t)) & -\sin(\theta(t)) \\ \sin(\theta(t)) & \cos(\theta(t)) \end{bmatrix} \begin{bmatrix} x_1 - 0.5 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0.5 \\ y(t) \end{bmatrix} \tag{50}$$

where

$$y(t) = A\sin(2\pi f t), \quad \theta(t) = -B\sin(2\pi f t). \tag{51}$$

We choose the free-stream velocity $U_0 = 1$ in the x_1-direction, the oscillating frequency $f = 0.2$, and amplitudes $A = 0.5$ and $B = \pi/12$. The free-stream Mach number of the flow is 0.2 and the Reynolds number is 5000.

The reference domain and associated computational grid is shown in Fig. 13. We use polynomials of degree $k = 4$ within each element to represent the solution. For temporal discretization we employ the DIRK (3,3) scheme with a variable time-step size $\Delta t^n = 0.0005 + (n-1)(0.025 - 0.0005)/9$ for $n = 1,2,\ldots,10$ and $\Delta t^n = 0.025$ for $n \geqslant 10$. We show in Fig. 14 the vortical structure of the flow around the foil at different times in the simulation. We observe dominant vortex structures below the airfoil when it is pitching up and above the airfoil when it is pitching down. These vortex structures are typical of an airfoil in heaving and pitching motion.

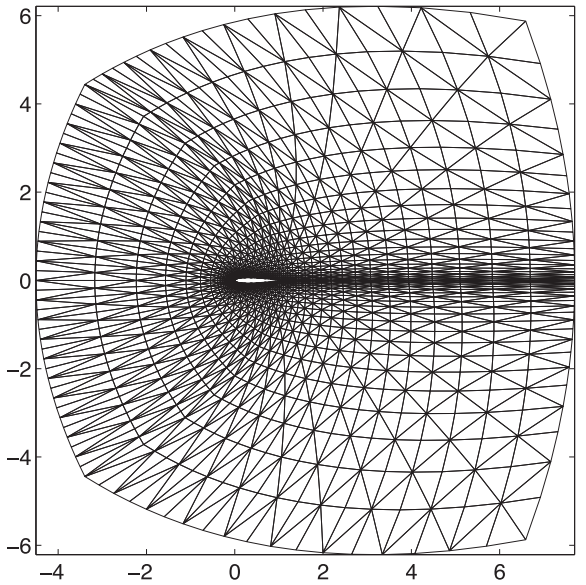

Fig. 13. Finite element mesh for the reference domain of the moving NACA 0012 airfoil example.

4. HDG methods for PDEs in solid mechanics

In this section we develop HDG methods for solving static and dynamical systems of partial differential equations in solid mechanics. We begin with introducing a HDG method for linear elasticity. We next extend this method to nonlinear elasticity with large deformation modeled by hyperelastic materials. We then focus on devising HDG methods for nonlinear elastodynamics. The methods we develop can be applied to compressible, nearly incompressible, and incompressible solids. Furthermore, they possess several attractive features which render them ideally suited for solving many problems in solid mechanics. We provide numerical results to illustrate the main characteristics of the HDG methods and compare their performance with that of the standard continuous Galerkin method in the linear case.

4.1. Linear elasticity

4.1.1. Governing equations

Let $\boldsymbol{u}$ represent the vector field of the displacement components, λ and μ the Lamé moduli, ρ the density of the elastic isotropic material, and $\boldsymbol{f}$ a body force. The small deformation of the elastic isotropic body Ω is governed by:

$$-\nabla\cdot[\mu\nabla\boldsymbol{u}+(\mu+\lambda)(\nabla\cdot\boldsymbol{u})\boldsymbol{I}]=\boldsymbol{f},\quad \text{in } \Omega. \tag{52}$$

We introduce the displacement gradient tensor $\boldsymbol{H}=\nabla\boldsymbol{u}$ and the hydrostatic pressure $p=-(\mu+\lambda)(\nabla\cdot\boldsymbol{u})$. We then rewrite (52) as a first-order system of equations

$$\begin{aligned}&\boldsymbol{H}-\nabla\boldsymbol{u}=0,\quad \text{in } \Omega,\\ &-\nabla\cdot(\mu\boldsymbol{H}-p\boldsymbol{I})=\boldsymbol{f},\quad \text{in } \Omega,\\ &\epsilon p+\nabla\cdot\boldsymbol{u}=0,\quad \text{in } \Omega,\end{aligned} \tag{53}$$

with $\epsilon=1/(\mu+\lambda)$. Note that the linear elasticity Eq. (53) are similar to the Stokes Eq. (2) except that (53) has an additional pressure term ϵp in the third equation. Indeed, in the incompressible case (i.e., $\epsilon=0$), the two systems are exactly identical. The boundary conditions are given as

$$\begin{aligned}&\boldsymbol{u}=\boldsymbol{g}_D,\quad \text{on } \partial\Omega_D,\\ &\boldsymbol{B}\boldsymbol{n}=\boldsymbol{g}_N,\quad \text{on } \partial\Omega_N,\end{aligned} \tag{54}$$

where $\partial\Omega_N$ is a part of the boundary $\partial\Omega$ such that $\partial\Omega_N\cup\partial\Omega_D=\partial\Omega$ and $\partial\Omega_N\cap\partial\Omega_D=\emptyset$. Here $\boldsymbol{B}$ is a linear operator that depends on $(\boldsymbol{H},p)$. Examples of the form of $\boldsymbol{B}$ are given in Table 5.

4.1.2. Formulation

The HDG method seeks an approximation $(\boldsymbol{H}_h,\boldsymbol{u}_h,p_h,\hat{\boldsymbol{u}}_h)\in\boldsymbol{\mathcal{Q}}_h^k\times\boldsymbol{\mathcal{V}}_h^k\times\mathcal{W}_h^k\times\boldsymbol{\mathcal{M}}_h^k$ such that

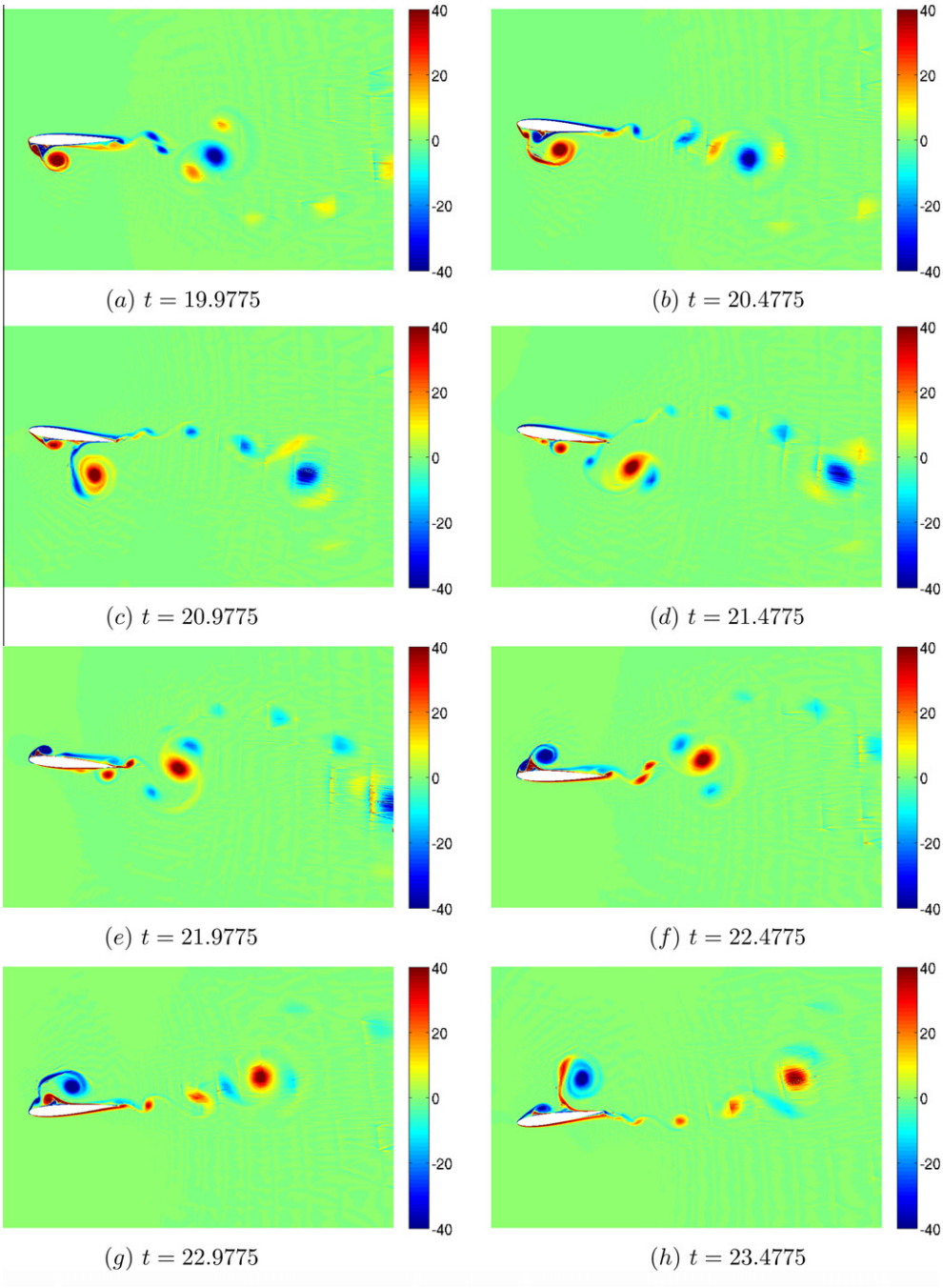

Fig. 14. Vorticity at different times in the simulation for compressible viscous flow past a moving NACA 0012 airfoil at $Re = 5 \times 10^3$ and $M_\infty = 0.2$.

$$(\boldsymbol{H}_h, \boldsymbol{E})_{\mathcal{T}_h} + (\boldsymbol{u}_h, \nabla \cdot \boldsymbol{E})_{\mathcal{T}_h} - \langle \hat{\boldsymbol{u}}_h, \boldsymbol{E} \cdot \boldsymbol{n} \rangle_{\partial \mathcal{T}_h} = 0, \tag{55a}$$

$$(\mu \boldsymbol{H}_h - p_h \boldsymbol{I}, \nabla \boldsymbol{w})_{\mathcal{T}_h} + \langle \hat{\boldsymbol{h}}_h, \boldsymbol{w} \rangle_{\partial \mathcal{T}_h} = (\boldsymbol{f}, \boldsymbol{w})_{\mathcal{T}_h}, \tag{55b}$$

$$(\epsilon p_h, \nabla q)_{\mathcal{T}_h} - (\boldsymbol{u}_h, \nabla q)_{\mathcal{T}_h} + \langle \hat{\boldsymbol{u}}_h \cdot \boldsymbol{n}, q \rangle_{\partial \mathcal{T}_h} = 0, \tag{55c}$$

$$\langle \hat{\boldsymbol{h}}_h, \boldsymbol{\mu} \rangle_{\partial \mathcal{T}_h \setminus \partial \Omega} + \langle \hat{\boldsymbol{u}}_h - \boldsymbol{g}_D, \boldsymbol{\mu} \rangle_{\partial \Omega_D} + \langle \hat{\boldsymbol{b}}_h - \boldsymbol{g}_N, \boldsymbol{\mu} \rangle_{\partial \Omega_N} = 0, \tag{55d}$$

for all $(\boldsymbol{E}, \boldsymbol{w}, q, \boldsymbol{\mu}) \in \mathcal{Q}_h^k \times \mathcal{V}_h^k \times \mathcal{W}_h^k \times \mathcal{M}_h^k$, where the numerical flux $\hat{\boldsymbol{h}}_h$ is given by

$$\hat{\boldsymbol{h}}_h = (-\mu \boldsymbol{H}_h + p_h \boldsymbol{I}) \cdot \boldsymbol{n} + \boldsymbol{S}(\boldsymbol{u}_h - \hat{\boldsymbol{u}}_h), \tag{55e}$$

Table 5
Examples of other boundary conditions for the linear elasticity equations. Note that the asterisk symbol * indicates that the average pressure condition $(p_h,1)_{\Omega}=0$ is also imposed when $\epsilon=0$. The dagger symbol † indicates that a Dirichlet boundary condition for the normal component of the velocity has also to be provided on $\partial\Omega_N$.

Condition type	$\boldsymbol{B}$	$\hat{\boldsymbol{b}}_h$
Stress	$-\mu(\boldsymbol{H}+\boldsymbol{H}^T)+\epsilon\lambda p\boldsymbol{I}$	$(-\mu\left(\boldsymbol{H}_h+\boldsymbol{H}_h^T\right)+\epsilon\lambda p_h\boldsymbol{I})\cdot\boldsymbol{n}+\boldsymbol{S}(\boldsymbol{u}_h-\hat{\boldsymbol{u}}_h)$
Viscous stress*	$-\mu(\boldsymbol{H}+\boldsymbol{H}^T)$	$-\mu(\boldsymbol{H}_h+\boldsymbol{H}_h^T)\cdot\boldsymbol{n}+\boldsymbol{S}(\boldsymbol{u}_h-\hat{\boldsymbol{u}}_h)$
Vorticity + pressure	$-\mu(\boldsymbol{H}-\boldsymbol{H}^T)+\epsilon\lambda p\boldsymbol{I}$	$(-\mu\left(\boldsymbol{H}_h-\boldsymbol{H}_h^T\right)+\epsilon\lambda p_h\boldsymbol{I})\cdot\boldsymbol{n}+\boldsymbol{S}(\boldsymbol{u}_h-\hat{\boldsymbol{u}}_h)$
Vorticity*,†	$-\mu(\boldsymbol{H}-\boldsymbol{H}^T)$	$-\mu(\boldsymbol{H}_h-\boldsymbol{H}_h^T)\cdot\boldsymbol{n}+\boldsymbol{S}(\boldsymbol{u}_h-\hat{\boldsymbol{u}}_h)$
Gradient + pressure	$-\mu\boldsymbol{H}+\epsilon\lambda p\boldsymbol{I}$	$(-\mu\boldsymbol{H}_h+\epsilon\lambda p_h\boldsymbol{I})\cdot\boldsymbol{n}+\boldsymbol{S}(\boldsymbol{u}_h-\hat{\boldsymbol{u}}_h)$
Gradient*	$-\mu\boldsymbol{H}$	$-\mu\boldsymbol{H}_h\cdot\boldsymbol{n}+\boldsymbol{S}(\boldsymbol{u}_h-\hat{\boldsymbol{u}}_h)$

and the boundary numerical flux $\hat{\boldsymbol{b}}_h$ is given in Table 5. The stabilization matrix is chosen as

$$\boldsymbol{S}=\frac{\mu}{\ell}\boldsymbol{I}, \tag{56}$$

where ℓ is a characteristic length scale. Note that both the Dirichlet and Neumann boundary conditions have been enforced weakly by the Eq. (55d) of (55).

The HDG method for linear elasticity can be implemented by following the implementation steps of the HDG method for Stokes flow [17,49]. In fact, for the incompressible case $\epsilon=0$, the two methods are identical. For the compressible case $\epsilon>0$, the HDG method for linear elasticity does not need the augmented Lagrangian treatment. In such case, we solve the system (55) by reducing it to a global weak formulation in terms of the approximate trace $\hat{\boldsymbol{u}}_h$ only. This results in considerable savings for both the computational cost and memory storage.

4.1.3. Local postprocessing

We describe here a simple yet effective postprocessing procedure proposed in [63] and employed in [44] to improve the numerical approximation of the displacement field. In particular, the new approximate displacement $\boldsymbol{u}_h^*\in(\mathcal{P}^{k+1}(K))^d$ satisfies, on every simplex $K\in\mathcal{T}_h$,

$$\begin{aligned}&(\nabla\boldsymbol{u}_h^*,\nabla\boldsymbol{w})_K=(\boldsymbol{H}_h,\nabla\boldsymbol{w})_K,\quad\forall\boldsymbol{w}\in(\mathcal{P}^{k+1}(K))^d,\\&(\boldsymbol{u}_h^*,1)_K=(\boldsymbol{u}_h,1)_K.\end{aligned} \tag{57}$$

Note that each component of $\boldsymbol{u}_h^*$ can be solved independently of each other. Moreover, since the local postprocessing is performed at the element level, the postprocesssed displacement $\boldsymbol{u}_h^*$ are less expensive to compute than the original displacement $\boldsymbol{u}_h$.

4.1.4. A numerical example

This example is studied in [62]. We consider the elastic Eq. (53) on a unit square $\Omega=(0,1)\times(0,1)$ with $\mu=1$. The problem has the following exact solution

$$\begin{aligned}u_1&=-x^2y(2y-1)(x-1)^2(y-1),\\u_2&=xy^2(2x-1)(x-1)(y-1)^2.\end{aligned}$$

The source term $\boldsymbol{f}$ is determined from the above solution. The Dirichlet boundary data are determined as the restriction of the exact solution on the boundary. The stabilization tensor is set to the identity tensor. The FE meshes are triangular and constructed upon regular $n\times n$ Cartesian grids ($h=1/n$).

We present the L^2 errors and orders of convergence of the numerical approximations in Table 6 for $\lambda=1$ (compressible case as the Poisson ratio $\nu=0.25$) and in Table 7 for $\lambda=1000$ (nearly incompressible case as $\nu\approx0.4995$) for both the HDG method and the standard CG method. We observe that the approximate displacement and gradient converge with the optimal order $k+1$ even for the nearly incompressible case. Note that the approximate stress derived from the approximate displacement gradient also converges with the optimal order $k+1$. Optimal convergence of the stress is an important advantage of the HDG method because many quantities of engineering interest involve the stress. Furthermore, we observe that the postprocessed displacement $\boldsymbol{u}_h^*$ converges with order $k+2$ which are one order higher than the original approximation. Since the postprocessed displacement is inexpensive to compute, the HDG method provides better convergence and accuracy than the standard CG method for the same mesh and polynomial degree k.

Table 6
Comparison of the convergence of the $L^2(\Omega)$ errors in the displacement and displacement gradient between the HDG method and the standard CG method for a linear elasticity example with $\mu = 1$ and $\lambda = 1$.

Degree	Mesh	HDG						Standard CG			
		$\|\boldsymbol{u}-\boldsymbol{u}_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{H}-\boldsymbol{H}_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{u}-\boldsymbol{u}_h^*\|_{\mathcal{T}_h}$		$\|\boldsymbol{u}-\boldsymbol{u}_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{H}-\boldsymbol{H}_h\|_{\mathcal{T}_h}$	
k	n	Error	Order	Error	Order	Error	Order	Error	Order	Error	Order
1	2	5.71e−3	–	5.98e−3	–	7.02e−4	–	3.78e−3	–	2.72e−2	–
	4	1.50e−3	1.93	1.66e−3	1.85	1.06e−4	2.72	1.80e−3	1.07	1.81e−2	0.59
	8	3.94e−4	1.93	4.46e−4	1.90	1.46e−5	2.86	6.33e−4	1.51	9.85e−3	0.88
	16	1.01e−4	1.97	1.16e−4	1.94	1.90e−6	2.95	1.79e−4	1.82	4.98e−3	0.98
	32	2.55e−5	1.98	2.95e−5	1.97	2.41e−7	2.98	4.64e−5	1.95	2.49e−3	1.00
2	2	1.07e−3	–	1.98e−3	–	1.65e−4	–	1.44e−3	–	1.50e−2	–
	4	1.95e−4	2.46	2.86e−4	2.79	1.41e−5	3.55	2.13e−4	2.76	4.81e−3	1.64
	8	2.76e−5	2.82	3.67e−5	2.96	9.69e−7	3.86	2.31e−5	3.21	1.30e−3	1.89
	16	3.59e−6	2.94	4.55e−6	3.01	6.20e−8	3.97	2.46e−6	3.23	3.29e−4	1.98
	32	4.55e−7	2.98	5.64e−7	3.01	3.89e−9	3.99	2.86e−7	3.10	8.23e−5	2.00
3	2	3.53e−4	–	6.04e−4	–	4.53e−5	–	2.35e−4	–	4.45e−3	–
	4	3.10e−5	3.51	4.31e−5	3.81	1.70e−6	4.73	2.34e−5	3.33	8.60e−4	2.37
	8	2.13e−6	3.86	2.83e−6	3.93	5.52e−8	4.95	1.32e−6	4.14	1.16e−4	2.90
	16	1.38e−7	3.95	1.81e−7	3.97	1.73e−9	5.00	7.41e−8	4.16	1.45e−5	2.99
	32	8.72e−9	3.98	1.14e−8	3.99	5.38e−11	5.01	4.31e−9	4.10	1.80e−6	3.01
4	2	7.79e−5	–	1.87e−4	–	1.10e−5	–	7.92e−5	–	1.71e−3	–
	4	2.98e−6	4.71	6.50e−6	4.85	1.90e−7	5.85	3.00e−6	4.72	1.36e−4	3.65
	8	9.96e−8	4.90	2.10e−7	4.95	3.02e−9	5.97	9.79e−8	4.94	8.99e−6	3.92
	16	3.19e−9	4.96	6.64e−9	4.98	4.74e−11	6.00	3.08e−9	4.99	5.69e−7	3.98
	32	1.01e−10	4.98	2.09e−10	4.99	7.40e−13	6.00	9.60e−11	5.00	3.57e−8	4.00

Table 7
Comparison of the convergence of the $L^2(\Omega)$ errors in the displacement and displacement gradient between the HDG method and the standard CG method for a linear elasticity example with $\mu = 1$ and $\lambda = 1000$.

Degree	Mesh	HDG						Standard CG			
		$\|\boldsymbol{u}-\boldsymbol{u}_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{H}-\boldsymbol{H}_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{u}-\boldsymbol{u}_h^*\|_{\mathcal{T}_h}$		$\|\boldsymbol{u}-\boldsymbol{u}_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{H}-\boldsymbol{H}_h\|_{\mathcal{T}_h}$	
k	n	Error	Order	Error	Order	Error	Order	Error	Order	Error	Order
1	2	5.81e−3	–	7.51e−3	–	7.84e−4	–	3.78e−3	–	2.72e−2	–
	4	1.51e−3	1.94	2.17e−3	1.79	1.19e−4	2.72	3.84e−3	−0.02	2.83e−2	−0.05
	8	3.96e−4	1.94	5.90e−4	1.88	1.64e−5	2.85	3.74e−3	0.04	2.75e−2	0.04
	16	1.01e−4	1.97	1.53e−4	1.95	2.13e−6	2.94	3.36e−3	0.15	2.48e−2	0.15
	32	2.55e−5	1.99	3.88e−5	1.98	2.71e−7	2.98	2.47e−3	0.45	1.85e−2	0.42
2	2	1.08e−3	–	2.31e−3	–	1.73e−4	–	3.83e−3	–	2.83e−2	–
	4	1.97e−4	2.46	4.03e−4	2.52	1.59e−5	3.44	1.27e−3	1.60	1.56e−2	0.86
	8	2.77e−5	2.83	5.47e−5	2.88	1.09e−6	3.86	2.85e−4	2.15	7.10e−3	1.13
	16	3.59e−6	2.94	6.97e−6	2.97	6.96e−8	3.97	5.76e−5	2.31	2.86e−3	1.31
	32	4.55e−7	2.98	8.74e−7	2.99	4.36e−9	4.00	8.92e−6	2.69	8.76e−4	1.71
3	2	3.58e−4	–	8.43e−4	–	4.79e−5	–	5.98e−4	–	9.07e−3	–
	4	3.12e−5	3.52	6.40e−5	3.72	1.80e−6	4.74	9.76e−5	2.62	2.65e−3	1.77
	8	2.14e−6	3.87	4.23e−6	3.92	5.79e−8	4.96	1.07e−5	3.19	5.92e−4	2.16
	16	1.38e−7	3.95	2.69e−7	3.97	1.81e−9	5.00	9.99e−7	3.42	1.12e−4	2.41
	32	8.72e−9	3.98	1.69e−8	3.99	5.62e−11	5.01	6.67e−8	3.91	1.50e−5	2.90
4	2	7.87e−5	–	2.22e−4	–	1.13e−5	–	2.86e−4	–	4.66e−3	–
	4	3.00e−6	4.71	7.74e−6	4.84	1.96e−7	5.85	1.14e−5	4.64	3.97e−4	3.55
	8	9.98e−8	4.91	2.50e−7	4.95	3.12e−9	5.97	3.10e−7	5.21	2.34e−5	4.09
	16	3.20e−9	4.96	7.89e−9	4.98	4.89e−11	5.99	8.03e−9	5.27	1.31e−6	4.16
	32	1.01e−10	4.98	2.47e−10	4.99	7.65e−13	6.00	2.15e−10	5.22	7.49e−8	4.13

4.2. Nonlinear elasticity

4.2.1. Governing equations

We consider large deformation of an elastic continuum body defined by a deformation mapping $\boldsymbol{\varphi}$ between a reference configuration Ω and a current configuration Ω_c of the form $\boldsymbol{y} = \boldsymbol{\varphi}(\boldsymbol{x})$. Here, $\boldsymbol{x}$ is the coordinate in the reference configuration Ω and $\boldsymbol{y}$ denotes the position of particle $\boldsymbol{x}$ in subsequent deformations. Let $\boldsymbol{f}$ be the body force per unit reference volume. The boundary $\partial\Omega$ is divided into two complementary disjoint parts $\partial\Omega_D$ and $\partial\Omega_N$, where the prescribed deformation $\boldsymbol{g}_D$ and traction $\boldsymbol{g}_N$ are imposed, respectively. The continuum problem is governed by the following equations stated in Lagrangian form

$$\begin{aligned} &\boldsymbol{F} - \nabla \boldsymbol{\varphi} = 0, \quad \text{in } \Omega, \\ &-\nabla \cdot \boldsymbol{P} = \boldsymbol{f}, \quad \text{in } \Omega, \\ &\boldsymbol{\varphi} = \boldsymbol{g}_D, \quad \text{on } \partial\Omega_D, \\ &\boldsymbol{P}\boldsymbol{n} = \boldsymbol{g}_N, \quad \text{on } \partial\Omega_N. \end{aligned} \tag{58}$$

This system of equations defines an equilibrium state of the elastic body under large deformation. Unless otherwise indicated, the gradient and divergence operators are taken with respect to the coordinate $\boldsymbol{x}$ of the reference configuration. (In the literature, it is customary to using $\boldsymbol{X}$ to denote the coordinates of the reference configuration and $\boldsymbol{x}$ to denote the coordinates of the deformed configuration. However, in order to maintain consistency in our use of notation throughout the paper, we choose to use $\boldsymbol{x}$ for the coordinates of the reference configuration.)

For simple reversible elasticity the first Piola–Kirchhoff stress tensor $\boldsymbol{P}$ is derived as the derivative of a strain energy function ψ with respect to the deformation gradient $\boldsymbol{F}$, namely,

$$\boldsymbol{P} = \frac{\partial \psi(\boldsymbol{F})}{\partial \boldsymbol{F}}. \tag{59}$$

It is customary to decoupling the strain energy function ψ into isochoric and volumetric components

$$\psi = \psi_{\text{iso}}(\boldsymbol{F}) + \psi_{\text{vol}}(J), \tag{60}$$

where $J = \det(\boldsymbol{F})$ is the determinant of $\boldsymbol{F}$. This in turn leads to the following decomposition of the first Piola–Kirchhoff stress tensor

$$\boldsymbol{P} = \boldsymbol{D}(\boldsymbol{F}) + pJ\boldsymbol{F}^{-T}, \tag{61}$$

where

$$\boldsymbol{D}(\boldsymbol{F}) = \frac{\partial \psi_{\text{iso}}(\boldsymbol{F})}{\partial \boldsymbol{F}}, \quad p = \lambda U(J), \qquad U(J) = \frac{1}{\lambda}\frac{\partial \psi_{\text{vol}}(J)}{\partial J}. \tag{62}$$

Note that λ is the bulk modulus and that $U(J) = 0$ for $J = 1$. Once ψ_{iso} and ψ_{vol} are given, $\boldsymbol{P}$ can be determined by (61) and (62).

In order to deal with nearly incompressible and incompressible materials, we introduce p as an independent variable to obtain the following system

$$\begin{aligned} &\boldsymbol{F} - \nabla \boldsymbol{\varphi} = 0, \quad \text{in } \Omega, \\ &-\nabla \cdot \boldsymbol{P} = \boldsymbol{f}, \quad \text{in } \Omega, \\ &\varepsilon p - U(J) = 0, \quad \text{in } \Omega, \\ &\boldsymbol{\varphi} = \boldsymbol{g}_D, \quad \text{on } \partial\Omega_D, \\ &\boldsymbol{P}\boldsymbol{n} = \boldsymbol{g}_N, \quad \text{on } \partial\Omega_N, \end{aligned} \tag{63}$$

where $\varepsilon = 1/\lambda$. For incompressible elastic materials ($\lambda \to \infty, \varepsilon = 0$) the equation $U(J) = 0$ yields $J = 1$ and thus imposes the incompressibility condition since the volume of the elastic body does not change during motion. Below we introduce a HDG method for solving the nonlinear elasticity Eq. (63).

4.2.2. Formulation

We seek an approximation $(\boldsymbol{F}_h, \boldsymbol{\varphi}_h, p_h, \hat{\boldsymbol{\varphi}}_h) \in \boldsymbol{\mathcal{Q}}_h^k \times \boldsymbol{\mathcal{V}}_h^k \times \mathcal{W}_h^k \times \boldsymbol{\mathcal{M}}_h^k$ such that

$$(\boldsymbol{F}_h, \boldsymbol{E})_{\mathcal{T}_h} + (\boldsymbol{\varphi}_h, \nabla \cdot \boldsymbol{E})_{\mathcal{T}_h} - \langle \hat{\boldsymbol{\varphi}}_h, \boldsymbol{E} \cdot \boldsymbol{n} \rangle_{\partial\mathcal{T}_h} = 0, \tag{64a}$$

$$(\boldsymbol{P}(\boldsymbol{F}_h, p_h), \nabla \boldsymbol{w})_{\mathcal{T}_h} - \langle \hat{\boldsymbol{h}}_h, \boldsymbol{w} \rangle_{\partial\mathcal{T}_h} - (\boldsymbol{f}, \boldsymbol{v})_{\mathcal{T}_h} = 0, \tag{64b}$$

$$(\varepsilon p_h, q)_{\mathcal{T}_h} - (U(\boldsymbol{F}_h), q)_{\mathcal{T}_h} = 0, \tag{64c}$$

$$\langle \hat{\boldsymbol{h}}_h, \boldsymbol{\mu} \rangle_{\partial\mathcal{T}_h \setminus \partial\Omega} + \langle \hat{\boldsymbol{\varphi}}_h - \boldsymbol{g}_D, \boldsymbol{\mu} \rangle_{\partial\Omega_D} + \langle \hat{\boldsymbol{h}}_h - \boldsymbol{g}_N, \boldsymbol{\mu} \rangle_{\partial\Omega_N} = 0, \tag{64d}$$

for all $(\boldsymbol{E}, \boldsymbol{w}, q, \boldsymbol{\mu}) \in \boldsymbol{\mathcal{Q}}_h^k \times \boldsymbol{\mathcal{V}}_h^k \times \mathcal{W}_h^k \times \boldsymbol{\mathcal{M}}_h^k$, where the numerical flux $\hat{\boldsymbol{h}}_h$ is given by

$$\hat{\boldsymbol{h}}_h = \boldsymbol{P}(\boldsymbol{F}_h, p_h) \cdot \boldsymbol{n} - \boldsymbol{S}(\boldsymbol{\varphi}_h - \hat{\boldsymbol{\varphi}}_h). \tag{64e}$$

Here the stabilization tensor $\boldsymbol{S}$ does have an important effect on both the stability and accuracy of the method. The selection of the value of the stabilization tensor will be described later. Recall that $\boldsymbol{P}$ and U are given by (61) and (62), respectively. We note that if $\varepsilon = 0$ (incompressible material) and $\partial\Omega_N$ is empty then p is only defined up to an additive constant. In such case we must impose the average pressure constraint $(p_h, 1)_{\mathcal{T}_h} = 0$ to ensure the well-posedness of the HDG method.

Let us briefly comment on the equations defining the HDG method. The first three Eqs. (64a)–(64c), (64c) are obtained by multiplying the governing Eq. (63) by test functions and integrating the resulting equations by parts. The fourth Eq. (64d) enforces the continuity of the L^2 projection of the numerical flux and imposes weakly the Dirichlet and Neumann boundary conditions. The last Eq. (64e) defines the numerical flux.

4.2.3. Stabilization matrix

In the linear elasticity case, the material elasticity tensor is symmetric positive-definite everywhere in the domain. However, in the nonlinear elasticity case, the material elasticity tensor is in general not symmetric positive-definite in regions where large deformation occurs. We simply make a minor modification of the stabilization formulas (56) for the nonlinear case as

$$\boldsymbol{S} = C\frac{\mu}{\ell}\boldsymbol{I}, \tag{65}$$

where the constant $C \geqslant 1$ is problem-dependent. In spite of its simplicity, the above stabilization tensor works well for many test cases.

4.2.4. Implementation

We first consider the compressible case corresponding to $\varepsilon > 0$. By applying the Newton–Raphson method to linearize the nonlinear system (64) we obtain that the Newton increment $(\delta\boldsymbol{F}_h^\ell, \delta\boldsymbol{\varphi}_h^\ell, \delta p_h^\ell, \delta\hat{\boldsymbol{\varphi}}_h^\ell) \in \boldsymbol{\mathcal{Q}}_h^k \times \boldsymbol{\mathcal{V}}_h^k \times \mathcal{W}_h^k \times \boldsymbol{\mathcal{M}}_h^k$ satisfies

$$(\delta\boldsymbol{F}_h^\ell, \boldsymbol{E})_{\mathcal{T}_h} + (\delta\boldsymbol{\varphi}_h^\ell, \nabla\cdot\boldsymbol{E})_{\mathcal{T}_h} - \langle\delta\hat{\boldsymbol{\varphi}}_h^\ell, \boldsymbol{E}\cdot\boldsymbol{n}\rangle_{\partial\mathcal{T}_h} = r_1(\boldsymbol{E}), \tag{66a}$$

$$((\partial_{\boldsymbol{F}_h}\boldsymbol{P})\delta\boldsymbol{F}_h^\ell + (\partial_{p_h}\boldsymbol{P})\delta p_h^\ell, \nabla\boldsymbol{w})_{\mathcal{T}_h} - \left\langle(\partial_{\boldsymbol{F}_h}\hat{\boldsymbol{h}}_h)\delta\boldsymbol{F}_h^\ell + (\partial_{\boldsymbol{\varphi}_h}\hat{\boldsymbol{h}}_h)\delta\boldsymbol{\varphi}_h^\ell + (\partial_{p_h}\hat{\boldsymbol{h}}_h)\delta p_h^\ell + (\partial_{\hat{\boldsymbol{\varphi}}_h}\hat{\boldsymbol{h}}_h)\delta\hat{\boldsymbol{\varphi}}_h^\ell, \boldsymbol{w}\right\rangle_{\partial\mathcal{T}_h} = r_2(\boldsymbol{w}), \tag{66b}$$

$$(\varepsilon\delta p_h^\ell, q)_{\mathcal{T}_h} - ((\partial_{\boldsymbol{F}_h}U)\delta\boldsymbol{F}_h^\ell, q)_{\mathcal{T}_h} = r_3(q), \tag{66c}$$

$$\left\langle(\partial_{\boldsymbol{F}_h}\hat{\boldsymbol{h}}_h)\delta\boldsymbol{F}_h^\ell + (\partial_{\boldsymbol{\varphi}_h}\hat{\boldsymbol{h}}_h)\delta\boldsymbol{\varphi}_h^\ell + (\partial_{p_h}\hat{\boldsymbol{h}}_h)\delta p_h^\ell + (\partial_{\hat{\boldsymbol{\varphi}}_h}\hat{\boldsymbol{h}}_h)\delta\hat{\boldsymbol{\varphi}}_h^\ell, \boldsymbol{\mu}\right\rangle_{\partial\mathcal{T}_h\setminus\partial\Omega_D} + \langle\delta\hat{\boldsymbol{\varphi}}_h^\ell, \boldsymbol{\mu}\rangle_{\partial\Omega_D} = r_4(\boldsymbol{\mu}), \tag{66d}$$

for all $(\boldsymbol{E}, \boldsymbol{w}, q, \boldsymbol{\mu}) \in \boldsymbol{\mathcal{Q}}_h^k \times \boldsymbol{\mathcal{V}}_h^k \times \mathcal{W}_h^k \times \boldsymbol{\mathcal{M}}_h^k$, where

$$r_1(\boldsymbol{E}) = -\Big((\boldsymbol{F}_h^\ell, \boldsymbol{E})_{\mathcal{T}_h} + (\boldsymbol{\varphi}_h^\ell, \nabla\cdot\boldsymbol{E})_{\mathcal{T}_h} - \langle\hat{\boldsymbol{\varphi}}_h^\ell, \boldsymbol{E}\cdot\boldsymbol{n}\rangle_{\partial\mathcal{T}_h}\Big), \tag{67a}$$

$$r_2(\boldsymbol{w}) = -\Big((\boldsymbol{P}(\boldsymbol{F}_h^\ell, p_h^\ell), \nabla\boldsymbol{w})_{\mathcal{T}_h} - \langle\hat{\boldsymbol{h}}_h^\ell, \boldsymbol{w}\rangle_{\partial\mathcal{T}_h} - (\boldsymbol{f}, \boldsymbol{v})_{\mathcal{T}_h}\Big), \tag{67b}$$

$$r_3(q) = -\Big((\varepsilon p_h^\ell, q)_{\mathcal{T}_h} - (U(\boldsymbol{F}_h^\ell), q)_{\mathcal{T}_h}\Big), \tag{67c}$$

$$r_3(\boldsymbol{\mu}) = -\left(\left\langle\hat{\boldsymbol{h}}_h^\ell, \boldsymbol{\mu}\right\rangle_{\partial\mathcal{T}_h\setminus\partial\Omega} + \langle\hat{\boldsymbol{\varphi}}_h^\ell - \boldsymbol{g}_D, \boldsymbol{\mu}\rangle_{\partial\Omega_D} + \left\langle\hat{\boldsymbol{h}}_h^\ell - \boldsymbol{g}_N, \boldsymbol{\mu}\right\rangle_{\partial\Omega_N}\right), \tag{67d}$$

are the residual functionals evaluated at the current iterate $(\boldsymbol{F}_h^\ell, \boldsymbol{\varphi}_h^\ell, p_h^\ell, \hat{\boldsymbol{\varphi}}_h^\ell)$. In (66), $(\partial_{\boldsymbol{F}_h}\boldsymbol{P})$ and $(\partial_{p_h}\boldsymbol{P})$ denote the partial derivatives of $\boldsymbol{P}$ with respect to $\boldsymbol{F}_h$ and p_h, respectively. The same notation is applied to the partial derivatives of $\hat{\boldsymbol{h}}_h$. Moreover, all the partial derivatives are evaluated at the current iterate $(\boldsymbol{F}_h^\ell, \boldsymbol{\varphi}_h^\ell, p_h^\ell, \hat{\boldsymbol{\varphi}}_h^\ell)$. We next update the numerical approximations as

$$\left(\boldsymbol{F}_h^{\ell+1}, \boldsymbol{\varphi}_h^{\ell+1}, p_h^{\ell+1}, \hat{\boldsymbol{\varphi}}_h^{\ell+1}\right) := (\boldsymbol{F}_h^\ell, \boldsymbol{P}_h^\ell, p_h^\ell, \hat{\boldsymbol{\varphi}}_h^\ell) + \alpha(\delta\boldsymbol{F}_h^\ell, \delta\boldsymbol{\varphi}_h^\ell, \delta p_h^\ell, \delta\hat{\boldsymbol{\varphi}}_h^\ell),$$

where the parameter α is determined by the line search algorithm to ensure a sufficient decrease in the residual. We then increment $\ell$and repeat the Newton–Raphson process until convergence.

Although it is large, the linearized problem (66) can be solved very efficiently by using local elimination. In particular, the matrix form of the linearized problem (66) is

$$\begin{bmatrix}\mathbf{A}^\ell & \mathbf{B}^\ell\\ \mathbf{C}^\ell & \mathbf{D}^\ell\end{bmatrix}\begin{pmatrix}\mathbf{a}^\ell\\ \mathbf{b}^\ell\end{pmatrix} = \begin{pmatrix}\mathbf{f}^\ell\\ \mathbf{g}^\ell\end{pmatrix}, \tag{68}$$

where $\mathbf{a}^\ell$ and $\mathbf{b}^\ell$ are the vectors of degrees of freedom of $(\delta\boldsymbol{F}_h^\ell, \delta\boldsymbol{\varphi}_h^\ell, \delta p_h^\ell)$ and $\delta\hat{\boldsymbol{\varphi}}_h^\ell$, respectively. It should be clear that the matrix $\mathbf{A}^\ell$ has a block-diagonal structure due to the discontinuous nature of the approximation spaces. Therefore, we can eliminate $\mathbf{a}^\ell$ to obtain a reduced system in terms of $\mathbf{b}^\ell$ as

$$\mathbf{K}^\ell\mathbf{b}^\ell = \mathbf{r}^\ell, \tag{69}$$

where

$$\mathbf{K}^\ell = -\mathbf{C}^\ell(\mathbf{A}^\ell)^{-1}\mathbf{B}^\ell + \mathbf{D}^\ell, \quad \mathbf{r}^\ell = \mathbf{g}^\ell - \mathbf{C}^\ell(\mathbf{A}^\ell)^{-1}\mathbf{f}^\ell. \tag{70}$$

Note that $(\mathbf{A}^\ell)^{-1}$ is also a block-diagonal matrix. To form $\mathbf{K}^\ell$ we does not need to explicitly compute the matrices $\mathbf{A}^\ell$, $\mathbf{B}^\ell$, $\mathbf{C}^\ell$, and $\mathbf{D}^\ell$. Instead we compute the elemental matrices and elemental vectors over each element as

$$\mathbf{K}_K^\ell = -\mathbf{C}_K^\ell\left(\mathbf{A}_K^\ell\right)^{-1}\mathbf{B}_K^\ell + \mathbf{D}_K^\ell, \quad \mathbf{r}_K^\ell = \mathbf{g}_K^\ell - \mathbf{C}_K^\ell\left(\mathbf{A}_K^\ell\right)^{-1}\mathbf{f}_K^\ell, \quad \forall K \in \mathcal{T}_h, \tag{71}$$

and then perform the standard finite element assembly to form the system (69). This will save a considerable memory for storing $\mathbf{A}^\ell$, $\mathbf{B}^\ell$, $\mathbf{C}^\ell$, and $\mathbf{D}^\ell$.

It should be noted that the above implementation breaks down in the case of an incompressible material ($\varepsilon = 0$) because the matrix $\mathbf{A}^\ell$ is not invertible in such case. This problem can be treated by using the augmented Lagrangian method

proposed in [45]. In this method, we introduce an artificial time derivative of the pressure increment to the Eq. (66c) and integrate the resulting system until the pressure increment converges. In each iteration of the augmented Lagrangian method, we solve a linear system whose matrix has exactly the same structure as the matrix in (68). A detailed discussion of this method can be found in [45].

4.2.5. Numerical examples

We consider two numerical examples to illustrate the performance of the HDG method. The geometry and boundary conditions for the two test cases are shown in Fig. 15. In both the examples we use a Neo-Hookean material model with the following isochoric and volumetric components

$$\psi_{\text{iso}} = \frac{\mu}{2}(\text{tr}(\boldsymbol{F}^T\boldsymbol{F}) - 3) - \mu \log(J), \quad \psi_{\text{vol}} = \frac{\lambda}{2}(J-1)^2, \tag{72}$$

where μ is the shear modulus and λ is the bulk modulus.

The first example involves a unit thickness square plate which is clamped on the bottom side and subject to shear load on the other sides as shown in Fig. 15(a). The material parameters are $\mu = \lambda = 1$ N/mm^2. The problem has the following exact solution

$$u_1 = x_1 + x_2^3/2 + 0.5\sin(\pi x_2/2), \quad u_2 = x_2, \quad p = 0.$$

The body force $\boldsymbol{f}$ and shear load $\boldsymbol{g}_N$ are determined from the above solution. We consider triangular meshes that are obtained by splitting a regular $n \times n$ Cartesian grid into a total of $2n^2$ triangles, giving uniform element sizes of $h = 2/n$. On these meshes, we use polynomials of degree k to represent all the approximate variables using a nodal basis within each element for the HDG method. We take the stabilization tensor to be the identity tensor for this problem. We present in Table 8 the error and order of convergence in L^2-norm. We observe that the approximate displacement and pressure converges with order $k+1$, so does the approximate stress tensor $\boldsymbol{P}_h$. It is interesting to see that the approximate displacement gradient $\boldsymbol{F}_h$ converges with order k for $k = 1, 2$, but with the optimal order $k+1$ for $k = 3, 4$. As a result, the postprocessed displacement $\boldsymbol{u}_h^*$ converges with order $k+1$ for $k = 1, 2$, but with order $k+2$ for $k = 3, 4$. Fig. 16 shows both the original solution and postprocessed solution for $k = 2$ and $h = 1/2$. We see that the local postprocessing enhances the approximation quality as the postprocessed solution looks better than the original solution.

The second example is the well-known cook's membrane problem studied in [59]. The geometry, loading and boundary conditions are given in Fig. 15(b). The values for the material parameters are $\mu = 80.194$ N/mm^2, $\lambda = 120.291$ N/mm^2 for the compressible case and $\mu = 80.194$ N/mm^2, $\lambda = 400889.806$ N/mm^2 for the nearly incompressible case. Note that we apply $\boldsymbol{g}_N = (0, 250/16)$ N/mm^2 for the compressible case and $\boldsymbol{g}_N = (0, 125/16)$ N/mm^2 for the incompressible case. The stabilization tensor is chosen as $\boldsymbol{S} = 4\,\mu\boldsymbol{I}$. The quantity of interest is the vertical displacement of the top right corner of the plate. The convergence as a function of the number of elements and polynomial degrees are used as criteria to investigate the performance of the HDG method. We present in Tables 9 and 10 the computed results for the compressible case and nearly incompressible case, respectively. In both cases, we observe that using $k = 4$ on the coarsest mesh 2×2 yields very similar results as $k = 1$ on the finest mesh 30×16. Fig. 17 depicts the deformed configuration and the Von Mises stress for the compressible case

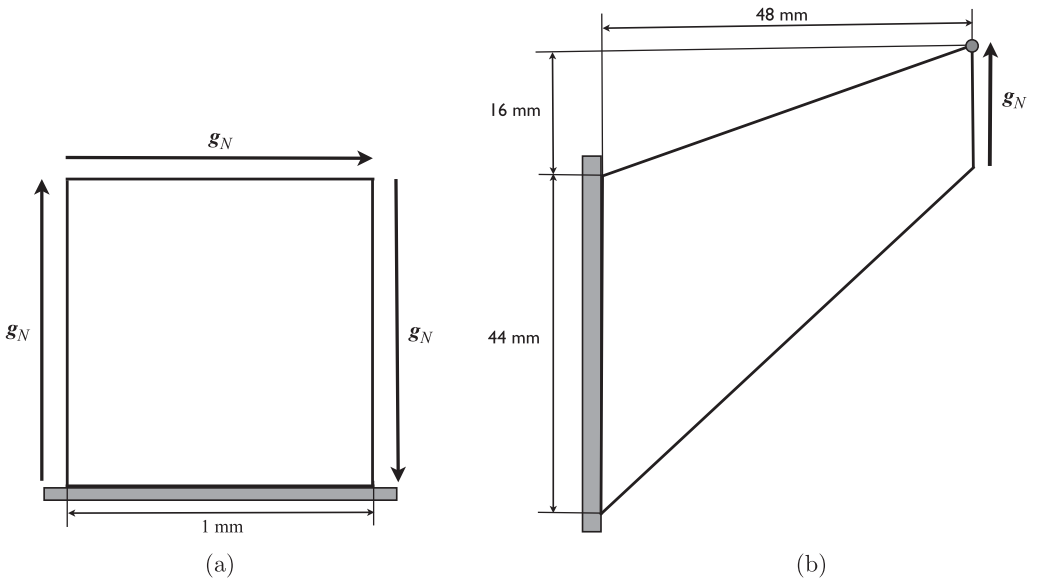

Fig. 15. Geometry and boundary conditions for the shearing plate example (a) and cook's membrane example (b).

Table 8
History of convergence of the HDG method for the shearing plate example.

Degree	Mesh	$\|\boldsymbol{u}-\boldsymbol{u}_h\|_{\mathcal{T}_h}$		$\|p-p_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{F}-\boldsymbol{F}_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{P}-\boldsymbol{P}_h\|_{\mathcal{T}_h}$		$\|\boldsymbol{u}-\boldsymbol{u}_h^*\|_{\mathcal{T}_h}$	
k	n	Error	Order	Error	Order	Error	Order	Error	Order	Error	Order
1	2	7.53e−3	–	5.12e−3	–	3.99e−2	–	3.53e−2	–	4.27e−3	–
	4	1.84e−3	2.04	1.32e−3	1.95	1.62e−2	1.30	1.05e−2	1.76	7.78e−4	2.46
	8	4.51e−4	2.03	3.25e−4	2.02	7.12e−3	1.18	2.90e−3	1.85	1.54e−4	2.34
	16	1.12e−4	2.01	8.01e−5	2.02	3.39e−3	1.07	7.66e−4	1.92	3.48e−5	2.14
	32	2.80e−5	2.00	1.96e−5	2.03	1.68e−3	1.02	1.97e−4	1.96	8.49e−6	2.04
2	2	6.16e−4	–	4.91e−4	–	3.75e−3	–	3.18e−3	–	2.49e−4	–
	4	7.48e−5	3.04	6.21e−5	2.98	6.20e−4	2.60	4.55e−4	2.81	1.85e−5	3.75
	8	9.25e−6	3.02	7.62e−6	3.03	1.11e−4	2.48	6.09e−5	2.90	1.53e−6	3.60
	16	1.15e−6	3.01	9.13e−7	3.06	2.33e−5	2.25	7.85e−6	2.96	1.52e−7	3.33
	32	1.44e−7	3.00	1.09e−7	3.07	5.48e−6	2.09	9.93e−7	2.98	1.74e−8	3.13
3	2	2.21e−5	–	2.27e−5	–	1.70e−4	–	1.38e−4	–	7.08e−6	–
	4	1.35e−6	4.03	1.52e−6	3.90	1.33e−5	3.67	9.55e−6	3.85	2.53e−7	4.81
	8	8.39e−8	4.01	1.02e−7	3.90	1.02e−6	3.70	6.44e−7	3.89	8.96e−9	4.82
	16	5.22e−9	4.01	6.88e−9	3.89	7.61e−8	3.75	4.28e−8	3.91	3.12e−10	4.84
	32	3.26e−10	4.00	4.57e−10	3.91	5.39e−9	3.82	2.79e−9	3.94	1.06e−11	4.88
4	2	8.68e−7	–	1.15e−6	–	8.41e−6	–	6.04e−6	–	2.60e−7	–
	4	2.65e−8	5.03	3.74e−8	4.95	2.99e−7	4.81	2.06e−7	4.88	4.41e−9	5.88
	8	8.21e−10	5.01	1.20e−9	4.96	1.03e−8	4.86	6.81e−9	4.92	7.33e−11	5.91
	16	2.56e−11	5.00	3.86e−11	4.96	3.42e−10	4.91	2.22e−10	4.94	1.20e−12	5.94

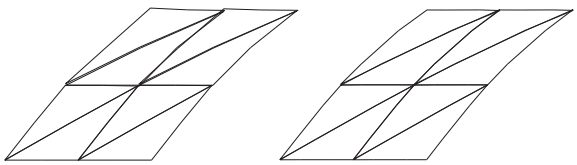

Fig. 16. The deformed configuration of the shearing plate: original solution (left) and the postprocessed solution (right) for $k = 2$ and $h = 1/2$.

Table 9
Vertical displacement of top right conner as a function of h and k for the compressible case of the Cook's membrane problem.

Mesh	Original solution				Postprocessed solution			
	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 1$	$k = 2$	$k = 3$	$k = 4$
2×2	8.5772	15.3386	15.8822	15.9896	8.5974	15.3186	15.8859	15.9903
6×4	13.8347	15.9610	16.0123	16.0204	13.8312	15.9593	16.0115	16.0196
14×8	15.6074	16.0171	16.0237	16.0331	15.6042	16.0155	16.0227	16.0323
30×16	15.9592	16.0285	16.0424	16.0446	15.9565	16.0271	16.0414	16.0438

obtained using $k = 1$, 2 and 4 on different meshes. Fig. 18 shows the same quantities for the nearly incompressible case. We observe that the results for $k = 4$ are just as good as those for $k = 1$ and $k = 2$ although the mesh for $k = 4$ is significantly coarser than the other meshes for $k = 1$ and $k = 2$. These results clearly demonstrate the benefit of using high-order methods for nonlinear elasticity.

4.3. Nonlinear elastodynamics

In this subsection, we extend the HDG method developed above to nonlinear elastodynamics. (For linear elastodynamics we prefer to our prior work [44].) We consider the backward difference formula (BDF) schemes and diagonally implicit Runge–Kutta (DIRK) methods for temporal discretization. The resulting methods are fully implicit, unstructured, and high-order accurate in both space and time. They are also efficient because they reduce the globally coupled unknowns to the numerical trace of the velocity field. Moreover, we can apply the local processing described above to obtain an improved velocity and displacement field. These attractive features render them ideally suited for solving many problems in structural dynamics.

4.3.1. Governing equations

We consider the motion of an elastic body described by the deformation mapping $\boldsymbol{\varphi}$. We denote the velocity by $\boldsymbol{v} = \partial\,\boldsymbol{\varphi}/\partial t$ and the density of the reference configuration by ρ. The other quantities are already introduced in Section 4.2. In an Lagrangian framework, the motion of the elastic solid is governed by the equations

Table 10
Vertical displacement of top right conner as a function of h and k for the nearly incompressible case of the Cook's membrane problem.

Mesh	Original solution				Postprocessed solution			
	$k=1$	$k=2$	$k=3$	$k=4$	$k=1$	$k=2$	$k=3$	$k=4$
2×2	3.4209	7.2720	8.0767	8.2364	3.4299	7.2551	8.0782	8.2398
6×4	6.0013	8.2077	8.3097	8.3415	6.0068	8.2094	8.3107	8.3417
14×8	7.7917	8.3280	8.3521	8.3613	7.7951	8.3286	8.3522	8.3612
30×16	8.2493	8.3569	8.3657	8.3681	8.2513	8.3569	8.3655	8.3676

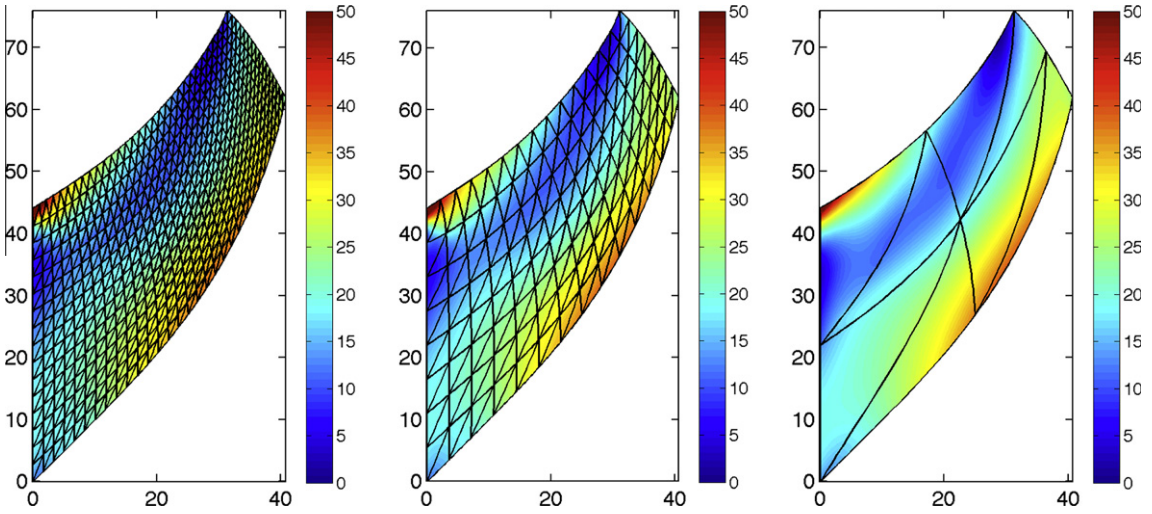

Fig. 17. Deformed configuration and the Von Mises stress field for $k=1$ (left), $k=2$ (middle), and $k=4$ (right) for the compressible case. Also shown are the computational meshes.

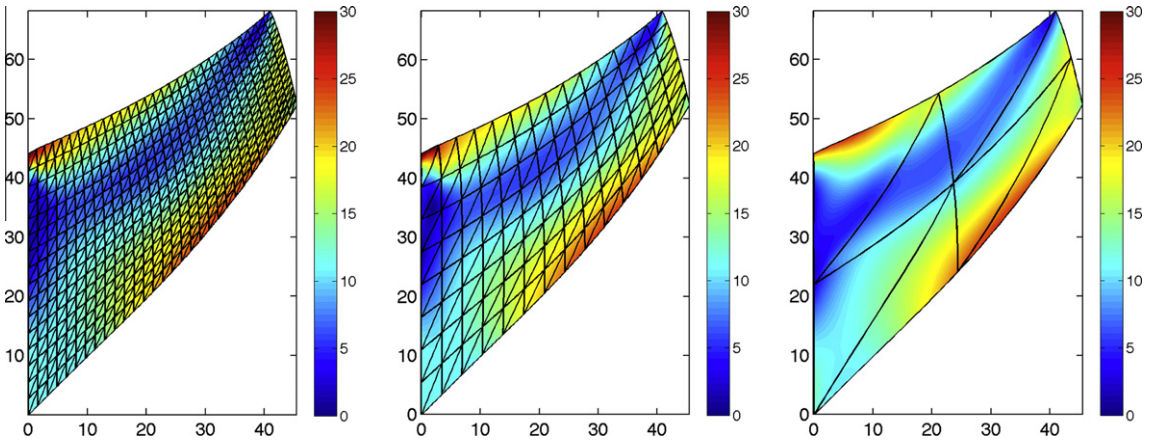

Fig. 18. Deformed configuration and the Von Mises stress field for $k=1$ (left), $k=2$ (middle), and $k=4$ (right) for the nearly incompressible case. Also shown are the computational meshes.

$$
\begin{aligned}
&\frac{\partial \boldsymbol{F}}{\partial t} - \nabla \boldsymbol{v} = 0, \quad \text{in } \Omega \times (0,T],\\
&\rho \frac{\partial \boldsymbol{v}}{\partial t} - \nabla \cdot \boldsymbol{P} = \boldsymbol{f}, \quad \text{in } \Omega \times (0,T],\\
&\varepsilon p - U(J) = 0, \quad \text{in } \Omega \times (0,T],\\
&\boldsymbol{v} = \boldsymbol{g}_D, \quad \text{on } \partial\Omega_D \times (0,T],\\
&\boldsymbol{P}\boldsymbol{n} = \boldsymbol{g}_N, \quad \text{on } \partial\Omega_N \times (0,T],
\end{aligned}
\tag{73}
$$

with initial conditions $\boldsymbol{v}(\boldsymbol{x},t=0)=\boldsymbol{v}_0(\boldsymbol{x})$ and $\boldsymbol{F}(\boldsymbol{x},t=0)=\boldsymbol{F}_0(\boldsymbol{x})$ for all $\boldsymbol{x}\in\Omega$. The spatial and temporal discretization of (73) is described in detail as follows.

4.3.2. Time integration using backward difference formulas

We directly consider the backward-Euler (BE) method for temporal discretization as higher-order time-stepping methods admit a similar procedure. We use the same notation introduced in Section 3.5 for time integration. The HDG-BE method seeks an approximation $(\boldsymbol{F}_h^n,\boldsymbol{v}_h^n,p_h^n,\hat{\boldsymbol{v}}_h^n)\in\mathcal{Q}_h^k\times\mathcal{V}_h^k\times\mathcal{W}_h^k\times\mathcal{M}_h^k$ at time $t^n=n\Delta t^n$ such that

$$\left(\frac{\boldsymbol{F}_h^n-\boldsymbol{F}_h^{n-1}}{\Delta t^n},\boldsymbol{E}\right)_{\mathcal{T}_h}+\left(\boldsymbol{v}_h^n,\nabla\cdot\boldsymbol{E}\right)_{\mathcal{T}_h}-\left\langle\hat{\boldsymbol{v}}_h^n,\boldsymbol{E}\cdot\boldsymbol{n}\right\rangle_{\partial\mathcal{T}_h}=0, \tag{74a}$$

$$\left(\rho\frac{\boldsymbol{v}_h^n-\boldsymbol{v}_h^{n-1}}{\Delta t^n},\boldsymbol{w}\right)_{\mathcal{T}_h}+\left(\boldsymbol{P}(\boldsymbol{F}_h^n,p_h^n),\nabla\boldsymbol{w}\right)_{\mathcal{T}_h}-\left\langle\hat{\boldsymbol{h}}_h^n,\boldsymbol{w}\right\rangle_{\partial\mathcal{T}_h}-\left(\boldsymbol{f}^{\,n},\boldsymbol{v}\right)_{\mathcal{T}_h}=0, \tag{74b}$$

$$\left(\varepsilon p_h^n,q\right)_{\mathcal{T}_h}-\left(U(\boldsymbol{F}_h^n),q\right)_{\mathcal{T}_h}=0, \tag{74c}$$

$$\left\langle\hat{\boldsymbol{h}}_h^n,\boldsymbol{\mu}\right\rangle_{\partial\mathcal{T}_h\setminus\partial\Omega}+\left\langle\hat{\boldsymbol{v}}_h^n-\boldsymbol{g}_D,\boldsymbol{\mu}\right\rangle_{\partial\Omega_D}+\left\langle\hat{\boldsymbol{h}}_h^n-\boldsymbol{g}_N,\boldsymbol{\mu}\right\rangle_{\partial\Omega_N}=0, \tag{74d}$$

for all $(\boldsymbol{E},\boldsymbol{w},q,\boldsymbol{\mu})\in\mathcal{Q}_h^k\times\mathcal{V}_h^k\times\mathcal{W}_h^k\times\mathcal{M}_h^k$, where the numerical flux $\hat{\boldsymbol{h}}_h^n$ is given by

$$\hat{\boldsymbol{h}}_h^n=\boldsymbol{P}(\boldsymbol{F}_h^n,p_h^n)\cdot\boldsymbol{n}-\boldsymbol{S}^n(\boldsymbol{v}_h^n-\hat{\boldsymbol{v}}_h^n). \tag{74e}$$

Here the stabilization matrix $\boldsymbol{S}^n$ is chosen as in (65), and the numerical approximations $\left(\boldsymbol{v}_h^0,\boldsymbol{F}_h^0\right)$ at time $t=0$ are computed as the L^2 projection of the initial conditions.

Since the discrete system (74) is similar to the system (64) for the static case, it can be solved by following the solution procedure described in Section 4.2.4. Once $\boldsymbol{v}_h^n$ is available we can compute $\boldsymbol{\varphi}_h^n\in\mathcal{V}_h^k$ by solving the following equation

$$\frac{1}{\Delta t^n}\left(\boldsymbol{\varphi}_h^n,\boldsymbol{w}\right)_{\mathcal{T}_h}=\left(\boldsymbol{v}_h^n,\boldsymbol{w}\right)_{\mathcal{T}_h}+\frac{1}{\Delta t^n}\left(\boldsymbol{\varphi}_h^{n-1},\boldsymbol{w}\right)_{\mathcal{T}_h},\quad\forall\boldsymbol{w}\in\mathcal{V}_h^k. \tag{75}$$

This equation can be solved at the element level thanks to the discontinuous nature of the space $\mathcal{V}_h^k$.

The application of higher-order BDF schemes leads to discrete systems similar to (74). The resulting method is thus high-order accurate in both space and time. Temporal discretization can also be done by using DIRK methods as discussed next.

4.3.3. Time integration using diagonally implicit Runge–Kutta methods

To begin we denote by $\boldsymbol{Y}_h^n\equiv(\boldsymbol{F}_h^n,\boldsymbol{v}_h^n)$, $t^{n,i}\equiv t^n+c_i\Delta t^n$ and $\boldsymbol{Y}_h^{n,i}\equiv\left(\boldsymbol{F}_h^{n,i},\boldsymbol{v}_h^{n,i}\right)$, $1\leqslant i\leqslant q$, where $\left(\boldsymbol{F}_h^{n,i},\boldsymbol{v}_h^{n,i}\right)=(\boldsymbol{F}_h(t^{n,i}),\boldsymbol{v}_h(t^{n,i}))$ are the intermediate solutions. The numerical solution $\boldsymbol{Y}_h^{n+1}\equiv\left(\boldsymbol{F}_h^{n+1},\boldsymbol{v}_h^{n+1}\right)$ at time level $n+1$ given by the DIRK (q,p) method is computed as follows:

$$\boldsymbol{Y}_h^{n+1}=\boldsymbol{Y}_h^n+\Delta t^n\sum_{i=1}^{q}b_i\boldsymbol{Z}_h^{n,i} \tag{76}$$

where

$$\begin{aligned}
\boldsymbol{Z}_h^{n,1}&=\frac{\boldsymbol{Y}_h^{n,1}-\boldsymbol{Y}_h^n}{a_{11}\Delta t^n},\\
\boldsymbol{Z}_h^{n,2}&=\frac{\boldsymbol{Y}_h^{n,2}-\boldsymbol{Y}_h^n}{a_{22}\Delta t^n}-\frac{a_{21}}{a_{22}}\boldsymbol{Z}_h^{n,1},\\
&\cdots\\
\boldsymbol{Z}_h^{n,q}&=\frac{\boldsymbol{Y}_h^{n,q}-\boldsymbol{Y}_h^n}{a_{qq}\Delta t^n}-\sum_{j=1}^{q-1}\frac{a_{qj}}{a_{qq}}\boldsymbol{Z}_h^{n,j}.
\end{aligned} \tag{77}$$

The intermediate states $\boldsymbol{Y}_h^{n,i}=\left(\boldsymbol{F}_h^{n,i},\boldsymbol{v}_h^{n,i}\right)$, $1\leqslant i\leqslant q$, are determined as follows: $\left(\boldsymbol{F}_h^{n,i},\boldsymbol{v}_h^{n,i},p_h^{n,i},\hat{\boldsymbol{v}}_h^{n,i}\right)\in\mathcal{Q}_h^k\times\mathcal{V}_h^k\times\mathcal{W}_h^k\times\mathcal{M}_h^k$, $1\leqslant i\leqslant q$, satisfies

$$\left(\frac{\boldsymbol{F}_h^{n,i}}{a_{ii}\Delta t^n},\boldsymbol{E}\right)_{\mathcal{T}_h}+\left(\boldsymbol{v}_h^{n,i},\nabla\cdot\boldsymbol{E}\right)_{\mathcal{T}_h}-\left\langle\hat{\boldsymbol{v}}_h^{n,i},\boldsymbol{E}\cdot\boldsymbol{n}\right\rangle_{\partial\mathcal{T}_h}-\left(\boldsymbol{D}_h^{n,i},\boldsymbol{E}\right)_{\mathcal{T}_h}=0, \tag{78a}$$

$$\left(\rho\frac{\boldsymbol{v}_h^{n,i}}{a_{ii}\Delta t^n},\boldsymbol{w}\right)_{\mathcal{T}_h}+\left(\boldsymbol{P}\left(\boldsymbol{F}_h^{n,i},p_h^{n,i}\right),\nabla\boldsymbol{w}\right)_{\mathcal{T}_h}-\left\langle\hat{\boldsymbol{h}}_h^{n,i},\boldsymbol{w}\right\rangle_{\partial\mathcal{T}_h}-\left(\rho\boldsymbol{s}_h^{n,i}+\boldsymbol{f}^{\,n,i},\boldsymbol{v}\right)_{\mathcal{T}_h}=0, \tag{78b}$$

$$\left(\varepsilon p_h^{n,i},r\right)_{\mathcal{T}_h}-\left(U\left(\boldsymbol{F}_h^{n,i}\right),r\right)_{\mathcal{T}_h}=0, \tag{78c}$$

$$\left\langle\hat{\boldsymbol{h}}_h^{n,i},\boldsymbol{\mu}\right\rangle_{\partial\mathcal{T}_h\setminus\partial\Omega}+\left\langle\hat{\boldsymbol{v}}_h^{n,i}-\boldsymbol{g}_D,\boldsymbol{\mu}\right\rangle_{\partial\Omega_D}+\left\langle\hat{\boldsymbol{h}}_h^{n,i}-\boldsymbol{g}_N,\boldsymbol{\mu}\right\rangle_{\partial\Omega_N}=0, \tag{78d}$$

for all $(\boldsymbol{E},\boldsymbol{w},r,\boldsymbol{\mu}) \in \mathcal{Q}_h^k \times \mathcal{V}_h^k \times \mathcal{W}_h^k \times \boldsymbol{\mathcal{M}}_h^k$, where the terms $\left(\boldsymbol{D}_h^{n,i}, \boldsymbol{s}_h^{n,i}\right), 1 \leqslant i \leqslant q$, in (78a) and (78b) are given by

$$\begin{aligned}
\left(\boldsymbol{D}_h^{n,1}, \boldsymbol{s}_h^{n,1}\right) &= \frac{\boldsymbol{Y}_h^n}{a_{11}\Delta t^n},\\
\left(\boldsymbol{D}_h^{n,2}, \boldsymbol{s}_h^{n,2}\right) &= \frac{\boldsymbol{Y}_h^n}{a_{22}\Delta t^n} + \frac{a_{21}}{a_{22}}\left(\frac{\boldsymbol{Y}_h^{n,1}}{a_{11}\Delta t^n} - \left(\boldsymbol{D}_h^{n,1}, \boldsymbol{s}_h^{n,1}\right)\right),\\
&\cdots\\
\left(\boldsymbol{D}_h^{n,q}, \boldsymbol{s}_h^{n,q}\right) &= \frac{\boldsymbol{Y}_h^n}{a_{qq}\Delta t^n} + \sum_{j=1}^{q-1}\frac{a_{qj}}{a_{qq}}\left(\frac{\boldsymbol{Y}_h^{n,j}}{a_{jj}\Delta t^n} - \left(\boldsymbol{D}_h^{n,j}, \boldsymbol{s}_h^{n,j}\right)\right).
\end{aligned} \tag{79}$$

Once $\left(\boldsymbol{F}_h^{n+1}, \boldsymbol{v}_h^{n+1}\right)$ has been determined as above, we can compute $(p_h^{n+1}, \hat{\boldsymbol{v}}_h^{n+1}) \in \mathcal{W}_h^k \times \boldsymbol{\mathcal{M}}_h^k$ such that

$$\left(\varepsilon p_h^{n+1}, r\right)_{\mathcal{T}_h} - \left(U\left(\boldsymbol{F}_h^{n+1}\right), r\right)_{\mathcal{T}_h} = 0, \tag{80a}$$

$$\left\langle \hat{\boldsymbol{h}}_h^{n+1}, \boldsymbol{\mu}\right\rangle_{\partial\mathcal{T}_h\setminus\partial\Omega} + \left\langle \hat{\boldsymbol{v}}_h^{n+1} - \boldsymbol{g}_D, \boldsymbol{\mu}\right\rangle_{\partial\Omega_D} + \left\langle \hat{\boldsymbol{h}}_h^{n+1} - \boldsymbol{g}_N, \boldsymbol{\mu}\right\rangle_{\partial\Omega_N} = 0, \tag{80b}$$

for all $(r,\boldsymbol{\mu}) \in \mathcal{W}_h^k \times \boldsymbol{\mathcal{M}}_h^k$. Finally, we recover the approximate deformation $\boldsymbol{\varphi}_h^{n+1}$ by applying the DIRK (q,p) method to solve $\partial \boldsymbol{\varphi}_h/\partial t = \boldsymbol{v}_h$ at the same timestep.

We note that the system (78) at each ith stage of the DIRK (q,p) method is very similar to the system (74) of the Backward-Euler method. We further note that the Eq. (80a) can be solved in an element-by-element fashion for p_h^{n+1} and that the Eq. (80b) can then be solved at the element level for $\hat{\boldsymbol{v}}_h^{n+1}$. This is different from the HDG-DIRK (p,q) method for the time-dependent Navier–Stokes equations.

4.3.4. Local postprocessing

The local postprocessing for the time-dependent case is a straightforward extension of the procedure introduced above for the static case. In particular, the new approximate deformation $\boldsymbol{\varphi}_h^{n*} \in (\mathcal{P}^{k+1}(K))^d$ satisfies, on every simplex $K \in \mathcal{T}_h$,

$$\begin{aligned}
\left(\nabla\boldsymbol{\varphi}_h^{n*}, \nabla\boldsymbol{w}\right)_K &= \left(\boldsymbol{F}_h^n, \nabla\boldsymbol{w}\right)_K, \quad \forall \boldsymbol{w} \in (\mathcal{P}^{k+1}(K))^d,\\
\left(\boldsymbol{\varphi}_h^{n*}, 1\right)_K &= \left(\boldsymbol{\varphi}_h^n, 1\right)_K.
\end{aligned} \tag{81}$$

To do the postprocessing for the new approximate velocity, we first need to compute the approximate velocity gradient $\boldsymbol{L}_h^n \in (\mathcal{P}^k(K))^{d\times d}$ such that

$$\left(\boldsymbol{L}_h^n, \boldsymbol{E}\right)_K = -\left(\boldsymbol{v}_h^n, \nabla\cdot\boldsymbol{E}\right)_K + \left\langle \hat{\boldsymbol{v}}_h^n, \boldsymbol{E}\cdot\boldsymbol{n}\right\rangle_{\partial K}, \quad \forall \boldsymbol{E} \in (\mathcal{P}^k(K))^{d\times d}. \tag{82}$$

We then define the new approximate velocity $\boldsymbol{v}_h^{n*} \in (\mathcal{P}^{k+1}(K))^d$ to satisfy, on every simplex $K \in \mathcal{T}_h$,

$$\begin{aligned}
\left(\nabla\boldsymbol{v}_h^{n*}, \nabla\boldsymbol{w}\right)_K &= \left(\boldsymbol{L}_h^n, \nabla\boldsymbol{w}\right)_K, \quad \forall \boldsymbol{w} \in (\mathcal{P}^{k+1}(K))^d,\\
\left(\boldsymbol{v}_h^{n*}, 1\right)_K &= \left(\boldsymbol{v}_h^n, 1\right)_K.
\end{aligned} \tag{83}$$

Note that each component of $\boldsymbol{\varphi}_h^{n,*}$, $\boldsymbol{L}_h^n$ and $\boldsymbol{v}_h^{n*}$ can be solved independently.

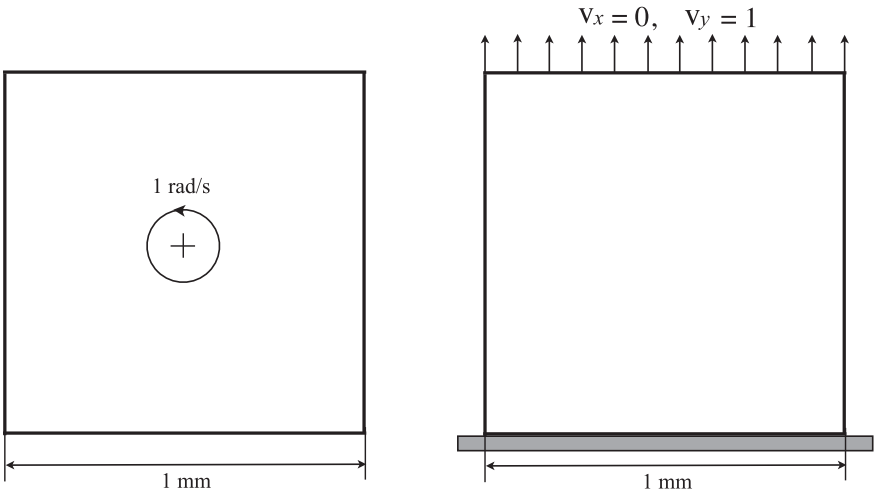

Fig. 19. Geometry and boundary conditions for the spinning plate example (a) and tensile plate example (b).

Since the local postprocessing can be carried out at any particular timestep and performed at the element level, the post-processsed deformation and velocity are less inexpensive to compute than the original approximations. Therefore, the local postprocessing adds very little to the overall computational cost. Note however that our local postprocessing is effective only when the temporal accuracy is of order $k+2$.

4.3.5. Numerical examples

We consider two numerical examples studied in [36] to illustrate the performance of the HDG method. The geometry and boundary conditions for the two test cases are shown in Fig. 19. The Neo-Hookean material model (72) is used in both the examples.

In the first example, a unit thickness square plate without any constraints is released without any initial deformation and an initial angular velocity of 1 rad/s. The statement of the problem can be seen in Fig. 19(a). The density of the plate and edge length are chosen to be unity. The material parameters are $\mu = 3.4483$ N/mm^2 and $\lambda = 33.3333$ N/mm^2. Note that the initial conditions used here are such that there is no steady state solution, even in a rotating reference frame. This problem is

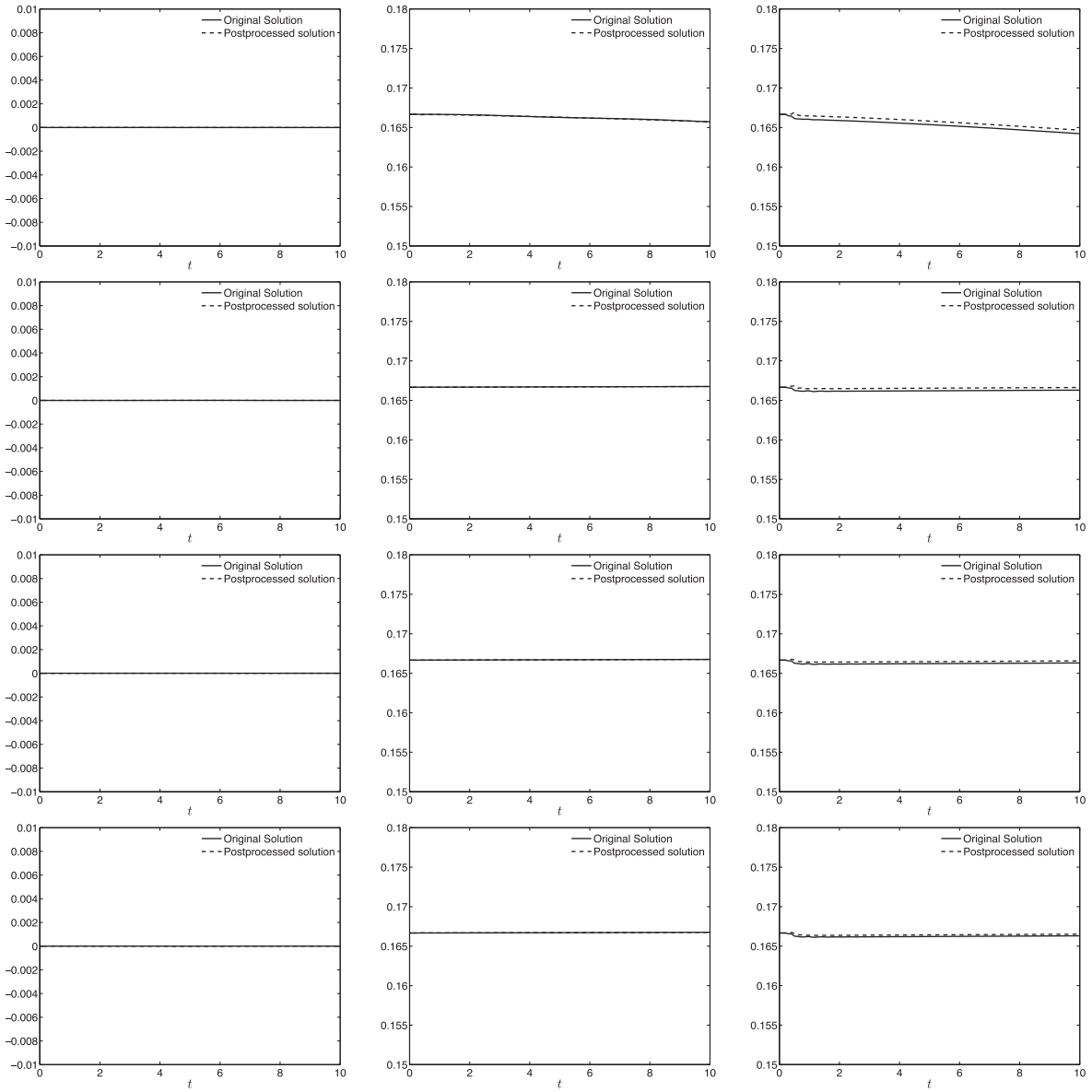

Fig. 20. Numerical results for the spinning plate problem for $k = 1$ (first row), $k = 2$ (second row), $k = 3$ (third row), and $k = 4$ (last row) on a coarse grid 4×4: the total linear momentum, angular momentum, and energy are shown in the first, second, and third column, respectively.

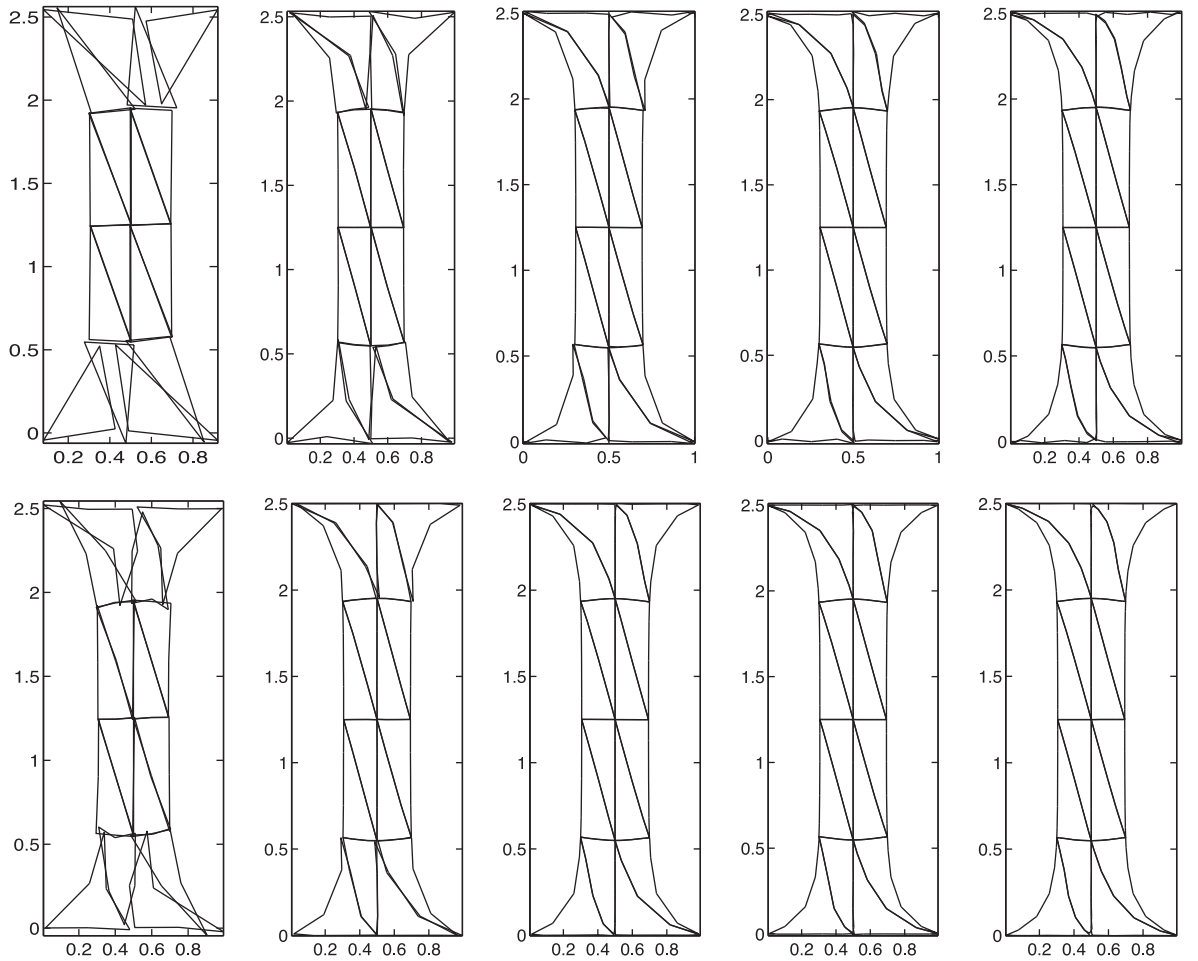

Fig. 21. The deformed configuration of the square plate pulled rapidly at the top surface at time $t = 1.5$ s for the original solution (top row) and the postprocessed solution (bottom row): the results for $k = 1$, $k = 2$, $k = 3$, $k = 4$, and $k = 5$, are shown in the first, second, third, fourth, and fifth column, respectively.

chosen to illustrate the conservation properties of the HDG method. The quantities of interest are the total linear momentum L, total angular momentum J, and total energy E, which are defined as

$$L = \int_\Omega \rho \boldsymbol{v} \cdot \boldsymbol{e}\, d\boldsymbol{x}, \quad J = \int_\Omega \rho(\boldsymbol{\varphi} \times \boldsymbol{v}) \cdot \boldsymbol{e}\, d\boldsymbol{x}, \quad E = \int_\Omega (\rho \boldsymbol{v} \cdot \boldsymbol{v} + \psi)\, d\boldsymbol{x}, \tag{84}$$

where $\boldsymbol{e}$ denotes a vector whose components are equal to 1. We choose $\boldsymbol{S} = \mu \boldsymbol{I}$. We present in Fig. 20 the computed results using a grid 4×4 with $h = 1/4$ for spatial discretization and the DIRK (3,4) scheme with a timestep size $\Delta t = 0.1$ for temporal discretization. We observe that the linear momentum is exactly zero for all values of k and that the angular momentum is conserved very well. Indeed, the difference in the angular momentum at $t = 0$ and $t = 10$ is 9.43×10^{-4} for $k = 1$, 8.84×10^{-5} for $k = 2$, 8.82×10^{-5} for $k = 3$, and 8.82×10^{-5} for $k = 4$. Moreover, the energy is conserved quite well. These results show the excellent conservation properties of the HDG method.

In the second example, a unit thickness square plate is clamped at the bottom size and pulled rapidly by a constant velocity $\boldsymbol{v}_D = (0,1)$ mm/s at the top surface. The statement of the problem can be seen in Fig. 19(b). The density of the plate and edge length are chosen to be unity. The material parameters are $\mu = 344.83$ N/mm^2 and $\lambda = 3333.33$ N/mm^2. This problem is chosen to illustrate the effectiveness of the local posprocessing in improving the accuracy. We set $\boldsymbol{S} = 5\,\mu \boldsymbol{I}$ and use the DIRK (3,4) scheme with a timestep size $\Delta t = 0.05$ for temporal discretization. We show in Fig. 21 the deformed configuration of the square plate at time $t = 1.5$ s for both the original solution and the postpocessed solution. It is clearly seen that the local postprocessing enhances the accuracy quite significantly since the postprocessed solution look much better than the original solution for all k. Note that the enhancement of the accuracy requires very little additional cost since the local postprocessing is performed at the element level and only at the timestep where an enhanced accuracy is desired. The results presented here show the excellent stability and accuracy of the HDG method for nonlinear elastodynamics problems.

5. Conclusions

We have presented hybridizable discontinuous Galerkin methods for solving steady and time-dependent PDEs in continuum mechanics, with the objective of introducing a unified discretization framework for computational fluid dynamics and computational solid mechanics. The essential ingredients are a local Galerkin projection of the underlying PDEs at the element level onto spaces of polynomials of degree k to parametrize the numerical solution in terms of the numerical trace; a judicious choice of the numerical flux to provide stability and consistency; a global jump condition that enforces the continuity of the numerical flux to arrive at a global weak formulation in terms of the numerical trace; an appropriate definition of the numerical flux on the boundary to weakly impose boundary conditions; and a local postprocessing to improve the accurate order of convergence. We have shown how to use these ingredients to devise HDG methods for a wide variety of PDEs in continuum mechanics: incompressible Stokes and Navier–Stokes equations, compressible Euler and Navier–Stokes equations, linear and nonlinear elasticity, and dynamic nonlinear elasticity. The power of the proposed framework lies in its simplicity and generality, so that it can be applied to develop HDG methods for other systems of PDEs. Indeed, HDG methods for electromagnetics have already been introduced in [43].

The proposed HDG methods inherit all the advantages of DG methods and possess several unique features which distinguish themselves from other DG methods. The main features include a reduction of globally coupled unknowns to the approximate trace of the field variables, an optimal convergence of order $k+1$ for all the approximate variables for smooth diffusion-dominated problems, a higher convergence rate of order $k+2$ for the postprocessed variables, and an effective treatment of incompatible boundary conditions. In addition, they possess other interesting properties for specific applications. Their numerical solution can be postprocessed to yield an exactly divergence-free and $\boldsymbol{H}(\mathrm{div})$-conforming velocity field for incompressible flows. They do not exhibit volumetric locking behavior for nearly incompressible solids. They have excellent conservation, stability, and accuracy properties for nonlinear elastodynamics. The HDG methods and their distinct characteristics have been demonstrated through several numerical examples and comparison with continuous Galerkin methods for some examples.

Several extensions of the proposed framework are possible. Other projection methods such as the Petrov–Galerkin method and the collocation method may be used for both the local problem and the global weak formulation. These projection methods may help to increase the stability of the resulting scheme. This research direction is investigated in [42] by adapting some ideas from the discontinuous Petrov–Galerkin method introduced in [20]. In the HDG framework, we have not explored other possible choices for the numerical flux and the stabilization matrix, which may potentially yield better results than the ones used in this paper. Another possible extension is the introduction of non-polynomial basis functions to represent the approximate solution. For instance, in the spirit of the extended finite element method [40], we can enrich the space of polynomials with problem-dependent basis functions in some particular elements in order to capture discontinuities, singularities, and boundary layers. The HDG framework may lend itself more natural than the CG method for this task because the enrichment can be done at the element level thanks to the discontinuous nature of the approximation spaces.

We close by making a remark that it is highly desirable to develop an efficient HDG-like method to be used with explicit time discretization for time-dependent problems. Of course, the method should be computationally competitive to other explicit DG methods, while retaining some important advantages of the current HDG method such as the superconvergence property. This is a subject worthy of further investigation.

Acknowledgements

We are grateful to Professor Bernardo Cockburn of UMN for his invaluable contribution to this work. We would like to acknowledge the Singapore-MIT Alliance and the Air Force Office of Scientific Research under the MURI project on Biologically Inspired Flight for supporting this work.

References

[1] R. Alexander, Diagonally implicit Runge–Kutta methods for stiff ODEs, SIAM J. Numer. Anal. 14 (1977) 1006–1021.
[2] D.A. Anderson, J.C. Tannehill, R.H. Pletcher, Computational Fluid Dynamics and Heat Transfer, Hemisphere Publishing, New York, 1984.
[3] D.N. Arnold, F. Brezzi, B. Cockburn, L.D. Marini, Unified analysis of discontinuous Galerkin methods for elliptic problems, SIAM J. Numer. Anal. 39 (5) (2001) 1749–1779.
[4] D.N. Arnold, F. Brezzi, L.D. Marini, A family of discontinuous Galerkin finite elements for the Reissner–Mindlin plate, J. Sci. Comput. 22/23 (2005) 25–45.
[5] F. Bassi, S. Rebay, A high-order accurate discontinuous finite element method for the numerical solution of the compressible Navier–Stokes equations, J. Comput. Phys. 131 (2) (1997) 267–279.
[6] C. Baumann, J. Oden, A discontinuous hp finite element method for convection–diffusion problems, Comput. Methods Appl. Mech. Eng. 175 (1999) 311–341.
[7] F. Brezzi, J. Douglas Jr., L.D. Marini, Two families of mixed finite elements for second order elliptic problems, Numer. Math. 47 (1985) 217–235.
[8] F. Brezzi, M. Fortin, Mixed and Hybrid Finite Element Methods, Springer Verlag, 1991.
[9] G. Chiocchia. Exact solutions to transonic and supersonic flows. Technical Report AR-211, AGARD, 1985.
[10] B. Cockburn, B. Dong, J. Guzmán, A superconvergent LDG-hybridizable Galerkin method for second-order elliptic problems, Math. Comput. 77 (2008) 1887–1916.
[11] B. Cockburn, B. Dong, J. Guzmán, M. Restelli, R. Sacco, A hybridizable discontinuous galerkin method for steady-state convection–diffusion–reaction problems, SIAM J. Sci. Comput. 31 (5) (2009) 3827–3846.

[12] B. Cockburn, J. Gopalakrishnan, The derivation of hybridizable discontinuous Galerkin methods for Stokes flow, SIAM J. Numer. Anal. 47 (2009) 1092–1125.
[13] 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.
[14] B. Cockburn, J. Gopalakrishnan, N.C. Nguyen, J. Peraire, F.-J. Sayas, Analysis of HDG methods for Stokes flow, Math. Comput. 80 (2011) 723–760.
[15] B. Cockburn, J. Gopalakrishnan, F.-J. Sayas, A projection-based error analysis of HDG methods, Math. Comput. 79 (2010) 1351–1367.
[16] B. Cockburn, J. Guzmán, H. Wang, Superconvergent discontinuous Galerkin methods for second-order elliptic problems, Math. Comput. 78 (2009) 1–24.
[17] B. Cockburn, N.C. Nguyen, J. Peraire, A comparison of HDG methods for Stokes flow, J. Sci. Comput. 45 (1–3) (2010) 215–237.
[18] B. Cockburn, C.W. Shu, The local discontinuous Galerkin method for convection–diffusion systems, SIAM J. Numer. Anal. 35 (1998) 2440–2463.
[19] B. Cockburn, C.-W. Shu, Runge–Kutta discontinuous Galerkin methods for convection-dominated problems, J. Sci. Comput. 16 (3) (2001) 173–261.
[20] L. Demkowicz, J. Gopalakrishnan. A class of discontinuous Petrov–Galerkin methods. Part I: The transport equation, Comput. Methods Appl. Mech. Eng., 199 (23–24) (2010) 1558-1572.
[21] C. Farhat, P. Geuzaine, Design and analysis of robust ALE time-integrators for the solution of unsteady flow problems on moving grids, Comput. Methods Appl. Mech. Eng. 193 (2004) 4073–4095.
[22] C. Farhat, P. Geuzaine, C. Grandmont, The discrete geometric conservation law and the nonlinear stability of ALE schemes for the solution of flow problems on moving grids, J. Comput. Phys. 174 (2001) 669–694.
[23] K.J. Fidkowski, T.A. Oliver, J. Lu, D.L. Darmofal, *p*-Multigrid solution of high-order discontinuous Galerkin discretizations of the compressible Navier–Stokes equations, J. Comput. Phys. 207 (1) (2005) 92–113.
[24] M. Fortin, R. Glowinski, Augmented Lagrangian methods, Studies in Mathematics and its Applications, 15, North-Holland Publishing Co., Amsterdam, 1983. Applications to the numerical solution of boundary value problems, Translated from the French by B. Hunt and D.C. Spicer.
[25] S. Güzey, B. Cockburn, H. Stolarski, The embedded discontinuous Galerkin methods: application to linear shells problems, Int. J. Numer. Methods Eng. 70 (2007) 757–790.
[26] S. Güzey, H.K. Stolarski, B. Cockburn, K.K. Tamma, Design and development of a discontinuous Galerkin method for shells, Comput. Methods Appl. Mech. Eng. 195 (2006) 3528–3548.
[27] P. Hansbo, M.G. Larson, Discontinuous Galerkin methods for incompressible and nearly incompressible elasticity by Nitsche's method, Comput. Methods Appl. Mech. Eng. 191 (2002) 895–1908.
[28] R. Hartmann, P. Houston, Adaptive discontinuous Galerkin finite element methods for the compressible Euler equations, J. Comput. Phys. 183 (2002) 508–532.
[29] J.S. Hesthaven, T. Warburton, Nodal high-order methods on unstructured grids I. Time-domain solution of Maxwell's equations, J. Comput. Phys. 181 (1) (2002) 186–221.
[30] T.J.R. Hughes, G. Scovazzi, P.B. Bochev, A. Buffa, A multiscale discontinuous Galerkin method with the computational structure of a continuous Galerkin method, Comput. Methods Appl. Mech. Eng. 195 (2006) 2761–2787.
[31] T.J.R. Hughes, L.P. Franca, M. Balestra, A new finite element formulation for computational fluid dynamics: V. Circumventing the Babuska–Brezzi condition: a stable Petrov–Galerkin formulation of the Stokes problem accomodating equal order interpolations, Comput. Methods Appl. Mech. Eng. 59 (1986) 85–99.
[32] T.J.R. Hughes, L.P. Franca, G.M. Hulbert, A new finite element formulation for computational fluid dynamics: VIII. The Galerkin least-squares method for advective–diffusive equations, Comput. Methods Appl. Mech. Eng. 73 (1989) 173–189.
[33] C.M. Klaij, J.J.W. van der Vegt, H. van der Ven, Space-time discontinuous Galerkin method for the compressible Navier–Stokes equations, J. Comput. Phys. 217 (2) (2006) 589–611.
[34] L.I.G. Kovasznay, Laminar flow behind two-dimensional grid, Proc. Cambridge Philos. Soc. 44 (1948) 58–62.
[35] R.J. Labeur, G.N. Wells, A Galerkin interface stabilisation method for the advection–diffusion and incompressible Navier–Stokes equations, Comput. Methods Appl. Mech. Eng. 196 (2007) 4985–5000.
[36] S.K. Lahiri, J. Bonet, J. Peraire, A variationally consistent mesh adaptation method for triangular elements in explicit Lagrangian dynamics, Int. J. Numer. Methods Eng. 82 (2010) 1073–1113.
[37] C. Liu, X. Zheng, C.H. Sung, Preconditioned multigrid methods for unsteady incompressible flows, J. Comput. Phys. 139 (1) (1998) 35–57.
[38] I. Lomtev, G.E. Karniadakis, A discontinuous Galerkin method for the Navier–Stokes equations, Int. J. Numer. Methods Fluids 29 (1999) 587–603.
[39] I. Lomtev, R.M. Kirby, G.E. Karniadakis, A discontinuous Galerkin ALE method for compressible viscous flows in moving domains, J. Comput. Phys. 155 (1) (1999) 128–159.
[40] N. Moës, J. Dolbow, T. Belytschko, A finite element method for crack growth without remeshing, Int. J. Numer. Methods Eng. 46 (1999) 131–150.
[41] A. Montlaur, S. Fernández-Méndez, J. Peraire, A. Huerta, Discontinuous Galerkin methods for the Navier–Stokes equations using solenoidal approximations Internat, J. Numer. Methods Fluids 64 (2010) 549–564.
[42] D. Moro, N.C. Nguyen, J. Peraire, J. Gopalakrishnan, A hybridized discontinuous Petrov–Galerkin method for compressible flows, in: Proceedings of the 49th AIAA Aerospace Sciences Meeting and Exhibit, Orlando, Florida, (AIAA Paper 2011-197), Janury 2011.
[43] N.C. Nguyen, J. Peraire, B. Cockburn. Hybridizable discontinuous Galerkin methods for the time-harmonic Maxwell's equations, J. Comput. Phys., 230 (19) (2011) 7151–7175.
[44] N.C. Nguyen, J. Peraire, B. Cockburn, Implicit high-order hybridizable discontinuous Galerkin methods for acoustics and elastodynamics, J. Comput. Phys. 230 (2011) 3695–3718.
[45] N.C. Nguyen, J. Peraire, B. Cockburn, An implicit high-order hybridizable discontinuous Galerkin method the incompressible Navier–Stokes equations, J. Comput. Phys. 230 (2011) 1147–1170.
[46] N.C. Nguyen, J. Peraire, B. Cockburn. Hybridizable discontinuous Galerkin methods, in: J.S. Hesthaven, E.M. Ronquist, (Eds.), Spectral and high order methods for partial differential equations, Lecture Notes in Computational Science and Engineering, 2011, Springer, vol. 76, pp. 63–84.
[47] N.C. Nguyen, J. Peraire, B. Cockburn, An implicit high-order hybridizable discontinuous Galerkin method for linear convection–diffusion equations, J. Comput. Phys. 228 (2009) 3232–3254.
[48] N.C. Nguyen, J. Peraire, B. Cockburn, An implicit high-order hybridizable discontinuous Galerkin method for nonlinear convection–diffusion equations, J. Comput. Phys. 228 (2009) 8841–8855.
[49] N.C. Nguyen, J. Peraire, B. Cockburn, A hybridizable discontinuous Galerkin method for Stokes flow, Comput. Methods Appl. Mech. Eng. 199 (2010) 582–597.
[50] N.C. Nguyen, J. Peraire, B. Cockburn. A hybridizable discontinuous Galerkin method for the incompressible Navier–Stokes equations, in: Proceedings of the 48th AIAA Aerospace Sciences Meeting and Exhibit, Orlando, Florida, (AIAA Paper 2010-362), January 2010.
[51] L. Noels, R. Radovitzky, A general discontinuous Galerkin method for finite hyperelasticity. Formulation and numerical applications, Int. J. Numer. Methods Eng. 68 (2006) 64–97.
[52] A.T. Patera, A spectral element method for fluid dynamics: laminar flow in a channel expansion, J. Comput. Phys. 54 (1984) 468–488.
[53] J. Peraire, N.C. Nguyen, B. Cockburn. An Embedded discontinuous Galerkin method for the compressible Euler and Navier–Stokes equations, in: Proceedings of the 20th AIAA Computational Fluid Dynamics Conference, Honolulu, Hawaii, (AIAA Paper 2011-3228), June 2011.
[54] J. Peraire, P.O. Persson, The compact discontinuous Galerkin (CDG) method for elliptic problems, SIAM J. Sci. Comput. 30 (4) (2008) 1806–1824.

[55] P.O. Persson, J. Peraire, Newton-GMRES preconditioning for discontinuous Galerkin discretizations of the Navier–Stokes equations, SIAM J. Sci. Comput. 30 (6) (2008) 2709–2733.
[56] P.-O. Persson, J. Bonet, J. Peraire, Discontinuous Galerkin solution of the Navier–Stokes equations on deformable domains, Comput. Methods Appl. Mech. Eng. 198 (2009) 1585–1595.
[57] P.A. Raviart, J.M. Thomas, A mixed finite element method for second order elliptic problems, in: I. Galligani, E. Magenes (Eds.), Mathematical Aspects of Finite Element Method, Lecture Notes in Mathematics, 606, Springer-Verlag, New York, 1977, pp. 292–315.
[58] W.H. Reed, T.R. Hill. Triangular mesh methods for the neutron transport equation, Technical Report LA-UR-73-479, Los Alamos Scientific Laboratory, 1973.
[59] S. Reese, M. Küssner, B.D. Reddy, A new stabilization technique for finite elements in non-linear elasticity, Int. J. Numer. Methods Eng. 44 (1999) 1617–1652.
[60] P.L. Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, J. Comput. Phys. 43 (1981) 357–372.
[61] K. Shahbazi, P.F. Fischer, C.R. Ethier, A high-order discontinuous Galerkin method for the unsteady incompressible Navier–Stokes equations, J. Comput. Phys. 222 (2007) 391–407.
[62] S.-C. Soon, B. Cockburn, H.K. Stolarski, A hybridizable discontinuous Galerkin method for linear elasticity, Int. J. Numer. Methods Eng. 80 (8) (2009) 1058–1092.
[63] R. Stenberg, Some new families of finite elements for the Stokes equations, Numer. Math. 56 (1990) 827–838.
[64] A. Ten Eyck, A. Lew, Discontinuous Galerkin methods for non-linear elasticity, Int. J. Numer. Methods Eng. 67 (2006) 1204–1243.
[65] M.P. Ueckermann, P.F.J. Lermusiaux, High order schemes for 2D unsteady biogeochemical ocean models, Ocean Dyn. 60 (6) (2010) 1415–1445.
[66] A. Uranga, P.O. Persson, M. Drela, J. Peraire, Implicit large eddy simulation of transitional flows over airfoils and wings, in: Proceedings of the AIAA Computational Fluid Dynamics Conference, San Antonio, TX, (AIAA Paper 2009-4131), June 2009.
[67] M.R. Visbal, D.V. Gaitonde, On the use of higher-order finite difference schemes on curvilinear and deforming meshes, J. Comput. Phys. 181 (1) (2002) 155–185.