Escola de Camins
Escola Tècnica Superior d'Enginyeria de Camins, Canals i Ports
UPC BARCELONATECH

TREBALL FINAL DE MÀSTER

NURBS-Enhanced Finite Element Hybridizable Discontinuous Galerkin method with degree adaptivity for steady Stokes flow

Treball realitzat per:
Karthik Neerala Suresh

Dirigit per:
Prof Antonio Huerta
Dr Rubén Sevilla

Màster en:
Erasmus Mundus M.Sc. in Computational Mechanics

Barcelona, **13 June 2017**

Departament: **Laboratori de Càlcul Numèric**

UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

NURBS-Enhanced Finite Element Hybridizable Discontinuous Galerkin method with degree adaptivity for steady Stokes flow

submitted in the partial fulfillment
for the Erasmus Mundus Master
in
Computational Mechanics
Centre Internacional de Mètodes Numèrics a l'Enginyeria (CIMNE)

Submitted by
Karthik Neerala Suresh
MSc Computational Mechanics
Universitat Politécnica de Catalunya, BarcelonaTech (UPC)

Supervised by
Prof Antonio Huerta
Universitat Politécnica de Catalunya
Dr Rubén Sevilla
Swansea University

13 June 2017

Abstract

High-order methods have the potential to deliver high accuracy with less degrees of freedom when compared with lower order methods. However high-order methods are still not being utilized in industrially relevant computational fluid dynamics (CFD) problems because of their claimed higher computational cost. The hybridizable discontinuous Galerkin (HDG) method is a novel discontinuous Galerkin (DG) method with very attractive properties. While retaining the advantages of other DG methods like the inherent stabilization and the local conservation properties, the HDG method reduces the total number of degrees of freedom compared to other DG methods such as the local DG or the compact DG. The HDG method achieves this by performing a hybridization similar to the static condensation traditionally used in high-order continuous finite element methods. Another salient feature of the HDG is that an element-by-element postprocessing technique resulting in a superconvergent solution. The superconvergent postprocess solution can be used to define a reliable and computationally cheap error estimator. The main advantages of HDG, and other DG methods, which are found when a high-order functional approximation is considered can, however, be limited by the accuracy of the geometric approximation of curved boundaries when isoparamteric elements are used.

This work proposes the combination of HDG and NURBS-enhanced finite element method (NEFEM) for the solution of Stokes flow problems using degree adaptivity. The ability of NEFEM to exactly represent the boundary of the domain by means of its NURBS boundary representation is used to ensure that the geometric description is independent on the degree of the polynomials used in the functional approximation. As a result, the degree adaptivity process is only driven by the error of the functional approximation and not by the error of both the functional and the geometric approximation.

Acknowledgements

After an intensive period of eight months, today is the day: writing this note of thanks is the finishing touch on my thesis. It has been a period of intense learning for me, not only in the scientific arena, but also on a personal level. Writing this thesis has had a big impact on me. I would like to reflect on the people who have supported and helped me so much throughout this period.

First and foremost, I would like to thank my advisor at UPC, Barcelona. Prof Antonio Huerta has been a great source of inspiration. I have learned a great deal during every interaction with you. Your support and kind words during the toughest phase of my life has indeed helped me grow as a person. Dr Rubén Sevilla has been much more than a supervisor to me. Since the day I arrived at Swansea University you have been a constant support to me. I cannot thank you enough for your excellent cooperation and for all of the opportunities I was given to conduct my research and further my thesis initially at Swansea University and then at UPC, Barcelona. I would have definitely not completed my M.Sc. had it not been for your constant encouragement. I would also like to thank my colleagues at LaCàN, Luca, Jordi and Matteo for their wonderful collaboration. You supported me greatly and were always willing to help me.

My parents need to be thanked for their wise counsel and sympathetic ear. You are always there for me. Then, there are my friends, Salik, Kamal and Xiaolong. We were not only able to support each other by deliberating over our problems and findings, but also helped each other stay happy by talking about things other than just our papers.

Finally, three amazing people have helped me the most to stay focused and relaxed during the days of this project. I would have been in no position to write this thesis had it not been for these incredible people: my classmate and most reliable friend Sai Chandana, my sister Amrutha and my sweetheart Surabhi. I thank you whole-heartedly for putting up with my tantrums and anxiety.

Thank you very much, everyone!

Karthik Neerala Suresh
Barcelona, June 13, 2017.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

The last few years has witnessed an increase in enthusiasm in the development of high-order methods within the finite element (FE) community. This has been a hot topic of research all over the world, especially for Euler and Navier-Stokes equations [1]. High-order approximations have proved to be of special importance while solving high Reynolds number and transient fluid problems [1]. The ability and efficiency of high-order approximations have been discussed and proven in recent papers [2]. Developments of high-order numerical methods for unstructured meshes could offer significant advantages for the simulation of complex high-speed flows and compressible turbulence [3]. The discontinuous Galerkin (DG), the spectral volume (SV) and the spectral difference (SD) methods have shown promise for high-resolution computations of complex flows because they have a compact stencil, and retain the design order of accuracy even for meshes of moderate quality that would often result from grid generation over complex three-dimensional configurations [4–7]. Specifically, the interest in DG methods has increased over the past years. This is because of their proven suitability to construct robust stabilized high-order numerical schemes on arbitrary unstructured and non-conforming grids for a variety of physical phenomena, such as, for the solution of incompressible flows [8].

DG are finite element methods that are locally conservative and stable, and allow to achieve high-order accuracy. The DG formulation uses discontinuous approximation element by element, with the information that passes through the elements by means of numerical fluxes. Since their introduction, DG methods have been used to solve a large variety of partial differential equations, gaining an increasing interest in the scientific computing community. The main drawback of DG methods, compared to continuous Galerkin (CG), is the increased number of

degrees of freedom (DOF) due to the duplication of the nodes belonging to faces of the elements. It has been often claimed that this disadvantage becomes less important for high-order elements, due to the increasing number of interior nodes. However, until very recently, DG methods coupled the nodes in the interior of the elements, for high-order approximations, with neighboring element nodes, even for reduced stencil approaches such as interior penalty methods, see for [9], and Compact Discontinuous Galerkin method (CDG), see [10]. This coupling precluded any technique, such as static condensation of the interior nodes for CG, to reduce the number of DOF. Under these circumstances, DG are penalized respect to CG.

This is where the Hybridizable Discontinuous Galerkin (HDG) method, with its stability features, its reduced number of degrees of freedom, and its superconvergence properties has attracted attention among all DG methods for implicit schemes [11–16]. Similar to static condensation, hybridization of DG methods allows to reduce the globally coupled degrees of freedom to those of an approximation of the solution defined only on the boundaries of the elements (trace solution). The HDG method, was formally introduced for second order elliptic problems [12], and then extended to the solution of a great variety of physical fenomena, see for example [11, 17–20]. The similarity between hybridization in DG and static condensation in CG is highlighted in [21], both techniques leading to an important reduction in the number of DOF for high-order computations. Nevertheless, it can be seen in [22] that hybridization is not only an implementation expedient to reduce the linear system size, but the trace unknown contains extra informations on the exact solution. In fact, unlike other DG methods and standard CG, HDG exhibits optimal convergence in $\mathcal{L}_2$ norm not only for the primal unknown of the problem but also for its derivative, opening the path to an element-by-element postprocess which provides a superconvergent solution.

However, in order to be competitive, these methods have to be designed in such a way that the increased associated computational complexity is more than balanced. Despite the advantages and capabilities, some authors have maintained that the high-order methods are not yet mature and current implementations are strong limitations for their applications to large scale industrial problems [1]. The efficiency, both in terms of memory and computing time, and robustness of these high-order methods, in particular for turbulent high Reynolds number flows will be decisive in making these methods attractive for industries such as the aerodynamic aircraft design industry [1]. Adaptive mesh refinement is a well known strategy for reducing the cost of a computational simulation while achieving a given level of accuracy [1, 23]. The superconvergence properties of the HDG method can come in handy while building in an automatic a posteriori error estimate based adaptive mesh refinement. However, this induces fresh problems. A local p-adaptive mesh refinement carried out by utilising the superconvergence property of the HDG

method does not necessarily guarantee a reduction in the global error. This might be as a result of higher geometric errors emerging from the change in the boundary representation due to a local p-adaptive mesh refinement. Hence in addition to efficiency and robustness, higher-order boundary representation also continues to be a major obstacle in the path of introducing high-order methods into industrial design processes [1].

It is in the domain of high-order FE methodologies for the treatment of curved boundaries where NURBS-Enhanced Finite Element Method (NEFEM) has carved a niche for itself. For high-order methods to be viable in comparison with conventional FE techniques, it becomes quite essential to use considerably large elements while the variables of interest must have fewer degrees of freedom [24]. Also an interpolation polynomial of high degree must be adopted. However, the use of isoparametric elements will not be suitable in the adoption of the afore mentioned parameters, as it will lead to unsustainable geometric errors that will do away with the benefits of using high-order methods [25]. NEFEM, originally designed as a technique that allows a seamless integration of the CAD boundary representation of the domain and the finite element method (FEM) [24], can easily be extended to high-order methods such as to the HDG method. The work presented in this report has been carried forward to establish a seamless integration between NEFEM and the HDG method. While the HDG method comes with all the advantages of a high-order method, NEFEM helps tackle the issues of geometric error engendering out of the use of isoparametric elements. Combined, a method encompassing both NEFEM and the HDG method, seamlessly integrated with one another offers hope to overcome all of the previously mentioned obstacles lingering in the path of introducing high-order methods into industrial design processes.

1.2 Objective of the research

The main objective of this research is to study the HDG formulation for Stokes problems and to develop and validate an arbitrary order HDG solver in two and three dimensions that can be extended in the future to the full system of the Navier-Stokes equations. In addition NEFEM has to be studied to develop and validate an arbitrary order NEFEM-HDG code in two dimensions. Finally, the a degree adaptive technique is developed for HDG providing high-order accuracy and high computational efficiency.

In order to achieve this goals, the following objectives are considered:

- The development of HDG formulation for the numerical solution of the Stokes equations in two and three dimensions. The derivation of the Stokes equations are presented in Chapter 2. The formulation is presented in detail, from the strong form of the problem to the HDG formulation alongside with

the discretized version of the systems of equations that are obtained. Two different formulations of the HDG method are developed for Stokes equations, the so called classical (Dirichlet) and the Neumann formulations. This is presented in chapter 3.

- The discretization of the weak form of the HDG formulation for Stokes equatiosn using the isoparametric and NEFEM formulations. Discretization using NEFEM formulation is presented explaining the details of high-order approximation and numerical integration in curved elements. This is presented in the first section of chapter 4.

- The development of degree adaptive technique based on the concept of Richardson extrapolation using the postprocessed solution obtained using the HDG method for error estimation. The Richardson extrapolation for h-adaptivity is studied and then concept is extended to degree adaptivity. Also come caveats concerning the implementation, such as compatibility of numerical integration and geometrical compatibility of the mesh upon adaptive refinement are detailed.

- Validate the implementation of the HDG codes. Numerical examples with analytical solution are considered in two and three dimensions for validating the codes developed to solve the Stokes equations. In all cases the objective is to test the optimal convergence of the solution and the implementation of the postprocess that offers superconvergence properties in the HDG framework. The workability of the code using both the formulations of HDG are presented. Further the ability of the code to handle meshes with different types of elements in both two and three dimensions are shown. Also the code is validated to work with meshes that have varying degrees of approximations in their elements. Code developed for NEFEM-HDG formulation applied to Stokes' equations is validated. Finally cases to show the performance of degree adaptive method providing high-order accuracy and high computational efficiency are presented. All these examples are collectively shown in Chapter 5.

Chapter 2

Derivation of Stokes' equations

In certain fluids viscosity may be so large that it only flows with difficulty. Heavy oils, honey, even tight crowds of people, show insignificant effects of inertia, and are instead dominated by internal friction. Such fluids do not make spinning vortices or become turbulent, but rather ooze or creep around obstacles. Fluid flow which is dominated by viscosity is quite appropriately called creeping flow.

Stokes flow, another name for creeping flow which is sometimes called creeping motion, is a type of fluid flow where advective inertial forces are small compared with viscous forces [26]. This is a typical situation in flows where the fluid velocities are very slow, the viscosities are very large, or the length-scales of the flow are very small. Though creeping flow was first studied to understand lubrication, in nature this type of flow occurs in the swimming of microorganisms and sperm [27] and the flow of lava. Blood flowing through a microscopic capillary can be as sluggish as heavy oil. Tiny organisms like bacteria live in air and water like ourselves, but theirs is a world of creeping and oozing rather than whirls and turbulence, and movement requires special devices, for example oar-like cilia or whip-like flagella [28]. Some bacteria have even mounted a rotating helical tail in a journal bearing (the only one known to biology), which like a corkscrew allows them to advance through the thick fluid that they experience water to be. A spermatozoan pushes forward like a slithering snake in the grass by undulating its tail. In technology, creeping flow occurs in paint, MEMS devices, and in the flow of viscous polymers generally.

2.1 The basic conservation equations

To serve as an introduction to the discussion on incompressible Stokes flow, in this section the differential forms of the conservation equations for mass, momentum and energy are presented.

2.1.1 Mass equation

A fundamental law of Newtonian mechanics is the conservation of the mass contained in a material volume. The law of mass conservation for a time varying material volume V_t occupied by fluid is given by [29]

$$0 = \frac{dM}{dt} = \frac{d}{dt}\int_{V_t} \rho \, dV, \tag{2.1}$$

where ρ is the fluid density. The material time derivative of the integral of a scalar function $f(\boldsymbol{x}, t)$ (where f is defined in the spatial domain) over V_t is given by the following well-known expression, often referred to as the Reynolds transport theorem:

$$\frac{d}{dt}\int_{V_t} f(\boldsymbol{x}, t) \, dV = \int_{V_c \equiv V_t} \frac{\partial f(\boldsymbol{x}, t)}{\partial t} \, dV + \int_{S_c \equiv S_t} f(\boldsymbol{x}, t)\boldsymbol{u} \cdot \boldsymbol{n} \, dS, \tag{2.2}$$

which holds for smooth functions $f(\boldsymbol{x}, t)$. Here $\boldsymbol{u}$ is the velocity of the fluid. The volume integral on the r.h.s. is defined over a control volume V_c (fixed in space) which coincides with the moving material volume V_t at the considered instant, t, in time. Similarly, the fixed control surface S_c coincides at time t with the closed surface S_t bounding the material volume V_t.

Applying to integral expression in (2.1), the formula in (2.2) for the rate of change of integrals over a moving volume and the divergence theorem, one obtains

$$0 = \frac{dM}{dt} = \int_{V_t} \frac{\partial \rho}{\partial t} \, dV + \int_{S_t} \rho \boldsymbol{u} \cdot \boldsymbol{n} \, dS = \int_{V_t} \left(\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u}) \right) \, dV.$$

Since this relation is valid for all choices of the volume V_t, the integrand must be identically zero. Hence

$$\frac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u}) = 0. \tag{2.3}$$

at all points in the fluid. Equation (2.3) is the mass-conservation equation, or continuity equation.

2.1.2 Momentum equation

The momentum equation, also termed the equation of motion, is a relation equating the rate of change of momentum of a selected portion of fluid and the sum of all forces acting on that portion of fluid. For the portion of fluid of volume V_t enclosed by the material surface S_t , the momentum is $\int_{V_t} \rho \boldsymbol{u} \, dV$ and making use

of the Reynolds transport theorem in vector form, that is (2.2), its rate of change is found to be

$$\begin{aligned}\frac{d}{dt}\int_{V_t}\rho\boldsymbol{u}\,dV &= \int_{V_c\equiv V_t}\frac{\partial\rho\boldsymbol{u}}{\partial t}\,dV + \int_{S_c\equiv S_t}(\rho\boldsymbol{u}\otimes\boldsymbol{u})\cdot\boldsymbol{n}\,dS \\ &= \int_{V_t}\left(\frac{\partial\rho\boldsymbol{u}}{\partial t} + \boldsymbol{\nabla}\cdot(\rho\boldsymbol{u}\otimes\boldsymbol{u})\right)\,dV.\end{aligned} \tag{2.4}$$

where the notation $\boldsymbol{u}\otimes\boldsymbol{u}$ denotes the tensor $[u_iu_j], i,j = 1,...,\mathtt{n_{sd}}$. Here $\mathtt{n_{sd}}$ is the number of spatial dimensions.

The the well-known equation that relates the material and the spatial time derivatives is given by

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \boldsymbol{u}\cdot\boldsymbol{\nabla}f. \tag{2.5}$$

Making use of the continuity equation (2.3) and expression for the material time derivative given in formula (2.5), Equation (2.4) can be transformed to

$$\frac{d}{dt}\int_{V_t}\rho\boldsymbol{u}\,dV = \int_{V_t}\rho\frac{d\boldsymbol{u}}{dt}\,dV, \tag{2.6}$$

which is simply the sum of the product of mass and acceleration for all the elements dV of the material volume V_t.

In general, a portion of fluid is acted upon by both volume and surface forces. The volume force per unit mass of fluid is denoted by $\boldsymbol{s}$, so that the total volume force on the selected portion of fluid is $\int_{V_t}\rho\boldsymbol{s}\,dV$. On the other hand, the i-component of the surface force exerted across a surface element of area dS and normal $\boldsymbol{n}$ is given by $\sigma_{ij}n_{ij}dS$, so that the total force exerted on the selected portion of fluid by the surrounding matter can be expressed in terms of the Cauchy stress $\boldsymbol{\sigma}$ as

$$\int_{S_t}\boldsymbol{\sigma}\cdot\boldsymbol{n}\,dS = \int_{V_t}\boldsymbol{\nabla}\cdot\boldsymbol{\sigma}\,dV,$$

The momentum balance for the selected material volume of fluid, which accounts for both previous actions, is expressed by

$$\int_{V_t}\rho\frac{d\boldsymbol{u}}{dt}\,dV = \int_{V_t}\rho\boldsymbol{s}\,dV + \int_{V_t}\boldsymbol{\nabla}\cdot\boldsymbol{\sigma}\,dV,$$

This integral relation holds for all choices of the material volume V_t . Thus,

$$\rho\frac{d\boldsymbol{u}}{dt} = \rho\boldsymbol{s} + \boldsymbol{\nabla}\cdot\boldsymbol{\sigma}. \tag{2.7}$$

at all points of the fluid. This equation is the equation of motion; making use of (2.5) it becomes

$$\frac{\partial \rho \boldsymbol{u}}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u} \otimes \boldsymbol{u}) - \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} = \rho \boldsymbol{s}. \tag{2.8}$$

This is the conservation form of the momentum equation.

2.1.3 Total energy equation

The total energy per unit mass of the fluid is the sum

$$E = e + \frac{1}{2}||\boldsymbol{u}||^2$$

of the internal energy, e, and the kinetic energy. The rate of change of the total energy per unit mass of a material element is thus

$$\frac{dE}{dt} = \frac{de}{dt} + \boldsymbol{u} \cdot \frac{d\boldsymbol{u}}{dt}. \tag{2.9}$$

It can be shown that in the absence of heat transfer effects, the rate of change of internal energy per unit mass of a material element of fluid is thus given by [29]

$$\frac{de}{dt} = \frac{\boldsymbol{\sigma}}{\rho} : \boldsymbol{\nabla} \boldsymbol{u} \tag{2.10}$$

Making use of the momentum equation (2.8) and of the internal energy equation (2.10), it becomes

$$\rho \frac{dE}{dt} = \boldsymbol{\sigma} : \boldsymbol{\nabla} \boldsymbol{u} + \boldsymbol{u} \cdot (\rho \boldsymbol{s} + \boldsymbol{\nabla} \cdot \boldsymbol{\sigma})$$

In terms of the partial time derivative, the total energy equation can be written as

$$\frac{\partial}{\partial t}(\rho E) + \boldsymbol{\nabla} \cdot (\rho E \boldsymbol{u}) - \boldsymbol{\nabla} \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{v}) = \boldsymbol{u} \cdot \rho \boldsymbol{s}. \tag{2.11}$$

2.2 Simplifying compressible Navier-Stokes equations to Stokes equations

From the previous section the conservation equations in Eulerian form for mass, momentum and energy can be written as

$$\begin{cases} \text{Mass:} & \dfrac{\partial \rho}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u}) = 0, \\ \text{Momentum:} & \dfrac{\partial \rho \boldsymbol{u}}{\partial t} + \boldsymbol{\nabla} \cdot (\rho \boldsymbol{u} \otimes \boldsymbol{u}) - \boldsymbol{\nabla} \cdot \boldsymbol{\sigma} = \rho \boldsymbol{s}, \\ \text{Total energy:} & \dfrac{\partial}{\partial t}(\rho E) + \boldsymbol{\nabla} \cdot (\rho E \boldsymbol{u}) - \boldsymbol{\nabla} \cdot (\boldsymbol{\sigma} \cdot \boldsymbol{v}) = \boldsymbol{u} \cdot \rho \boldsymbol{s}. \end{cases} \tag{2.12a}$$

Typically, Newtonian fluids are considered, that is

$$\boldsymbol{\sigma} = (-p + \lambda \boldsymbol{\nabla} \cdot \boldsymbol{u}) \mathbf{I}_{\mathtt{n_{sd}}} + 2\mu \boldsymbol{\nabla}^{\mathtt{s}} \boldsymbol{u} = -p \mathbf{I}_{\mathtt{n_{sd}}} + \boldsymbol{\sigma}', \tag{2.13}$$

where subindexes are introduced to clarify the dimensions of matrices, $\boldsymbol{\nabla}^{\mathtt{s}} \boldsymbol{u}$ is the rate of deformation (or strain rate) tensor, i.e. $\boldsymbol{\nabla}^{\mathtt{s}} \boldsymbol{u} = \frac{1}{2}(\boldsymbol{\nabla} + \boldsymbol{\nabla}^T)$, p is the thermodynamical pressure, μ is the coefficient of dynamic viscosity of the fluid, and λ is the 2nd coefficient of viscosity. Under Stokes' Hypothesis, $3\lambda + 2\mu = 0$, the bulk viscosity of the fluid is null, i.e. $\lambda + 2\mu/3 = 0$, and that the thermodynamic pressure coincides with the mechanical one, i.e. $p = \mathrm{tr}(\boldsymbol{\sigma})$. $\boldsymbol{\sigma}'$ is the deviatoric stress.

The conservation and constitutive equations (2.12) form a total of $\mathtt{n_{sd}} + 2$ equations. The material parameters μ, λ are characteristic of a given fluid, and thus, a priori, known. However, these $\mathtt{n_{sd}} + 2$ equations contain $\mathtt{n_{sd}} + 3$ unknowns, ρ, $\rho \boldsymbol{u}$, ρE, and p . One further equation comes from the equation of state for pressure, namely $p = p(\rho, T)$, which introduces a new unknown T, thus making a total of $\mathtt{n_{sd}} + 4$ unknowns. The final equation comes from the equation of state for internal energy, $e = e(\rho, T)$. For a calorically perfect gas, such as air, these state equations read $p = \rho R T$ and $e = c_v T$, where R is the gas constant per unit mass and c_v is the specific heat at constant volume. Thus, the usual form for these equations of state are

$$\begin{aligned} T &= \frac{1}{c_v}\big(E - \frac{1}{2}\boldsymbol{u} \cdot \boldsymbol{u}\big), \\ p &= (\gamma - 1)\rho\big(E - \frac{1}{2}\boldsymbol{u} \cdot \boldsymbol{u}\big), \end{aligned} \tag{2.14}$$

where, see [29], the constant $\gamma = c_p/c_v$ is the ratio of the specific heat at constant pressure, c_p, and the specific heat at constant volume, c_v, and relates the specific

heat constants and the gas constant per unit mass, namely

$$c_p = \frac{\gamma}{\gamma - 1} R \qquad \text{and} \qquad c_v = \frac{R}{\gamma - 1}.$$

Thus, equations (2.14) can be rewritten as

$$\begin{aligned} T &= (\gamma - 1)\big(E - \frac{1}{2}\boldsymbol{u} \cdot \boldsymbol{u}\big)/R, \\ p &= (\gamma - 1)\rho\big(E - \frac{1}{2}\boldsymbol{u} \cdot \boldsymbol{u}\big), \end{aligned} \tag{2.15}$$

for a calorically perfect gas.

Remark 1 [Mach and Reynolds numbers] Other pertinent parameters are the Mach and Reynolds numbers. They are used to characterize the flow problem and are defined as

$$\begin{aligned} \text{Characteristic Mach: } & M_{\mathtt{a}} = u_{\text{ref}}/c_{\text{ref}}, \\ \text{Characteristic Reynolds: } & R_{\mathtt{e}} = \rho_{\text{ref}}\, u_{\text{ref}}\, \ell_{\text{ref}}/\mu_{\text{ref}}, \end{aligned}$$

where reference values are used for every variable. ℓ_{ref} is a characteristic length scale, which depends on the problem whereas all the others: velocity u_{ref}, speed of sound c_{ref}, density ρ_{ref}, pressure p_{ref}, and viscosity μ_{ref} are, in general, the inflow/far-field values, viz. $u_{\text{ref}} = \|\boldsymbol{u}^\infty\|$. The speed of sound is defined as $c = \sqrt{\gamma p/\rho} = \sqrt{\gamma RT}$.

2.2.1 Compact form of the conservation equations

Replacing the equations of state (2.15) and the constitute equations (2.13) in the conservation equations (2.12a) the final equations for unsteady viscous compressible flow are obtained as

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}_k}{\partial x_k} - \frac{\partial \boldsymbol{G}_k}{\partial x_k} = \boldsymbol{S}, \tag{2.16}$$

where $\boldsymbol{U}$ is the vector of conservative variables, $\boldsymbol{F}_k$ are the associated advection flux vectors for each spatial dimension ($k = 1, \ldots, \mathtt{n_{sd}}$), $\boldsymbol{G}_k$ are the corresponding diffusion ones, and $\boldsymbol{S}$ is a source term. As usual, repeated indexes indicate sum. These vectors, which are all in $\mathbb{R}^{\mathtt{n_{sd}}+2}$, are defined as follows:

$$\boldsymbol{U} = \begin{pmatrix} \rho \\ \rho \boldsymbol{U} \\ \rho E \end{pmatrix}, \quad \boldsymbol{S} = \begin{pmatrix} 0 \\ \rho \boldsymbol{s} \\ \boldsymbol{U} \cdot \rho \boldsymbol{s} \end{pmatrix},$$

and for $k = 1, \ldots, \mathtt{n_{sd}}$, then

$$\boldsymbol{F}_k = \begin{pmatrix} \rho\, u_k \\ \rho \boldsymbol{U}\, u_k + p\, \boldsymbol{e}_k \\ \rho E\, u_k + p\, u_k \end{pmatrix} = \begin{pmatrix} \rho\, u_k \\ \rho \boldsymbol{U}\, u_k + (\gamma-1)(\rho E - \frac{1}{2}\rho \boldsymbol{U}\cdot\boldsymbol{U})\boldsymbol{e}_k \\ \gamma \rho E\, u_k - (\gamma-1)\frac{1}{2}(\rho \boldsymbol{U}\cdot\boldsymbol{U})\, u_k \end{pmatrix},$$
$$\boldsymbol{G}_k = \begin{pmatrix} 0 \\ \boldsymbol{\sigma}'\cdot\boldsymbol{e}_k \\ \boldsymbol{U}\cdot\boldsymbol{\sigma}'\cdot\boldsymbol{e}_k \end{pmatrix} = \begin{pmatrix} 0 \\ \lambda(\boldsymbol{\nabla}\cdot\boldsymbol{U})\boldsymbol{e}_k + 2\mu \boldsymbol{\nabla}^{\mathtt{S}}\boldsymbol{U}\cdot\boldsymbol{e}_k \\ \lambda(\boldsymbol{\nabla}\cdot\boldsymbol{U})u_k + 2\mu\, \boldsymbol{U}\cdot\boldsymbol{\nabla}^{\mathtt{S}}\boldsymbol{U}\cdot\boldsymbol{e}_k \end{pmatrix},$$

Where $\boldsymbol{e}_k$ is the unit vector in the x_k direction (i.e. the standard canonical basis in $\mathbb{R}^{\mathtt{n_{sd}}}$).

Moreover, the previous equations are usually written and programmed in dimensionless form, namely:

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho\boldsymbol{U} \\ \rho E \end{pmatrix} + \boldsymbol{\nabla}\cdot\begin{pmatrix} \rho\, \boldsymbol{U}^T \\ \rho \boldsymbol{U}\otimes\boldsymbol{U} + (\gamma-1)(\rho E - \frac{1}{2}\rho\boldsymbol{U}\cdot\boldsymbol{U})\mathbf{I}_{\mathtt{n_{sd}}} \\ \left[\gamma\rho E - (\gamma-1)\frac{1}{2}\rho\boldsymbol{U}\cdot\boldsymbol{U}\right]\boldsymbol{U}^T \end{pmatrix}$$
$$- R\mathtt{e}^{-1}\boldsymbol{\nabla}\cdot\begin{pmatrix} \mathbf{0}^T_{\mathtt{n_{sd}}} \\ \lambda(\boldsymbol{\nabla}\cdot\boldsymbol{U})\mathbf{I}_{\mathtt{n_{sd}}} + 2\mu\boldsymbol{\nabla}^{\mathtt{S}}\boldsymbol{U} \\ \left[\lambda(\boldsymbol{\nabla}\cdot\boldsymbol{U})\boldsymbol{U} + 2\mu\, \boldsymbol{U}\cdot\boldsymbol{\nabla}^{\mathtt{S}}\boldsymbol{U}\right]^T \end{pmatrix} = F_{\mathtt{r}}{}^{-2}\begin{pmatrix} 0 \\ \rho\boldsymbol{b} \\ \boldsymbol{U}\cdot\rho\boldsymbol{b} \end{pmatrix}, \quad (2.17)$$

where, in spite of the notation, which is not changed to simplify the presentation, density, ρ, velocity, $\boldsymbol{U}$, and total energy per unit mass, E are in dimensionless form. The characteristic length ℓ_{ref} and velocity v_{ref} introduced in Remark 1 scale the spatial coordinates and velocities. Typically one can chose to define either a characteristic density ρ_{ref} or a characteristic pressure p_{ref} and they verify $p_{\text{ref}} = \rho_{\text{ref}} v_{\text{ref}}^2$. The characteristic time is $\ell_{\text{ref}}/v_{\text{ref}}$, the one for the total energy is v_{ref}^2 and the one for the body forces, $\boldsymbol{b}$, which are also in dimensionless form in (2.17), are scaled with g_{ref}. The Froude number, $F_{\mathtt{r}}$, which is a dimensionless number defined as the ratio of the flow inertia to the external field (typically due to gravity), affects the independent term and is defined as

$$F_{\mathtt{r}} = v_{\text{ref}}/\sqrt{l_{\text{ref}} g_{\text{ref}}}.$$

Remark 2 Some authors prefer to scale the total energy with a characteristic speed of sound, that is using c_{ref}^2 instead of v_{ref}^2. This modifies the last equation in (2.17) into

$$\frac{\partial \rho E}{\partial t} + \boldsymbol{\nabla}\cdot(\gamma\rho E - (\gamma-1)\frac{M_{\mathtt{a}}{}^2}{2}\rho\boldsymbol{u}\cdot\boldsymbol{u})\boldsymbol{u}$$
$$- \frac{M_{\mathtt{a}}{}^2}{R\mathtt{e}}\boldsymbol{\nabla}\cdot\left[\lambda(\boldsymbol{\nabla}\cdot\boldsymbol{u})\boldsymbol{u} + 2\mu\, \boldsymbol{u}\cdot\boldsymbol{\nabla}^{\mathtt{S}}\boldsymbol{u}\right] = \frac{M_{\mathtt{a}}{}^2}{F_{\mathtt{r}}{}^2}\boldsymbol{u}\cdot\rho\boldsymbol{b}.$$

With this scaling it is easy to analyze the incompressibility limit, i.e. $M_{\mathtt{a}} \to 0$, and see uncoupling of the conservation of energy with respect to the other conservation equations.

2.2.2 Simplifications leading to steady Stokes equations

In the case of highly viscous flow such as Stokes flow where convective forces are small compared with viscous forces, the convective terms in the Navier-Stokes equations can often be neglected if compared with the dominant viscous terms. The resulting equations are called equations of Stokes flow [29]. The Reynolds number for such flows is low, i.e. $R_{\mathtt{e}} \ll 1$. The choice of $R_{\mathtt{e}} \ll 1$ is very interesting and an important assumption, for it is relevant to many practical problems, especially in a world where many products of technology, including those manipulating fluids, are shrinking in size.

In this work the steady Stokes problem is considered. That is, in addition to the convective term, the time-dependent terms of the full Navier-Stokes equations are also neglected. Two distinct formulations of Stokes problem are possible. First, the momentum equation is written in terms of the Cauchy stress (also known as stress-divergence form). The Stokes' constitutive law is only invoked after setting the weak form. The advantage of this approach is that it can readily treat problems with fluid constitutive equations more general than the linear Stokes' law. Second, the problem is directly formulated in terms of velocity and pressure. Use is made of Stokes' law and of the incompressibility condition to express the viscous term as the Laplacian of velocity. This is the standard form of the Stokes equations.

Once the governing equations are defined boundary must be adequately prescribed in order to close the problem. Two types of boundary conditions are used in this report: Dirichlet and Neumann. Dirichlet boundary conditions prescribe the value of the unknown function. Neumann conditions impose the normal gradient of the unknown function along the boundary. Robin boundary conditions can also be prescribed, though such boundary conditions are not considered in this work.

2.2.2.1 Formulation in terms of Cauchy stress

In differential form, a steady Stokes problem is stated as follows in terms of Cauchy stress: given the body force $\boldsymbol{\sigma}$, prescribed velocities $\boldsymbol{u}_D$ on the Dirichlet portion Γ_D of the boundary and imposed boundary tractions $\boldsymbol{t}$ on the Neumann portion Γ_N, determine the velocity field $\boldsymbol{u}$ and the pressure field p such that

$$\begin{cases} -\boldsymbol{\nabla}\cdot\boldsymbol{\sigma} = \boldsymbol{s} & \text{in } \Omega, \\ \boldsymbol{\nabla}\cdot\boldsymbol{u} = 0 & \text{in } \Omega, \\ \boldsymbol{u} = \boldsymbol{u}_D & \text{on } \Gamma_D, \\ \boldsymbol{n}\cdot(\boldsymbol{\sigma} = \boldsymbol{t} & \text{on } \Gamma_N, \end{cases} \tag{2.18}$$

A constitutive equation is needed to close the problem. That is, the Cauchy stress, $\boldsymbol{\sigma}$, must be related to velocity, $\boldsymbol{u}$, and pressure, p, that is $\boldsymbol{\sigma} = \boldsymbol{\sigma}(p, \boldsymbol{u})$. The linear Stokes' law can be used as the constitutive equation [29]

$$\boldsymbol{\sigma} = -p\mathbf{I}_{\mathtt{n_{sd}}} + 2\mu\boldsymbol{\nabla}^{\mathbf{s}}\boldsymbol{u} \tag{2.19}$$

2.2.2.2 Formulation in terms of velocity and pressure

The formulation of the Stokes problem in terms of Cauchy stress has the advantage that it is applicable to arbitrary fluid constitutive relations. When the linear Stokes' law can be invoked, it is preferable to start from a strong form of written in terms of velocity and pressure because in this form the velocity components are uncoupled. The steady Stokes problem can then be restated as follows:

$$\begin{cases} -\boldsymbol{\nabla}\cdot(\nu\boldsymbol{\nabla}\boldsymbol{u}) + \boldsymbol{\nabla}p = \boldsymbol{s} & \text{in } \Omega, \\ \boldsymbol{\nabla}\cdot\boldsymbol{u} = 0 & \text{in } \Omega, \\ \boldsymbol{u} = \boldsymbol{u}_D & \text{on } \Gamma_D, \\ -p\boldsymbol{n} + \boldsymbol{n}\cdot(\nu\boldsymbol{\nabla}\boldsymbol{u}) = \boldsymbol{t} & \text{on } \Gamma_N, \end{cases} \tag{2.20}$$

where $\nu = \mu/\rho$ is the kinematic viscosity and p denotes the dynamic pressure. Cauchy stress tensor (normalized by density) can be obtained by Stokes' law (2.19). In the rest of this report, the velocity-pressure formulation of the Stokes equation is considered.

Chapter 3

HDG formulation for Stokes' equations

3.1 Problem statement

Let $\Omega \in \mathbb{R}^{\mathtt{n_{sd}}}$ be an open bounded domain with boundary $\partial\Omega$ and $\mathtt{n_{sd}}$ the number of spatial dimensions. The strong from for Stokes system along with boundary conditions in its velocity-pressure formulation can be written as

$$\begin{cases} -\boldsymbol{\nabla}\cdot(\nu\boldsymbol{\nabla}\boldsymbol{u}) + \boldsymbol{\nabla}p = \boldsymbol{s} & \text{in } \Omega, \\ \boldsymbol{\nabla}\cdot\boldsymbol{u} = 0 & \text{in } \Omega, \\ \boldsymbol{u} = \boldsymbol{u}_D & \text{on } \Gamma_D, \\ -p\boldsymbol{n} + \boldsymbol{n}\cdot(\nu\boldsymbol{\nabla}\boldsymbol{u}) = \boldsymbol{t} & \text{on } \Gamma_N, \end{cases} \tag{3.1}$$

where $\partial\Omega = \overline{\Gamma}_D \cup \overline{\Gamma}_N$, $\overline{\Gamma}_D \cap \overline{\Gamma}_N = \emptyset$, and $\boldsymbol{s} \in \mathcal{L}_2(\Omega)$ is a source term while $\boldsymbol{n}$ is the outward unit normal vector to $\partial\Omega$. Here standard Dirichlet and Neumann boundary conditions are considered. Other mixed (i.e. Robin) boundary conditions can also be imposed.

It is also assumed that Ω is partitioned in $\mathtt{n_{el}}$ disjoint subdomains Ω_i

$$\overline{\Omega} = \bigcup_{i=1}^{\mathtt{n_{el}}} \overline{\Omega}_i, \quad \Omega_i \cap \Omega_j = \emptyset \text{ for } i \neq j,$$

with boundaries Ω_i, which define an internal interface Γ

$$\Gamma := \left[\bigcup_{i=1}^{\mathtt{n_{el}}} \partial\Omega_i\right] \backslash \partial\Omega. \tag{3.2}$$

The corresponding strong form of the Stokes system given in Equation (3.1)

can be written in the broken computational domain as

$$\begin{cases} -\boldsymbol{\nabla}\cdot(\nu\boldsymbol{\nabla}\boldsymbol{u})+\boldsymbol{\nabla}p=\boldsymbol{s} & \text{in } \Omega_i, \quad \text{for } i=1,...,\mathtt{n_{el}} \\ \boldsymbol{\nabla}\cdot\boldsymbol{u}=0 & \text{in } \Omega_i, \quad \text{for } i=1,...,\mathtt{n_{el}} \\ \boldsymbol{u}=\boldsymbol{u}_D & \text{on } \partial\Omega_i\cap\Gamma_D, \\ -p\boldsymbol{n}+\boldsymbol{n}\cdot(\nu\boldsymbol{\nabla}\boldsymbol{u})=\boldsymbol{t} & \text{on } \partial\Omega_i\cap\Gamma_N, \\ [\![\boldsymbol{u}\otimes\boldsymbol{n}]\!]=\boldsymbol{0} & \text{on } \Gamma, \\ [\![-p\boldsymbol{n}+\boldsymbol{n}\cdot(\nu\boldsymbol{\nabla}\boldsymbol{u})]\!]=\boldsymbol{0} & \text{on } \Gamma, \end{cases} \tag{3.3}$$

Since this is the standard velocity-pressure formulation, the Neumann boundary conditions defined here do not correspond to regular stresses but to pseudo-stresses. The *jump* $[\![\cdot]\!]$ operator has been introduced following the definition in [8], such that, along each portion of the interface Γ it sums the values from the element on the left and right of say, Ω_i and Ω_j , namely

$$[\![\odot]\!]=\odot_i+\odot_j$$

The strong from is written in mixed form as a system of first order equations over the broken computational domain as

$$\begin{cases} \boldsymbol{\nabla}\cdot(\nu\boldsymbol{L}+p\boldsymbol{I})=\boldsymbol{s} & \text{in } \Omega_i, \quad \text{for } i=1,...,\mathtt{n_{el}} \\ \boldsymbol{L}+\boldsymbol{\nabla}\boldsymbol{u}=\boldsymbol{0} & \text{in } \Omega_i, \quad \text{for } i=1,...,\mathtt{n_{el}} \\ \boldsymbol{\nabla}\cdot\boldsymbol{u}=0 & \text{in } \Omega_i, \quad \text{for } i=1,...,\mathtt{n_{el}} \\ \boldsymbol{u}=\boldsymbol{u}_D & \text{on } \partial\Omega_i\cap\Gamma_D, \\ \boldsymbol{n}\cdot(\nu\boldsymbol{L}+p\boldsymbol{I})=-\boldsymbol{t} & \text{on } \partial\Omega_i\cap\Gamma_N, \\ [\![\boldsymbol{u}\otimes\boldsymbol{n}]\!]=\boldsymbol{0} & \text{on } \Gamma, \\ [\![\boldsymbol{n}\cdot(\nu\boldsymbol{L}+p\boldsymbol{I})]\!]=\boldsymbol{0} & \text{on } \Gamma, \end{cases} \tag{3.4}$$

where $\boldsymbol{I}$ is the identity tensor of dimension $\mathtt{n_{sd}}$ and $\boldsymbol{L}=-\boldsymbol{\nabla}\boldsymbol{u}$ is a new variable (the second order velocity gradient tensor), which is introduced after splitting the second order momentum conservation equation in two first order equations. The last two equations in Equation (3.4) impose the continuity of velocity and continuity of the normal component of the pseudo-stress across the interior faces respectively.

3.2 Functional and interpolation setting

Before describing the weak forms the usual basic notation for the functional products and discrete spaces is recalled. as usual, $(\cdot,\cdot)_D$ denotes the $\mathcal{L}_2$ scalar product in a generic subdomain D, that is

$$(u,v)_D=\int_D uv\,d\Omega; \quad (\boldsymbol{u},\boldsymbol{v})_D=\int_D \boldsymbol{u}\cdot\boldsymbol{v}\,d\Omega; \quad \text{and} \quad (\boldsymbol{G},\boldsymbol{H})_D=\int_D \boldsymbol{G}:\boldsymbol{H}\,d\Omega,$$

for scalars, vectors and second order tensors respectively.

Analogously, $< \cdot, \cdot >_S$ denotes the $\mathcal{L}_2$ scalar product in any domain $S \subset \Gamma \cup \partial\Omega$, that is

$$< u, v >_S = \int_S uv \, d\Gamma; \quad < \boldsymbol{u}, \boldsymbol{v} >_S = \int_S \boldsymbol{u} \cdot \boldsymbol{v} \, d\Gamma; \quad \text{and} \quad < \boldsymbol{G}, \boldsymbol{H} >_S = \int_S \boldsymbol{G} : \boldsymbol{H} \, d\Gamma$$

again, for scalars, vectors and second order tensors respectively.

The following discrete finite element spaces are also introduced

$$\begin{aligned}
\mathcal{Q}^h(\Omega) &= \{q \in \mathcal{L}_2(\Omega); q|_{\Omega_i} \in \mathcal{P}^k(\Omega_i) \ \forall \Omega_i\} && \subset \mathcal{Q}(\Omega), \\
\mathcal{V}^h(\Omega) &= \{\boldsymbol{v} \in [\mathcal{L}_2(\Omega)]^{\mathtt{n_{sd}}}; \boldsymbol{v}|_{\Omega_i} \in [\mathcal{P}^k(\Omega_i)]^{\mathtt{n_{sd}}} \ \forall \Omega_i\} && \subset \mathcal{V}(\Omega), \\
\mathcal{G}^h(\Omega) &= \{\boldsymbol{G} \in [\mathcal{L}_2(\Omega)]^{\mathtt{n_{sd}} \times \mathtt{n_{sd}}}; \boldsymbol{G}|_{\Omega_i} \in [\mathcal{P}^k(\Omega_i)]^{\mathtt{n_{sd}} \times \mathtt{n_{sd}}} \ \forall \Omega_i\} && \subset \mathcal{G}(\Omega), \\
\mathcal{M}^h(S) &= \{\boldsymbol{\mu} \in [\mathcal{L}_2(S)]^{\mathtt{n_{sd}}}; \boldsymbol{\mu}|_{\Gamma_i} \in [\mathcal{P}^k(\Gamma_i)]^{\mathtt{n_{sd}}} \ \forall \Gamma_i \subset S \subset \Gamma \cup \partial\Omega\} && \subset \mathcal{M}(S),
\end{aligned}$$

where $\mathcal{P}^k(\Omega_i)$ and $\mathcal{P}^k(\Gamma_i)$ are the spaces of polynomial functions of degree at most $k \geq 1$ in Ω_i and Γ_i respectively. $\mathcal{M}^h$ can be defined over all the mesh skeleton interior and exterior faces (or edges in two dimensions).

The element-by-element nodal interpolation of the variables, namely, p, $\boldsymbol{u}$, $\boldsymbol{L}$ and $\hat{\boldsymbol{u}}$, develops from these spaces as

$$p \approx p^h = \sum_{j=1}^{\mathtt{n_{en}}} N_j \mathrm{p}_j \qquad \in \mathcal{Q}^h, \tag{3.5a}$$

$$\boldsymbol{u} \approx \boldsymbol{u}^h = \sum_{j=1}^{\mathtt{n_{en}}} N_j \mathbf{u}_j \qquad \in \mathcal{V}^h, \tag{3.5b}$$

$$\boldsymbol{L} \approx \boldsymbol{L}^h = \sum_{j=1}^{\mathtt{n_{en}}} N_j \mathbf{L}_j \qquad \in \mathcal{G}^h, \tag{3.5c}$$

$$\hat{\boldsymbol{u}} \approx \hat{\boldsymbol{u}}^h = \sum_{j=1}^{\mathtt{n_{fn}}} \widehat{N}_j \hat{\mathbf{u}}_j \qquad \in \mathcal{M}^h(\Gamma \cup \Gamma_N) \text{ or } \mathcal{M}^h(\Gamma), \tag{3.5d}$$

where $\mathrm{p}_j, \mathbf{u}_j, \mathbf{L}_j$ and $\hat{\mathbf{u}}_j$ are nodal values, N_j are polynomial shape functions of order k in each element, $\mathtt{n_{en}}$ is the number of nodes per element, $\widehat{N}_j$ are polynomial shape functions of order k in each element face/edge, and $\mathtt{n_{fn}}$ is the corresponding number of nodes per face/edge. Note that for $\boldsymbol{L}^h$ the nodal values are matrices, whereas for $\boldsymbol{u}^h$ and $\hat{\boldsymbol{u}}^h$ the nodal values are vectors. The vector $\hat{\mathbf{u}}_j$ is defined globally over the mesh skeleton (faces/edges). Its dimension depends on the formulation and corresponds to the number of nodes on $\Gamma \cup \Gamma_N$ or on Γ. More precisely

$$\dim(\hat{\mathbf{u}}) = \mathtt{n_{sd}} \sum_{k=1}^{\mathtt{n_{ef}}} \mathtt{n}_{\mathtt{fn}}^k$$

where $\mathtt{n_{ef}}$ is the number of element faces/edges in the mesh skeleton and $\mathtt{n}^k_{\mathtt{fn}}$ is the number of nodes in the k-th face. The number of element faces/edges in the mesh skeleton always includes those on the interior, i.e. those belonging to Γ. But, depending on the formulation used, $\mathtt{n_{ef}}$ also includes the faces/edges on the Neumann boundary, Γ_N.

The HDG formulation solves the problem in Equation (3.4) in two phases, see [12–16, 19, 30]. First, an element-by-element problem is defined with ($\boldsymbol{L}$, $\boldsymbol{u}$, p) as unknowns, and then a global problem is set up to determine the traces of $\boldsymbol{u}$, denoted by $\hat{\boldsymbol{u}}$, on the element boundaries. The local problem determines $\boldsymbol{L}_i := \boldsymbol{L}|_{\Omega_i}$, $\boldsymbol{u}_i := \boldsymbol{u}|_{\Omega_i}$ and $p_i := p|_{\Omega_i}$ for $i = 1, ..., \mathtt{n_{el}}$ with a new variable $\hat{\boldsymbol{u}}$ along the interface Γ acting as a Dirichlet boundary condition. There are however several options for the detailed implementation. They are presented and discussed in the following sections.

3.3 HDG formulation

HDG formulation is applied to the strong form given in Equations (3.4), which is a first order system of partial differential equations over each element. This is the classical formulation and can be found in series of papers, see [14–16, 19, 30]. As seen in section 3.2, in the HDG scheme, the global unknowns are restricted to the mesh skeleton, more specifically,on $\Gamma \cup \partial\Omega$. Velocity and pressure on the mesh skeleton characterizes the complete problem, as a result of which, once velocity and pressure are know on the mesh skeleton, they can be obtained everywhere else.

The velocity field on the mesh skeleton is denoted by $\hat{\boldsymbol{u}}(x)$ and it is unknown everywhere except for the Dirichlet boundary. To determine velocity gradients, velocity and pressure (i.e. $\boldsymbol{L}$, $\boldsymbol{u}$ and p) inside every element Ω_i the velocity $\hat{\boldsymbol{u}}$ on $\partial\Omega_i$ is imposed (assumed known). In the HDG method, first, the local element-by-element problem with Dirichlet boundary conditions is defined. Second, a global problem is defined to determine $\hat{\boldsymbol{u}}$. The following two sections give a detailed description of the strong and week forms of the local and global problems, respectively.

3.3.1 The HDG local problem

For the strong form of the problem defined in Equation (3.4) and the definitions on the mesh skeleton $(\hat{\boldsymbol{u}}, \rho_i)$, a local element-by-element Stokes problem can be defined for each element. It comprises of the first three equations in Equation (3.4), which are already described element-by-element, along with the velocity

conditions along the mesh skeleton, $\hat{\boldsymbol{u}}$. Thus the local problem equations are,

$$\begin{cases} \boldsymbol{L}+\boldsymbol{\nabla}\boldsymbol{u}=\boldsymbol{0} & \text{in } \Omega_i, \quad \text{for } i=1,...,\mathtt{n_{el}} \\ \boldsymbol{\nabla}\cdot(\nu\boldsymbol{L}+p\boldsymbol{I})=\boldsymbol{s} & \text{in } \Omega_i, \quad \text{for } i=1,...,\mathtt{n_{el}} \\ \boldsymbol{\nabla}\cdot\boldsymbol{u}=0 & \text{in } \Omega_i, \quad \text{for } i=1,...,\mathtt{n_{el}} \\ \boldsymbol{u}=\boldsymbol{u}_D & \text{on } \partial\Omega_i\cap\Gamma_D, \\ \boldsymbol{u}=\hat{\boldsymbol{u}} & \text{on } \partial\Omega_i\backslash\Gamma_D. \end{cases} \tag{3.6}$$

Note that the local problem is a Dirichlet problem. Therefore, pressure is known up to a constant. There are several choices to remove this indeterminacy but here the mean pressure, ρ_i on the element boundary is imposed, namely

$$\int_{\partial\Omega_i} p\,d\Gamma=\rho_i \quad \text{for } i=1,...,\mathtt{n_{el}}. \tag{3.7}$$

It is worth noting that $\hat{\boldsymbol{u}}$ is imposed in the local problem assumed as known. This allows this problem to produce an element-by-element solution $\boldsymbol{L}_i$, $\boldsymbol{u}_i$ and p_i as a function of the unknown $\hat{\boldsymbol{u}}\in\mathcal{L}_2(\Gamma\cup\Gamma_N)$ in each face. Note that these problems can be solved independently element by element. However, since $\hat{\boldsymbol{u}}$ is actually an unknown, in order to ensure the well-posedness of the local problem, it is essential to establish that the net flux of $\hat{\boldsymbol{u}}$ is zero in each element. This is the so-called compatibility condition and it is defined as

$$\int_{\partial\Omega_i\cap\Gamma_D}\boldsymbol{u}_D\cdot\boldsymbol{n}\,d\Gamma+\int_{\partial\Omega_i\backslash\Gamma_D}\hat{\boldsymbol{u}}\cdot\boldsymbol{n}\,d\Gamma=0. \tag{3.8}$$

for $i=1,...,\mathtt{n_{el}}$. This condition, which is incorporated in the global problem, also ensures the solvability of the global problem. This will be clear in the next section.

The weak problem associated to Equation (3.6) and (3.7) becomes: find $(\boldsymbol{L},\boldsymbol{u},p)$ in the appropriate functional spaces for $i=1,...,\mathtt{n_{el}}$ such that

$$\begin{cases} (\boldsymbol{G},\boldsymbol{L})_{\Omega_i}-(\boldsymbol{\nabla}\cdot\boldsymbol{G},\boldsymbol{u})_{\Omega_i}=-<\boldsymbol{\nabla}\cdot\boldsymbol{G},\hat{\boldsymbol{u}}>_{\partial\Omega_i\backslash\Gamma_D}-<\boldsymbol{\nabla}\cdot\boldsymbol{G},\boldsymbol{u}_D>_{\partial\Omega_i\cap\Gamma_D}, & \forall\boldsymbol{G} \\ -(\boldsymbol{\nabla}\boldsymbol{v},\nu\boldsymbol{L})_{\Omega_i}-(\boldsymbol{\nabla}\cdot\boldsymbol{v},p)_{\Omega_i}+<\boldsymbol{v},\boldsymbol{n}\cdot(\nu\boldsymbol{L}+p\boldsymbol{I})>_{\partial\Omega_i}=(\boldsymbol{v},\boldsymbol{s})_{\Omega_i}, & \forall\boldsymbol{v} \\ -(\boldsymbol{\nabla}q,\boldsymbol{u})_{\Omega_i}=-<q,\hat{\boldsymbol{u}}\cdot\boldsymbol{n}>_{\partial\Omega_i\backslash\Gamma_D}-<q,\boldsymbol{u}_D\cdot\boldsymbol{n}>_{\partial\Omega_i\cap\Gamma_D}, & \forall q \\ <p,1>_{\partial\Omega_i}=\rho_i \end{cases} \tag{3.9}$$

where the test functions $(\boldsymbol{G},\boldsymbol{v},q)$ belong to the corresponding functional test spaces, and the unknown velocity $\boldsymbol{u}$ along $\partial\Omega_i\backslash\Gamma_D$ has been replaced by the the velocity along the mesh skeleton $\hat{\boldsymbol{u}}$ as indicated by the fourth equation in Equation (3.6).

It can be seen that the second equation in Equation (3.9) requires integration of $(\nu\boldsymbol{L}+p\boldsymbol{I})$ along the element boundary. For this, it is standard in HDG to define the trace of the fluxes, denoted by $\widehat{(\nu\boldsymbol{L}+p\boldsymbol{I})}$, as follows:

$$\boldsymbol{n}\cdot\widehat{(\nu\boldsymbol{L})+p\boldsymbol{I}} := \begin{cases} \boldsymbol{n}\cdot(\nu\boldsymbol{L}+p\boldsymbol{I})+\tau(\boldsymbol{u}-\boldsymbol{u}_D) & \text{on } \partial\Omega_i\cap\Gamma_D \\ \boldsymbol{n}\cdot(\nu\boldsymbol{L}+p\boldsymbol{I})+\tau(\boldsymbol{u}-\hat{\boldsymbol{u}}) & \text{elsewhere} \end{cases} \tag{3.10}$$

where τ is stabilization parameter. The selection of this parameter has an important effect on the stability, accuracy and convergence properties of the resulting HDG method. This has been discussed and analyzed for a large number of problems by Cockburn and coworkers, see, for instance, [12–16, 19]. Also, it has been proved in past papers that τ ensures the superconvergence of the post-processed solution, see [31, 32].

The weak problem presented in Equation (3.9) is rewritten for the discrete spaces and each term is identified with the corresponding part in the final system of equations. The equation thus obtained characterizes the so-called local problem for the strong form given in Equation (3.6), after substitution of the numerical fluxes defined in Equation (3.10) into the weak form presented in Equation (3.9). This is presented in Equations (3.11). This local problem is solved for each element, that is, for $i = 1, ..., \mathtt{n_{el}}$

$$\overbrace{(\boldsymbol{G},\boldsymbol{L}^h)_{\Omega_i}}^{\mathbf{A}_{LL}\mathbf{L}_i}\overbrace{-(\boldsymbol{\nabla}\cdot\boldsymbol{G},\boldsymbol{u}^h)_{\Omega_i}}^{\mathbf{A}_{Lu}\mathbf{u}_i} = \overbrace{-<\boldsymbol{n}\cdot\boldsymbol{G},\hat{\boldsymbol{u}}^h>_{\partial\Omega_i\setminus\Gamma_D}}^{\mathbf{A}_{L\hat{u}}\hat{\mathbf{u}}_i} \overbrace{-<\boldsymbol{n}\cdot\boldsymbol{G},\boldsymbol{u}_D>_{\partial\Omega_i\cap\Gamma_D}}^{\mathbf{f}_L}, \tag{3.11a}$$

$$\overbrace{(\boldsymbol{\nabla}\boldsymbol{v},\nu\boldsymbol{L}^h)_{\Omega_i}-<\boldsymbol{v},\nu\boldsymbol{n}\cdot\boldsymbol{L}^h>_{\partial\Omega_i}}^{\mathbf{A}_{uL}\mathbf{L}_i}\overbrace{-<\boldsymbol{v},\tau\boldsymbol{u}^h>_{\partial\Omega_i}}^{\mathbf{A}_{uu}\mathbf{u}_i} + \overbrace{(\boldsymbol{\nabla}\cdot\boldsymbol{v},p^h)_{\Omega_i}-<\boldsymbol{v},p^h\boldsymbol{n}>_{\partial\Omega_i}}^{\mathbf{A}_{up}\mathbf{p}_i} = \overbrace{-<\boldsymbol{v},\tau\hat{\boldsymbol{u}}^h>_{\partial\Omega_i\setminus\Gamma_D}}^{\mathbf{A}_{u\hat{u}}\hat{\mathbf{u}}_i} \overbrace{-<\boldsymbol{v},\tau\boldsymbol{u}_D>_{\partial\Omega_i\cap\Gamma_D}-(\boldsymbol{v},\boldsymbol{s})_{\Omega_i}}^{\mathbf{f}_u} \tag{3.11b}$$

$$\overbrace{(\boldsymbol{\nabla}q,\boldsymbol{u}^h)_{\Omega_i}}^{\mathbf{A}_{pu}\mathbf{u}_i} = \overbrace{<q,\boldsymbol{n}\cdot\hat{\boldsymbol{u}}^h>_{\partial\Omega_i\setminus\Gamma_D}}^{\mathbf{A}_{p\hat{u}}\hat{\mathbf{u}}_i} + \overbrace{<q,\boldsymbol{n}\cdot\boldsymbol{u}_D>_{\partial\Omega_i\cap\Gamma_D}}^{\mathbf{f}_p}, \tag{3.11c}$$

$$\overbrace{<p^h,1>_{\partial\Omega_i}}^{\boldsymbol{a}_{pp}\mathbf{p}_i} = \rho_i, \tag{3.11d}$$

These equations allow to determine the coefficients characterizing $(\mathbf{L}_i, \mathbf{u}_i, \mathbf{p}_i)$

as functions of $(\hat{\mathbf{u}}_i, \rho_i)$, and the system shown in Equations (3.11) becomes:

$$\mathbf{A}_i \left\{ \begin{array}{c} \mathbf{L}_i \\ \mathbf{u}_i \\ \mathbf{p}_i \end{array} \right\} = \left\{ \begin{array}{c} \mathbf{f}_L \\ \mathbf{f}_u \\ \mathbf{f}_p \\ 0 \end{array} \right\}_i + \left(\begin{array}{c} \mathbf{A}_{L\hat{u}} \\ \mathbf{A}_{u\hat{u}} \\ \mathbf{A}_{p\hat{u}} \\ \mathbf{0}_{1,\mathtt{n_{sd}n_{fn}}} \end{array} \right)_i \hat{\mathbf{u}}_i + \left\{ \begin{array}{c} \mathbf{0}_{\mathtt{n_{sd}^2 n_{en}},1} \\ \mathbf{0}_{\mathtt{n_{sd}n_{en}},1} \\ \mathbf{0}_{\mathtt{n_{en}},1} \\ 1 \end{array} \right\} \rho_i, \tag{3.12}$$

for $i = 1, ..., \mathtt{n_{el}}$, where

$$\mathbf{A}_i = \left[\begin{array}{ccc} \mathbf{A}_{LL} & \mathbf{A}_{Lu} & \mathbf{0}_{\mathtt{n_{sd}^2 n_{en}},\mathtt{n_{en}}} \\ \mathbf{A}_{uL} & \mathbf{A}_{uu} & \mathbf{A}_{up} \\ \mathbf{0}_{\mathtt{n_{en}},\mathtt{n_{sd}^2 n_{en}}} & \mathbf{A}_{pu} & \mathbf{0}_{\mathtt{n_{en}}} \\ \mathbf{0}_{1,\mathtt{n_{sd}^2 n_{en}}} & \mathbf{0}_{1,\mathtt{n_{sd}n_{en}}} & \boldsymbol{a}_{\rho p} \end{array} \right]_i$$

For implementation purposes, some auxiliary vectors are defined. Equations (3.12) can be written as

$$\left\{ \begin{array}{c} \mathbf{L}_i \\ \mathbf{u}_i \\ \mathbf{p}_i \end{array} \right\} = \mathbf{z}_i^{\mathbf{f}} + \mathbf{Z}_i^{\mathbf{f}} \hat{\mathbf{u}}_i + \mathbf{z}_i^{\rho} \rho_i, \quad \text{for } i = 1, ..., \mathtt{n_{el}}, \tag{3.13}$$

where

$$\mathbf{z}_i^{\mathbf{f}} = \mathbf{A}_i^{+} \left\{ \begin{array}{c} \mathbf{f}_L \\ \mathbf{f}_u \\ \mathbf{f}_p \\ 0 \end{array} \right\}_i ; \quad \mathbf{Z}_i^{\mathbf{f}} = \mathbf{A}_i^{+} \left(\begin{array}{c} \mathbf{A}_{L\hat{u}} \\ \mathbf{A}_{u\hat{u}} \\ \mathbf{A}_{p\hat{u}} \\ \mathbf{0}_{1,\mathtt{n_{sd}n_{fn}}} \end{array} \right)_i \quad \text{and} \quad \mathbf{z}_i^{\rho} = \mathbf{A}_i^{+} \left\{ \begin{array}{c} \mathbf{0}_{\mathtt{n_{sd}^2 n_{en}},1} \\ \mathbf{0}_{\mathtt{n_{sd}n_{en}},1} \\ \mathbf{0}_{\mathtt{n_{en}},1} \\ 1 \end{array} \right\}_i .$$

Here $\mathbf{A}_i^{+}$ are generalized inverse of the rectangular matrix $\mathbf{A}_i$ satisfying the following properties:

(1) Matrices $\mathbf{A}_i^{+}$ of dimensions $\big((\mathtt{n_{sd}^2 n_{en}} + \mathtt{n_{sd}n_{en}} + \mathtt{n_{en}}), (\mathtt{n_{sd}^2 n_{en}} + \mathtt{n_{sd}n_{en}} + \mathtt{n_{en}} + 1)\big)$ are generalized inverse of $\mathbf{A}_i$ as $\boldsymbol{x} = \mathbf{A}_i^{+}\boldsymbol{y}$ is a solution to the equation $\mathbf{A}_i\boldsymbol{x} = \boldsymbol{y}$ for any $\boldsymbol{y}$ such that the equation $\mathbf{A}_i\boldsymbol{x} = \boldsymbol{y}$ is consistent,

(2) Matrices $\mathbf{A}_i$ and $\mathbf{A}_i^{+}$ satisfy the condition $\mathbf{A}_i\mathbf{A}_i^{+}\mathbf{A}_i = \mathbf{A}_i$,

(3) $\mathbf{A}_i\mathbf{A}_i^{+}$ is idempotent and $R(\mathbf{A}_i\mathbf{A}_i^{+}) = R(\mathbf{A}_i)$ or $R(\mathbf{A}_i^{+}\mathbf{A}_i) = R(\mathbf{A}_i)$ where $R(\cdot)$ denotes the rank of the matrix,

for $i = 1, ..., \mathtt{n_{el}}$.

3.3.2 The HDG global problem

The local problem described in the previous section for each element allows to represent the approximations $(\mathbf{L}_i, \mathbf{u}_i, \mathbf{p}_i)$ in terms of the velocities on the mesh skeleton, $\hat{\mathbf{u}}_i$, and a vector of boundary mean pressures ρ_i. The global problems, which will be dealt with in this subsection, are used to determine the aforementioned variables, $\hat{\mathbf{u}}_i$ and ρ_i. The global problem is constructed imposing continuity of velocities and fluxes along the mesh skeleton, which are the so called *transmission conditions*, the Neumann boundary condition and the element-by-element compatibility condition. The transmission conditions were already introduced in (3.4) to ensure inter-element continuity when the broken computational domain formulation was presented,

$$\begin{cases} [\![\boldsymbol{u} \otimes \boldsymbol{n}]\!] = \mathbf{0} & \text{on } \Gamma, \\ [\![\boldsymbol{n} \cdot (-p\boldsymbol{I} + \nu\boldsymbol{L})]\!] = \mathbf{0} & \text{on } \Gamma \end{cases}$$

Note that the first equation in the transmission conditions imposes continuity of $\boldsymbol{u}$ along Γ. But $\boldsymbol{u} = \hat{\boldsymbol{u}}$ on Γ as imposed by the local problems in Equation (3.6). Hence, continuity of the primal variable, $[\![\boldsymbol{u} \otimes \boldsymbol{n}]\!] = 0$, is imposed automatically because $\hat{\boldsymbol{u}}$ is unique for adjacent elements. In summary, the global problem is simply

$$\begin{cases} [\![\boldsymbol{n} \cdot (-p\boldsymbol{I} + \nu\boldsymbol{L})]\!] = \mathbf{0} & \text{on } \Gamma, \\ \boldsymbol{n} \cdot (\nu\boldsymbol{L} + p\boldsymbol{I}) = -\boldsymbol{t} & \text{on } \Gamma_N, \end{cases} \tag{3.14}$$

with the compatibility condition

$$\int_{\partial\Omega_i \cap \Gamma_D} \boldsymbol{u}_D \cdot \boldsymbol{n} \, d\Gamma + \int_{\partial\Omega_i \setminus \Gamma_D} \hat{\boldsymbol{u}} \cdot \boldsymbol{n} \, d\Gamma = 0.$$

The discrete weak form equivalent to this global problem becomes: find $\hat{\boldsymbol{u}}^h \in \mathcal{M}^h$ and $\rho \in \mathbb{R}^{\mathtt{n_{el}}}$ such that

$$\begin{cases} \displaystyle\sum_{i=1}^{\mathtt{n_{el}}} \Big\{ < \boldsymbol{\mu}, \boldsymbol{n} \cdot (\nu\boldsymbol{L}^h + p^h\boldsymbol{I}) >_{\partial\Omega_i \setminus \Gamma_D} + < \boldsymbol{\mu}, \tau\boldsymbol{u}^h >_{\partial\Omega_i \setminus \Gamma_D} \\ \qquad\qquad - < \boldsymbol{\mu}, \tau\hat{\boldsymbol{u}}^h >_{\partial\Omega_i \setminus \Gamma_D} \Big\} = -\displaystyle\sum_{i=1}^{\mathtt{n_{el}}} < \boldsymbol{\mu}, \boldsymbol{t} >_{\partial\Omega_i \cap \Gamma_N} \qquad \forall \boldsymbol{\mu} \in \mathcal{M}^h \\ < \boldsymbol{n} \cdot \hat{\boldsymbol{u}}^h, 1 >_{\partial\Omega_i \setminus \Gamma_D} = - < \boldsymbol{n} \cdot \boldsymbol{u}_D, 1 >_{\partial\Omega_i \cap \Gamma_D} \qquad \text{for } i = 1, ..., \mathtt{n_{el}}. \end{cases}$$

These equations allow to identify the different terms, that will contribute to

the final global system of equations.

$$\sum_{i=1}^{\mathtt{n_{el}}}\left\{\overbrace{<\boldsymbol{\mu},\nu\boldsymbol{n}\cdot\boldsymbol{L}^h>_{\partial\Omega_i\setminus\Gamma_D}}^{(\mathbf{A}_{\hat{u}L})_i\mathbf{L}_i}+\overbrace{<\boldsymbol{\mu},\tau\boldsymbol{u}^h>_{\partial\Omega_i\setminus\Gamma_D}}^{(\mathbf{A}_{\hat{u}u})_i\mathbf{u}_i}+\overbrace{<\boldsymbol{\mu},p^h\boldsymbol{n}>_{\partial\Omega_i\setminus\Gamma_D}}^{(\mathbf{A}_{\hat{u}p})_i\mathbf{p}_i}\right.$$
$$\left.\overbrace{-<\boldsymbol{\mu},\tau\hat{\boldsymbol{u}}^h>_{\partial\Omega_i\setminus\Gamma_D}}^{(\mathbf{A}_{\hat{u}\hat{u}})_i\hat{\mathbf{u}}_i}\right\}=-\sum_{i=1}^{\mathtt{n_{el}}}\overbrace{<\boldsymbol{\mu},\boldsymbol{t}>_{\partial\Omega_i\cap\Gamma_N}}^{(\mathbf{f}_{\hat{u}})_i}, \tag{3.16a}$$

$$\overbrace{<\boldsymbol{n}\cdot\hat{\boldsymbol{u}}^h,1>_{\partial\Omega_i\setminus\Gamma_D}}^{(\mathbf{a}_{\rho\hat{u}})_i\hat{\boldsymbol{u}}_i}=\overbrace{-<\boldsymbol{n}\cdot\boldsymbol{u}_D,1>_{\partial\Omega_i\cap\Gamma_D}}^{(\mathbf{f}_{\rho})_i}\qquad\text{for } i=1,...,\mathtt{n_{el}}. \tag{3.16b}$$

which can be written in matrix as

$$\sum_{i=1}^{\mathtt{n_{el}}}\left\{\mathbf{B}_i\left\{\begin{array}{c}\mathbf{L}_i\\ \mathbf{u}_i\\ \mathbf{p}_i\end{array}\right\}+\left[\mathbf{A}_{\hat{u}\hat{u}}\right]_i\hat{\mathbf{u}}_i\right\}=\sum_{i=1}^{\mathtt{n_{el}}}\left\{\mathbf{f}_{\hat{u}}\right\}_i \tag{3.17a}$$

$$(\mathbf{a}_{\rho\hat{u}})_i\hat{\boldsymbol{u}}_i=(\mathbf{f}_{\rho})_i\qquad\text{for } i=1,...,\mathtt{n_{el}}, \tag{3.17b}$$

where

$$\mathbf{B}_i=\left[\begin{array}{ccc}\mathbf{A}_{\hat{u}L} & \mathbf{A}_{\hat{u}u} & \mathbf{A}_{\hat{u}p}\end{array}\right]_i\qquad\text{for } i=1,...,\mathtt{n_{el}}.$$

This equation clearly identifies the element-by-element contributions of different matrices and vectors. Upon introducing the local solution (see Equations (3.13)) into Equations (3.17a), we get

$$\sum_{i=1}^{\mathtt{n_{el}}}\left\{\mathbf{B}_i\left\{\mathbf{z}_i^{\mathbf{f}}+\mathbf{Z}_i^{\mathbf{f}}\hat{\mathbf{u}}_i+\mathbf{z}_i^{\rho}\rho_i\right\}+\left[\mathbf{A}_{\hat{u}\hat{u}}\right]_i\hat{\mathbf{u}}_i\right\}=\sum_{i=1}^{\mathtt{n_{el}}}\left\{\mathbf{f}_{\hat{u}}\right\}_i,$$

which can be written as

$$\sum_{i=1}^{\mathtt{n_{el}}}\left\{\left[\mathbf{B}\mathbf{Z}^{\mathbf{f}}+\mathbf{A}_{\hat{u}\hat{u}}\right]_i\hat{\mathbf{u}}_i+\mathbf{B}_i\mathbf{z}_i^{\rho}\rho_i\right\}=\sum_{i=1}^{\mathtt{n_{el}}}\left\{\left\{\mathbf{f}_{\hat{u}}\right\}_i-\mathbf{B}_i\mathbf{z}_i^{\mathbf{f}}\right\}.$$

Therefore, the elemental contributions to the global system

$$\widehat{\mathbf{K}}\hat{\mathbf{U}}=\hat{\mathbf{f}} \tag{3.18}$$

are

$$\widehat{\mathbf{K}} = \mathop{\mathbf{A}}_{i=1}^{\mathtt{n_{en}}} \begin{bmatrix} \mathbf{BZ^f} + \mathbf{A}_{\hat{u}\hat{u}} & \mathbf{Bz}^{\rho} \\ \mathbf{a}_{\rho\hat{u}} & 0 \end{bmatrix}_i, \quad \hat{\mathbf{f}} = \mathop{\mathbf{A}}_{i=1}^{\mathtt{n_{en}}} \begin{bmatrix} \mathbf{f}_{\hat{u}} - \mathbf{Bz^f} \\ \mathbf{f}_{\rho} \end{bmatrix}_i.$$

It is very clear that the global system is a non-symmetric matrix. Nevertheless, this system has only velocity as unknowns on the mesh skeleton, $\partial\Omega_i \backslash \Gamma_D$, and one unknown mean pressure per element. It is worth noting that the number of nodes on the mesh skeleton depends on the degree of the polynomials used for approximation, and this number is significantly lower than the number of nodes per element for higher values of the degree of the interpolation polynomial. Since the HDG method is a high order method, this feature is a huge advantage for the method. Also independent of the degree of the approximation polynomial for pressure, there is only one unknown pressure that has to be computed in each element in the global problem. Thus using a high degree approximation polynomial will be advantageous for the HDG method.

3.3.3 Local postprocessing for velocity

The solution of HDG problem defined in Equations 3.18 and 3.12 computes the numerical solution for both $\boldsymbol{u}$ and $\boldsymbol{L}$ with optimal rate of convergence $k+1$ in $\mathcal{L}_2$ norm. This is because the polynomial used to approximate both these variables are of the same degree. It is known that $\boldsymbol{L}$, which is the gradient of velocity, will in reality be one degree lower than $\boldsymbol{u}$, if $\boldsymbol{u}$ was to be a polynomial function. This cognizance is used to develop the concept of superconvergence using local postprocessing, which was proposed in [13, 36, 37]. Local postprocessing can be used to obtain a superconvergent solution for velocity, namely $\boldsymbol{u}^\star$. This is achieved by solving the following equations element-by-element, thus proving be computationally inexpensive.

$$\begin{cases} \boldsymbol{\nabla} \cdot \boldsymbol{\nabla} \boldsymbol{u}^\star = -\boldsymbol{\nabla} \cdot \boldsymbol{L} & \text{in } \Omega_i, \\ \boldsymbol{n} \cdot \boldsymbol{\nabla} \boldsymbol{u}^\star = -\boldsymbol{n} \cdot \boldsymbol{L} & \text{on } \partial\Omega_i, \end{cases} \tag{3.19}$$

with additional constraint

$$\int_{\Omega_i} \boldsymbol{u}^\star \, d\Omega = \int_{\Omega_i} \boldsymbol{u} \, d\Omega, \tag{3.20}$$

where $\boldsymbol{u}^\star \in \mathcal{P}^{k+1}(\Omega_i)$ and k is polynomial degree of approximation for both $\boldsymbol{u}$ and $\boldsymbol{L}$. The weak form of the problem can be written as,

$$(\boldsymbol{\nabla} \boldsymbol{v}, \boldsymbol{\nabla} \boldsymbol{u}^\star)_{\Omega_i} = -(\boldsymbol{\nabla} \boldsymbol{v}, \boldsymbol{L})_{\Omega_i} \quad \forall \boldsymbol{v} \in \mathcal{P}^{k+1}(\Omega_i), \tag{3.21}$$

$$(\boldsymbol{\nabla} \boldsymbol{u}^\star, 1)_{\Omega_i} = (\boldsymbol{\nabla} \boldsymbol{u}, 1)_{\Omega_i}. \tag{3.22}$$

$\boldsymbol{u}^\star$ converges asymptotically at a faster rate of $k+2$ in $\mathcal{L}_2$ norm for uniform degree of the approximation polynomial k.

3.3.4 Neumann local problems

It is possible to further reduce the number of variables that need to be computed by solving the global system. This can be done by a minor modification of the previous formulation which will induce a smaller global problem. It involves prescribing the Neumann boundary conditions already in the corresponding local problem. This modifies the original strong forms in Equations (3.6) and (3.14) into the HDG local and global problems as

$$\begin{cases} \boldsymbol{L} + \boldsymbol{\nabla} \boldsymbol{u} = \boldsymbol{0} & \text{in } \Omega_i, \quad \text{for } i = 1, ..., \mathtt{n_{el}} \\ \boldsymbol{\nabla} \cdot (\nu \boldsymbol{L} + p \boldsymbol{I}) = \boldsymbol{s} & \text{in } \Omega_i, \quad \text{for } i = 1, ..., \mathtt{n_{el}} \\ \boldsymbol{\nabla} \cdot \boldsymbol{u} = 0 & \text{in } \Omega_i, \quad \text{for } i = 1, ..., \mathtt{n_{el}} \\ \boldsymbol{u} = \boldsymbol{u}_D & \text{on } \partial\Omega_i \cap \Gamma_D, \\ \boldsymbol{n} \cdot (\nu \boldsymbol{L} + p \boldsymbol{I}) = -\boldsymbol{t} & \text{on } \partial\Omega_i \cap \Gamma_N, \\ \boldsymbol{u} = \hat{\boldsymbol{u}} & \text{on } \partial\Omega_i \cap \Gamma, \end{cases} \tag{3.23}$$

with the mean pressure, ρ_i on the element boundary imposed in those elements where the local problem is of Dirichlet type,

$$\int_{\partial\Omega_j} p \, d\Gamma = \rho_j \quad \forall \Omega_j \mid \partial\Omega_j \cap \Gamma_N \neq \emptyset. \tag{3.24}$$

and

$$[\![\boldsymbol{n} \cdot (\nu \boldsymbol{L} + p \boldsymbol{I})]\!] = \boldsymbol{0} \quad \text{on } \Gamma. \tag{3.25}$$

with the compatibility condition

$$\int_{\partial\Omega_i \cap \Gamma_D} \boldsymbol{u}_D \cdot \boldsymbol{n} \, d\Gamma + \int_{\partial\Omega_i \setminus \Gamma_D} \hat{\boldsymbol{u}} \cdot \boldsymbol{n} \, d\Gamma = 0 \quad \forall \Omega_j \mid \partial\Omega_j \cap \Gamma_N \neq \emptyset.$$

It also implies a new definition for the numerical traces of the fluxes, thus Equation (3.10) becomes

$$\boldsymbol{n} \cdot \widehat{(\nu \boldsymbol{L} + p \boldsymbol{I})} := \begin{cases} -\boldsymbol{t} & \text{on } \partial\Omega_i \cap \Gamma_N \\ \boldsymbol{n} \cdot (\nu \boldsymbol{L} + p \boldsymbol{I}) + \tau(\boldsymbol{u} - \boldsymbol{u}_D) & \text{on } \partial\Omega_i \cap \Gamma_D \\ \boldsymbol{n} \cdot (\nu \boldsymbol{L} + p \boldsymbol{I}) + \tau(\boldsymbol{u} - \hat{\boldsymbol{u}}) & \text{elsewhere} \end{cases} \tag{3.26}$$

Upon performing all the steps shown in subsections 3.3.1 and 3.3.2, the local and global problems in the discrete space for the Stokes problem for the Neumann formulation are as follow

$$\overbrace{(\boldsymbol{G},\boldsymbol{L}^h)_{\Omega_i}}^{\mathbf{A}_{LL}\mathbf{L}_i}\overbrace{-(\boldsymbol{\nabla}\cdot\boldsymbol{G},\boldsymbol{u}^h)_{\Omega_i}+<\boldsymbol{n}\cdot\boldsymbol{G},\boldsymbol{u}^h>_{\partial\Omega_i\cap\Gamma_N}}^{\mathbf{A}_{Lu}\mathbf{u}_i}$$
$$=\overbrace{-<\boldsymbol{n}\cdot\boldsymbol{G},\hat{\boldsymbol{u}}^h>_{\partial\Omega_i\setminus\partial\Omega}}^{\mathbf{A}_{L\hat{u}}\hat{\mathbf{u}}_i}\overbrace{-<\boldsymbol{n}\cdot\boldsymbol{G},\boldsymbol{u}_D>_{\partial\Omega_i\cap\Gamma_D}}^{\mathbf{f}_L}, \tag{3.27a}$$

$$\overbrace{(\boldsymbol{\nabla}\boldsymbol{v},\nu\boldsymbol{L}^h)_{\Omega_i}-<\boldsymbol{v},\nu\boldsymbol{n}\cdot\boldsymbol{L}^h>_{\partial\Omega_i\setminus\Gamma_N}}^{\mathbf{A}_{uL}\mathbf{L}_i}\overbrace{-<\boldsymbol{v},\tau\boldsymbol{u}^h>_{\partial\Omega_i\setminus\Gamma_N}}^{\mathbf{A}_{uu}\mathbf{u}_i}$$
$$+\overbrace{(\boldsymbol{\nabla}\cdot\boldsymbol{v},p^h)_{\Omega_i}-<\boldsymbol{v},p^h\boldsymbol{n}>_{\partial\Omega_i\setminus\Gamma_N}}^{\mathbf{A}_{up}\mathbf{p}_i}=\overbrace{-<\boldsymbol{v},\tau\hat{\boldsymbol{u}}^h>_{\partial\Omega_i\setminus\partial\Omega}}^{\mathbf{A}_{u\hat{u}}\hat{\mathbf{u}}_i}$$
$$\overbrace{-<\boldsymbol{v},\tau\boldsymbol{u}_D>_{\partial\Omega_i\cap\Gamma_D}-<\boldsymbol{v},\boldsymbol{t}>_{\partial\Omega_i\cap\Gamma_N}-(\boldsymbol{v},\boldsymbol{s})_{\Omega_i}}^{\mathbf{f}_u} \tag{3.27b}$$

$$\overbrace{(\boldsymbol{\nabla}q,\boldsymbol{u}^h)_{\Omega_i}-<q,\boldsymbol{n}\cdot\boldsymbol{u}^h>_{\partial\Omega_i\cap\Gamma_N}}^{\mathbf{A}_{pu}\mathbf{u}_i}$$
$$=\overbrace{<q,\boldsymbol{n}\cdot\hat{\boldsymbol{u}}^h>_{\partial\Omega_i\setminus\partial\Omega}}^{\mathbf{A}_{p\hat{u}}\hat{\mathbf{u}}_i}+\overbrace{<q,\boldsymbol{n}\cdot\boldsymbol{u}_D>_{\partial\Omega_i\cap\Gamma_D}}^{\mathbf{f}_p}, \tag{3.27c}$$

$$\overbrace{<p^h,1>_{\partial\Omega_i}}^{\boldsymbol{a}_{\rho p}\mathbf{p}_i}=\rho_i, \tag{3.27d}$$

and

$$\sum_{i=1}^{\mathtt{n_{el}}}\Bigg\{\overbrace{<\boldsymbol{\mu},\nu\boldsymbol{n}\cdot\boldsymbol{L}^h)>_{\partial\Omega_i\setminus\partial\Omega}}^{(\mathbf{A}_{\hat{u}L})_i\mathbf{L}_i}+\overbrace{<\boldsymbol{\mu},\tau\boldsymbol{u}^h>_{\partial\Omega_i\setminus\partial\Omega}}^{(\mathbf{A}_{\hat{u}u})_i\mathbf{u}_i}$$
$$\overbrace{<\boldsymbol{\mu},p^h\boldsymbol{n}>_{\partial\Omega_i\setminus\partial\Omega}}^{(\mathbf{A}_{\hat{u}p})_i\mathbf{p}_i}\overbrace{-<\boldsymbol{\mu},\tau\hat{\boldsymbol{u}}^h>_{\partial\Omega_i\setminus\partial\Omega}}^{(\mathbf{A}_{\hat{u}\hat{u}})_i\hat{\mathbf{u}}_i}\Bigg\}=0 \tag{3.28a}$$

$$\overbrace{<\boldsymbol{n}\cdot\hat{\boldsymbol{u}}^h,1>_{\partial\Omega_i\setminus\Gamma_D}}^{(\mathbf{a}_{\rho\hat{u}})_i\hat{\mathbf{u}}_i}=\overbrace{-<\boldsymbol{n}\cdot\boldsymbol{u}_D,1>_{\partial\Omega_i\cap\Gamma_D}}^{(\mathbf{f}_\rho)_i}\qquad\text{for } i=1,...,\mathtt{n_{el}}. \tag{3.28b}$$

Equation (3.27) is the weak form of the local problem in Neumann formulation while Equation (3.28a) is the first equation of the global system. Note that, in this case, the dimension of $\hat{\boldsymbol{u}}$ corresponds only to the degrees of freedom along the interior skeleton Γ, which is slightly smaller than in the previous case where unknowns had also to be determined along the Neumann boundary. In terms of computation time this formulation is slightly more advantageous.

Chapter 4

Spatial discretization and degree adaptivity

4.1 Spatial Discretization

This section is dedicated to the presentation of details of techniques employed during the discretization of the spatial solution domain. Two different formulations are used for the spatial discretization namely, the standard isoparametric formulation [38] and the NURBS-enhanced finite element method (NEFEM) [24].

4.1.1 Isoparametric formulation

In this formulation, the interpolation functions and their derivatives are defined in a reference element with local coordinates $\boldsymbol{\xi} = (\xi_1, \xi_2, \xi_3)$ and the isoparametric transformation is used to relate local and Cartesian coordinates, namely,

$$\boldsymbol{x}(\boldsymbol{\xi}) = \sum_{j=1}^{\mathtt{n}_{\mathtt{en}}} \boldsymbol{x}_j N_j(\boldsymbol{\xi}), \tag{4.1}$$

where $\boldsymbol{x}_j$ denote the elemental nodal coordinates at node j, while N_j is the shape function associated with node j. When using an isoparametric formulation for spatial discretization, the primal variable $\boldsymbol{u}$, the dual variable $\boldsymbol{L}$, pressure p, and the hybrid variable $\hat{\boldsymbol{u}}$ appearing in the HDG formulation of Stokes equations are all interpolated in this manner as,

$$\boldsymbol{u}(\boldsymbol{x}) \simeq \boldsymbol{u}^h(\boldsymbol{x}(\boldsymbol{\xi})) = \sum_{j=1}^{\mathtt{n}_{\mathtt{en}}} \boldsymbol{u}_j N_j(\boldsymbol{\xi}),$$

$$\boldsymbol{L}(\boldsymbol{x}) \simeq \boldsymbol{L}^h(\boldsymbol{x}(\boldsymbol{\xi})) = \sum_{j=1}^{\mathtt{n}_{\mathtt{en}}} \boldsymbol{L}_j N_j(\boldsymbol{\xi}),$$

Table 4.1: The reference elements of order k, their approximation spaces and the dimension of these spaces [39].

Element	Approximation space	Dimension
Triangle	$\mathbb{P}_k(\xi_1,\xi_2)=\xi_1^a\xi_2^b, a+b\leqslant k$	$\frac{1}{2}(k+1)(k+2)$
Quadrilateral	$\mathbb{Q}_k(\xi_1,\xi_2)=\xi_1^a\xi_2^b, a,b\leqslant k$	$(k+1)^2$
Tetrahedron	$\mathbb{P}_k(\xi_1,\xi_2,\xi_3)=\xi_1^a\xi_2^b\xi_3^c, a+b+c\leqslant k$	$\frac{1}{6}(k+1)(k+2)(k+3)$
Hexahedron	$\mathbb{Q}_k(\xi_1,\xi_2,\xi_3)=\xi_1^a\xi_2^b\xi_3^c, a,b,c\leqslant k$	$(k+1)^3$
Prism	$\mathbb{P}_k(\xi_1,\xi_2)\otimes\mathbb{P}_k(\xi_3)$	$\frac{1}{2}(k+1)^2(k+2)$
Pyramid	$\mathbb{P}_k(\xi_1,\xi_2,\xi_3)\oplus\sum_{c=0}^{r-1}\left(\frac{\xi_1\xi_2}{1-\xi_3}\right)^{r-k}\mathbb{P}_k(\xi_1,\xi_2)$	$\frac{1}{6}(k+1)(k+2)(2k+3)$

$$p(\boldsymbol{x})\simeq p^h(\boldsymbol{x}(\boldsymbol{\xi}))=\sum_{j=1}^{\mathtt{n_{en}}}p_jN_j(\boldsymbol{\xi}),$$
$$\hat{\boldsymbol{u}}(\boldsymbol{x})\simeq \hat{\boldsymbol{u}}^h(\boldsymbol{x}(\xi))=\sum_{j=1}^{\mathtt{n_{fn}}}\hat{\boldsymbol{u}}_j\widehat{N}_j(\xi),$$

where $\boldsymbol{u}_j$, $\boldsymbol{L}_j$, p_j and $\hat{\boldsymbol{u}}_j$ are nodal values of the respective variables.

Triangles and quadrilaterals are employed to provide a consistent discretization of the spatial solution domain, Ω, for two dimensional problems. In three dimensions, consistent meshes consisting of tetrahedra, hexahedra, prisms and pyramids are used. Apart from the pyramid, which requires special attention, optimal nodal finite elements of arbitrary order are readily defined for all these shapes [39]. For the pyramid, a recently proposed approximation space [40] is adopted. This space is well suited for discontinuous approximations and is optimal, i.e. the a priori error estimate is $\mathcal{O}(h^{k+1})$ in the $\mathcal{L}_2$ norm. The approximation spaces that are employed are summarised in Table 4.1.

In two dimensions, a Fekete nodal distribution is adopted for the triangle [41] and a tensor product of one dimensional Fekete nodal distributions for the quadrilateral. In three dimensions, the nodal distributions proposed in [42] for the tetrahedron and in [40] for the pyramid are used. A tensor product of one dimensional Fekete nodal distributions is used for the hexahedron and a tensor product of triangular and one dimensional Fekete nodal distributions is used for the prism.

4.1.2 NEFEM formulation

This formulation considers nodal polynomial interpolation in each element, see [24] for more details. To ensure reproducibility of polynomials in the physical space,

NEFEM approach defines the approximation directly with Cartesian coordinates, $\boldsymbol{x}$.

The solution domain is considered as an open bounded domain Ω, whose boundary $\partial\Omega$, or a portion of it, is parametrized by NURBS curves in 2D, is considered. For more information on curves, see Appendix A. In 2D every NURBS is assumed to be parametrized by

$$\boldsymbol{C} : [0,1] \rightarrow \boldsymbol{C}([0,1]) \subseteq \partial\Omega \subset \mathbb{R}^2.$$

A regular partition of the domain $\bar{\Omega} = \bigcup_e \bar{\Omega}_e$ in simplices is assumed, such that $\Omega_i \bigcap \Omega_j = \emptyset$, for $i \neq j$. As customary in FE mesh generation codes, it is considered that every curved boundary face belongs to a unique NURBS. As a result of this assumption, one element face can not be defined by portions of two, or more, different NURBS. It is, however, worthwhile to note that the piecewise definition of each NURBS is independent on the mesh discretization. Thus, within one face, the definition of NURBS parametrization can change, that is, there is no necessity for the edges of elements to coincide with knot lines. Note that special attention must to be paid to the numerical integration over elements affected by the NURBS boundary representation upon allowing changes of NURBS parametrization to be independent on the spatial discretization.

When an element does not have any edge or face in contact with NURBS boundaries, it is interpreted and considered as a standard FE in NEFEM. In this work which involves the integration of the HDG method and NEFEM, these elements not in contact with exterior curved boundaries are treated as standard isoparametric elements. Therefore, in the bulk of the domain, interpolation and numerical integration are typical of isoparametric HDG, preserving the computational efficiency of the HDG method. Specific numerical schemes for interpolation and numerical integration are required only in those elements influenced by NURBS boundaries.

4.1.2.1 Curved elements

In NEFEM, curved elements are characterised in terms of the NURBS boundary representation of the domain. The formal definition of curved faces and elements in a NEFEM mesh is given in this section.

Let Γ_e be an edge on the NURBS boundary parametrized by $\boldsymbol{C}$, and $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \partial\Omega$ the two vertices on the NURBS boundary, see Figure 4.1. The edge is defined by

$$\Gamma_e := \boldsymbol{C}([\lambda_1^e, \lambda_2^e,]),$$

where λ_1^e and λ_2^e are the parametric coordinates (in the parametric space of the NURBS) of the end points of Γ_e.

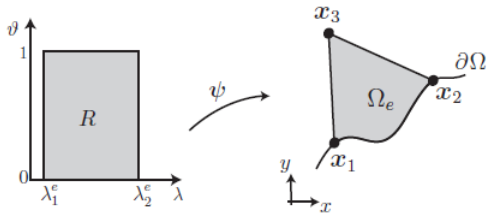


Figure 4.1: Parametrization of a curved triangular element with a edge on the NURBS boundary [24].

A curved triangular element with an edge on the NURBS boundary is defined as a convex linear combination of the curved NURBS edge and the interior vertex. For instance, element represented in Figure 4.1 is parametrized by

$$\begin{aligned} &\boldsymbol{\psi} : R \to \Omega_e, \\ &(\lambda, \vartheta) \mapsto \boldsymbol{\psi}(\lambda, \vartheta) := (1 - \vartheta)\boldsymbol{C}(\lambda) + \vartheta \boldsymbol{x_3}, \end{aligned} \tag{4.2}$$

where $R = [\lambda_1^e, \lambda_2^e] \times [0, 1]$ and $\boldsymbol{x}_3$ is the internal vertex of Γ_e, see Figure 4.1.

4.1.2.2 High-order approximation in curved elements

In this work we propose to use a Cartesian approximation for the variables $\boldsymbol{u}$, $\boldsymbol{L}$ and p, namely,

$$\begin{aligned} \boldsymbol{u}(\boldsymbol{x}) \simeq \boldsymbol{u}^h(\boldsymbol{x}) &= \sum_{j=1}^{\mathtt{n_{en}}} \boldsymbol{u}_j N_j(\boldsymbol{x}), \\ \boldsymbol{L}(\boldsymbol{x}) \simeq \boldsymbol{L}^h(\boldsymbol{x}) &= \sum_{j=1}^{\mathtt{n_{en}}} \boldsymbol{L}_j N_j(\boldsymbol{x}), \\ p(\boldsymbol{x}) \simeq p^h(\boldsymbol{x}) &= \sum_{j=1}^{\mathtt{n_{en}}} p_j N_j(\boldsymbol{x}), \end{aligned}$$

where $\boldsymbol{u}_j$, $\boldsymbol{L}_j$ and p_j are nodal values of the respective variables and N_j are polynomial shape functions (Lagrange polynomials) of order k in $\boldsymbol{x}$. This is in contrast to isoparametric FEM or p-FEM, where the approximation is defined in a reference element. However, in their cases, as opposed to NEFEM, the definition of the polynomial basis for high-order curved elements does not ensure reproducibility of polynomials in the physical space.

The computation of Lagrange polynomial basis is made more systematic, for any degree and for any distribution of nodes, by expressing the Lagrange polynomial basis $\{N_j(\boldsymbol{x})\}_{j=1}^{\mathtt{n_{en}}}$ for a given nodal set with coordinates $\{\boldsymbol{x}_j\}_{j=1}^{\mathtt{n_{en}}}$ in terms of the polynomial basis $\{P_j(\boldsymbol{x})\}_{j=1}^{\mathtt{n_{en}}}$ as

$$N_j(\boldsymbol{x}) = \sum_{l=1}^{\mathtt{n_{en}}} [\boldsymbol{V}^{-1}]_{lj} P_l(\boldsymbol{x}), \tag{4.3}$$

where the multidimensional Vandermonde matrix is defined as $V_{jl} := P_l(\boldsymbol{x}_j)$, for $j, l = 1, ..., \mathtt{n_{en}}$ [24].

A number of options can be considered in order to define a nodal distribution in Ω_e. One can make use of an equally-spaced nodal distribution, or a distribution adapted to curved boundaries. Nodal distributions adapted to curved boundaries do not represent any implementation advantage in the case of HDG method as boundary conditions are imposed in weak form, as usual in DG formulations. However, from [24] it is clear that adapted distributions of nodes induce an important reduction on condition number, even though a specific nodal distribution has to be defined for each curved element.

The approximation for $\hat{\boldsymbol{u}}$ is defined on a reference space, namely,

$$\hat{\boldsymbol{u}}(\boldsymbol{x}) \simeq \hat{\boldsymbol{u}}^h(\boldsymbol{x}(\lambda(\xi))) = \sum_{j=1}^{\mathtt{n_{fn}}} \hat{\boldsymbol{u}}_j \widehat{N}_j(\xi),$$

where $\hat{\boldsymbol{u}}_j$ are the nodal values of $\hat{\boldsymbol{u}}$. The interpolation function $\widehat{\mathbf{N}}$ defined on a reference edge is mapped to the physical space using the NURBS boundary parametrization $\boldsymbol{C}$. Let Γ_e be an edge on the NURBS boundary parametrized by $\boldsymbol{C}$, and $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \partial\Omega$, the two vertices of the NURBS boundary. The edge is given by

$$\Gamma_e := \boldsymbol{C}([\lambda_1^e, \lambda_2^e]),$$

where λ_1^e and λ_2^e are the parametric coordinates (in the parametric space of the NURBS) of the end points of Γ_e. Furthermore a second mapping, which is a linear transformation, is used to relate the parametric space $[\lambda_1^e, \lambda_2^e]$ to the reference face $[-1, 1]$. Thus the face Γ_e is parametrized as

$$\phi : [-1, 1] \longrightarrow [\lambda_1^e, \lambda_2^e] \quad \text{and} \quad \boldsymbol{C} : [\lambda_1^e, \lambda_2^e] \longrightarrow \Gamma_e$$

with

$$\boldsymbol{x}(\lambda(\xi)) = \boldsymbol{C}\left(\left(\frac{1+\xi}{2} \right) \lambda_1^e + \left(\frac{1-\xi}{2} \right) \lambda_2^e \right).$$

The reason for choosing to define $\hat{\boldsymbol{u}}$ is as follows. In an isoparametric formulation, any variable can be fully described along an edge using the values of that variable on the nodes lying on the edge. In other words, at a quadrature point for integrating along the edge, $\hat{\boldsymbol{u}}$ can be described fully as a function of $\hat{\boldsymbol{u}}$'s on the nodes on the edge. This is not true in the case of quantities with Cartesian approximation in the NEFEM formulation. In this case, a variable at a point on the curved edge will be a function of the values of the quantity at all the nodes of the element. As a result of this, upon using a Cartesian approximation for $\hat{\boldsymbol{u}}$, the number of degrees of freedom of $\hat{\boldsymbol{u}}$ along any curved edge will be equal to $\mathtt{n_{en}}$ instead of $\mathtt{n_{fn}}$, which will result in an increase in the size of the global system. In order to overcome this problem $\hat{\boldsymbol{u}}$ is defined in the reference space, thus ensuring that the number of degrees of freedom of $\hat{\boldsymbol{u}}$ along any edge, both planar and curved, will be equal to $\mathtt{n_{fn}}$.

4.1.2.3 Numerical integration in curved elements

From the weak form of the problem it can be seen that integration has to be performed over element faces/edges and in element interiors. In elements that do not have an edge or face in contact with NURBS boundaries, integrals are computed using standard procedures. For an element Ω_e affected by the NURBS boundary representation, it is necessary to design specific quadratures. In addition, while defining suitable quadratures accounting for changes of NURBS parametrization within an element face/edge, it is necessary to pay special attention.

Numerical integration on curved faces/edges (line/surface integrals, usually related to the implementation of natural boundary conditions or to flux evaluation over the face in a DG context) and in curved elements (volume integrals) is presented in [24]. This section gives a brief summary of the same.

A line integral to be computed along a curved boundary edge given by a trimmed NURBS, $\Gamma_e = \boldsymbol{C}([\lambda_1^e, \lambda_2^e])$, is written as

$$\int_{\Gamma_e} f \, dl = \int_{\lambda_1^e}^{\lambda_2^e} f(\boldsymbol{C}(\lambda)) \, \|J_{\boldsymbol{C}}(\lambda)\| \, d\lambda,$$

where f is a generic function (usually polynomial), and $\|J_{\boldsymbol{C}}\|$ denotes the norm of the differential of the NURBS parametrization $\boldsymbol{C}$ (which, in general, is not a polynomial). In order to numerically compute the integral, a 1D numerical quadrature is used, namely

$$\int_{\Gamma_e} f \, dl \approx \sum_{i=1}^{\mathtt{n_{ip}}} f(\boldsymbol{C}(\lambda_i)) \, \|J_{\boldsymbol{C}}(\lambda_i)\| \, \omega_i, \tag{4.4}$$

where λ_i and ω_i are the coordinates and weights of the $\mathtt{n_{ip}}$ integration points in $[\lambda_1^e, \lambda_2^e]$.

Since a NURBS parametrization, $\boldsymbol{C}$, is a piecewise rational function whose definition changes at the breakpoints, it is important that an independent numerical quadrature be considered for each one of the intervals between breakpoints. This is to take into account the discontinuous nature of the parametrization.

A very detailed explanation contrasting simple Gauss-Legendre quadratures and composite quadratures is provided in [24]. The number of integration points needed to integrate all the polynomials of degree less or equal to $2p$, when a polynomial interpolation of degree p is considered in the NEFEM context, increase slightly for simple quadratures, when the desired accuracy is increased from 10^{-6} to 10^{-10}. On the other hand, a similar increase in accuracy leads to a higher increase in computation cost in case of composite quadratures. However in [24], the authors also mention that the use of composite rules is very attractive as it allows the error to be controlled in a straightforward manner.

In NEFEM, there is also a necessity to compute integrals in an element Ω_e with one edge Γ_e on the NURBS boundary, see Figure 4.1. Using the transformation in Equation (4.2), element integrals are computed as

$$\int_{\Omega_e} f \, d\Omega = \int_R f(\boldsymbol{\psi}(\lambda, \vartheta)) \, |J_{\boldsymbol{\psi}}(\lambda, \vartheta)| \, d\lambda \, d\vartheta,$$

where $|J_{\boldsymbol{\psi}}|$ is the determinant of the Jacobian of the transformation $\boldsymbol{\psi}$. The integral can be evaluated using 1D Gauss-Legendre quadratures in each direction as

$$\int_{\Omega_e} f \, d\Omega \approx \sum_{i=1}^{\mathtt{n_{ip}}} \sum_{j=1}^{\mathtt{m_{ip}}} f(\boldsymbol{\xi}_{ij}) \, |J_{\boldsymbol{\psi}}(\lambda_i, \vartheta_j)| \, \omega_i \overline{\omega}_j. \tag{4.5}$$

where $\mathtt{n_{ip}}$ and $\mathtt{m_{ip}}$ are the number of integration points in λ and ϑ directions, respectively, $\boldsymbol{\xi}_{ij} := \boldsymbol{\psi}(\lambda_i, \vartheta_j)$, and $\{\lambda_i, \omega_i\}_{i=1}^{\mathtt{n_{ip}}}$ and $\{\vartheta_j, \overline{\omega}_j\}_{j=1}^{\mathtt{m_{ip}}}$ are the 1D quadrature points and weights for $[\lambda_1^e, \lambda_2^e]$ and $[0, 1]$ respectively.

4.2 Degree adaptivity

Previous papers have highlighted the effectiveness of high-order HDG computations for fluid problems compared to low-order approximations [31,32]. High-order elements provide better accuracy for the same computational cost, or require less computational cost for a desired accuracy level. However, the discontinuous character of the solution in the HDG method opens the path to a further optimization:

p-variable computations, that is, increasing the polynomial degree only where more accuracy is needed.

Typically, adaptive procedures start from a discretization designed to exploit the *a priori* knowledge of the solution (for example, the presence of boundary layers) and to accurately represent the geometry of the domain. Then, adaptivity automatically adjusts the elements locally, relying on error indicators or error estimators computed from the solution itself, in order to provide the necessary spatial resolution and to accurately capture the solution as it evolves.

In this section, a p-adaptive HDG method for Stokes flow problems is proposed. A simple and reliable error estimator is derived exploiting the superconvergent post-process of HDG. Since this involves only element-by-element computations the computational cost of the error estimator is significantly smaller than the cost of computing the solution. The error estimator drives the automatic update of the approximation degree in each element, which is aimed at obtaining a uniform error distribution below a user defined tolerance. The adaptive process is also very fast compared with the computation of the solution as no mesh topology changes are introduced. Despite the simplicity of the proposed error estimator and the adaptive technique, high accuracy computations are obtained.

4.2.1 Error estimation and adaptive strategy based on Richardson extrapolation

The use of *a posteriori* computable error estimates to drive an automatic adaptive process is nowadays a common practice to control the accuracy of the computation and to ensure the quality of the FE solution [46]. Error indicators and error estimators are the two different approaches can be used for assessing the error [47]. Error indicators are cheap to evaluate but are designed for specific problems and do not provide error bounds while error estimators are more accurate and can be used in linear and nonlinear problems [48] and can even produce bounds of the exact solution [49–52]. However, *a posteriori* error estimators have a computational overhead [53] when recovery techniques are used [54] or when equilibrated fluxes must be computed ,and also in flux-free implementations [50]. Therefore, the superconvergent postprocess inherent in HDG is a important asset, which can be readily employed to estimate the error.

4.2.1.1 Richardson extrapolation based size update for h-adaptivity

Based on a priori error estimate, the element error estimate is [55]

$$||u - u_h||_{\Omega_i} \leq C h_i^{k_i + \mathrm{n_{sd}}/2} \quad \text{for } i = 1, ..., \mathrm{n_{el}}, \tag{4.6}$$

where u is the exact solution, u_h is the FE approximation, h_i is the characteristic element size of Ω_i, k_i is the degree of approximation, C is a constant that depends on the element distortion and a measure of the derivatives of the solution u. In fact, Equation (4.6), is written assuming that $k+1$ derivatives of u in Ω are regular enough, thus stating that the local seminorm in $\mathcal{H}^{m+1}(\Omega_i)$ is bounded by the global one, $\mathcal{H}^{m+1}(\Omega)$, see [55] for more details. The one order higher accurate postprocessed solution obtained in the HDG method can be used to get a measure of the error even in the absence of an analytical solution, which is

Suppose we are able to estimate the error, that is, we know,

$$E_i \approx ||u - u_h||_{\Omega_i} \quad \text{for } i = 1, ..., \mathrm{n_{el}}. \tag{4.7}$$

From Equations (4.6) and (4.7)

$$E_i \leq C h_i^{k_i + \mathrm{n_{sd}}/2} \quad \text{for } i = 1, ..., \mathrm{n_{el}}. \tag{4.8}$$

The desired error is ε_i at element Ω_i. If the order of the approximation polynomial is not modified, in order to get the desired order of convergence of $k_i + 1$, the size of the element should be $\widehat{h}_i$ to get

$$\varepsilon_i \leq C \widehat{h}_i^{k_i + \mathrm{n_{sd}}/2} \quad \text{for } i = 1, ..., \mathrm{n_{el}}. \tag{4.9}$$

Equations (4.8) and (4.9) are inequalities as the constant C cannot be determined accurately. It is only possible to determine the upper bound of C as a measure of $k+1$ derivatives of u. Accounting for the worst case, where C is equal to its upper bound, Equations (4.8) and (4.9) can be considered as equalities. Dividing (4.9) by (4.8), we get

$$\frac{\varepsilon_i}{E_i} = \left(\frac{\widehat{h}_i}{h_i} \right)^{k_i + \mathrm{n_{sd}}/2} \quad \text{for } i = 1, ..., \mathrm{n_{el}}, \tag{4.10}$$

from which the expression for $\widehat{h}_i$ can be written as

$$\widehat{h}_i = h_i \left(\frac{\varepsilon_i}{E_i} \right)^{1/(k_i + \mathrm{n_{sd}}/2)} \quad \text{for } i = 1, ..., \mathrm{n_{el}}. \tag{4.11}$$

Equation (4.10) can be further simplified. In case an element is subdivided into smaller elements with subelements of equal characteristic size, $\widehat{h}_i$ can be written in terms of h_i as

$$\widehat{h}_i = h_i f_i \quad \text{for } i = 1, ..., \mathrm{n_{el}}, \tag{4.12}$$

where, f_i is the factor by which size of h_i is changed (Note that $f_i < 1$ when an element is divided into smaller subelements). f_i can be computed independent of the size of the element as

$$f_i = \left(\frac{\varepsilon_i}{E_i}\right)^{1/(k_i + \mathrm{n_{sd}}/2)} \quad \text{for } i = 1, ..., \mathrm{n_{el}}. \tag{4.13}$$

4.2.1.2 Extending Richardon extrapolation to obtain order update for degree adaptivity

The path followed in 4.2.1.1 involved modifying the size of the element in order to obtain results with error smaller than the desired error. An alternate approach to this would be to modify the approximation polynomial, maintaining the size of the element in order to achieve the objective of obtaining solutions with errors smaller than the desired error in each element. Equation (4.9) will then be modified as

$$\varepsilon_i \leq C {h_i}^{k_i + \Delta k_i + \mathrm{n_{sd}}/2} \quad \text{for } i = 1, ..., \mathrm{n_{el}}. \tag{4.14}$$

where Δk_i is the change in the approximation polynomial for each element in order to maintain the error within the desired value.

Following the same sequence of operations as followed in section 4.2.1.1, the expression for Δk_i can be obtained as

$$\Delta k_i = \frac{\log\left(\varepsilon_i / E_i\right)}{\log\left(h_i\right)} \quad \text{for } i = 1, ..., \mathrm{n_{el}}.$$

Since polynomials are used to approximate the solution it is necessary to ensure that the change in the order of approximation of the polynomial in every element is an integer. Taking this into account, the expression for Δk_i is modified as

$$\Delta k_i = \left\lceil \frac{\log\left(\varepsilon_i / E_i\right)}{\log\left(h_i\right)} \right\rceil \quad \text{for } i = 1, ..., \mathrm{n_{el}}. \tag{4.15}$$

where $\lceil \cdot \rceil$ denotes the ceiling function, that is, a function that maps a real number to the smallest following integer.

4.3 Caveats concerning implementation

4.3.1 Ensuring compatibility of numerical integration along faces/edges

In the case when varying orders of approximation are used in elements of the solution domain, approximating the trace of velocity, $\hat{\boldsymbol{u}}$, on the shared faces (edges in

2-D) with the correct order is important to ensure optimal convergence. Thus, on a face shared by two elements with differing orders of approximation polynomials in them, $\hat{\boldsymbol{u}}$ is interpolated using the polynomial whose order is greatest of the two. As a result of this care must be taken while integrating terms involving any of the local variables, $\boldsymbol{L}$, $\boldsymbol{u}$ or p, and the global variable $\hat{\boldsymbol{u}}$.

For clarity, consider two neighbouring elements, Ω_1 and Ω_2, sharing a face Γ_{12}. The orders of approximation used in Ω_1 and Ω_2, respectively, are k_1 and k_2, with $k_1 > k_2$. On face Γ_{12}, $\hat{\boldsymbol{u}}$ is approximated using polynomial of order k_1. While integrating functional composed of local variables and $\hat{\boldsymbol{u}}$ on Γ_{12} to compute elemental contributions of Ω_2, care must be taken to ensure that the quadrature points chosen are the ones that are capable of exactly integrating functional involving product of polynomials of degree k_1. It is a general practice during implementation to evaluate the interpolation functions and their derivatives only at the quadrature points required to exactly integrate polynomials of the order of the interpolation function. However, from the above discussion, it is clear that interpolation functions of order k_2 must be evaluated at the quadrature points required for integrating polynomials of order k_1. In general, in the case when varying order of approximation are used in elements of the solution domain, there is a necessity to evaluate lower order interpolation functions at higher order quadrature points to ensure compatible numerical integration. This is quite important in the context of p-adaptive mesh refinement as this process results in a mesh with varying order of approximation within its elements.

4.3.2 Ensuring geometrical compatibility in meshes with curved elements

When p-adaptive high order isoparametric formulation is used for meshes with curved edges, in addition to ensuring compatibility during numerical integration, establishing geometrical compatibility of shared faces and edges (only edges in 2-D) is also crucial. In isoparametric transformation, the nature of a curved face is decided by the order of the basis functions used for defining the isoparametric transformation used to relate the local and Cartesian coordinates. For instance in 2-D, basis functions of $k = 3$, results in an curved edge which at most a cubic curve. Further, upon increasing the order of the approximation polynomial in the element containing this curved edge during p-adaptivity, new nodes are placed along the cubic curve without changing the nature of the curve. However, upon reducing the order of the approximation, the nature of the curved edge changes. The curved edge becomes a parabola or a straight line upon reducing the order to $k = 2$ or $k = 1$, respectively. Thus, the situation of geometrical incompatibility arises only when the order of approximation in a neighbouring element goes lower

than the initial order of approximation used.

Geometrical incompatibility of the mesh, arising as a result of lowering the order of approximation in one of the elements below the initial order, is fixed by changing the nodal coordinates of the faces and edges shared by all the neighbouring elements with this element. The changed nodal coordinates will ensure the position of nodes on faces and edges shared by elements are same when seen from any element. This, in fact, results in a modified definition of the isoparametric transformation relating the reference space of the face/edge to its physical coordinates. Even though the order of basis functions used in the transformation to relate elemental reference coordinates to physical coordinates is higher in the neighbouring elements, the modified nodal coordinates of the shared face/edge will ensure that the Jacobian of the transformation of this face/edge is the same when seen from any element containing it.

Once the geometrical compatibility of the mesh has been ensured, it becomes imperative to ensure perfect placement of the interior nodes in elements with $k \geqslant 3$ so as to obtain optimal rates of convergence of the variables. Previous studies [24,25] have proved that the position of the interior nodes greatly influences the rate of convergence in high order methods. Incorrect placing of these nodes will result in a loss of high order accuracy of the methods and the rate of convergence obtained will always be 2 [24, 25], irrespective of the order of the approximation function. This is fixed by considering a linear elastic model to deform the modified mesh, hence an a posteriori approach [43].

The equation governing the static deformation of a linear elastic medium, Ω, with closed boundary Γ, is considered in the form

$$\boldsymbol{\nabla} \cdot \sigma = \mathbf{0} \quad \text{in } \Omega_e, \tag{4.16}$$

where the sub index e represents the elements whose faces/edges have been modified for geometrical compatibility of the mesh. The stress tensor is given by

$$\sigma = \lambda tr(\varepsilon) + 2\mu\varepsilon, \tag{4.17}$$

where ε is the deformation tensor and λ and μ denote the Lamé coefficients for the medium. This constitutive relation is often expressed, alternatively, in terms of Young's modulus E and Poisson's ratio ν for the medium [44, 45]. The solution to Equation (4.16) is sought subject to appropriate boundary conditions. In the current context, these conditions will be zero displacement u_D on the face nodes. The elastic parameter values $E = 10$ and $\nu = 0.49$ are used as it has been found that this combination allows the minimum scaled Jacobian to be maximised and, at the same time, allows the condition number of the linear system to be minimised [43]. Thus by elastically deforming the elements whose faces/edges have been modified will ensure accurate placement of the interior nodes.

Chapter 5

Numerical examples

5.1 Validation of HDG for Stokes equations

The first set of examples are performed in order to validate the code written to solve Stokes' equations using the HDG method. Validation of the code is done using convergence plots for both the 2-D and 3-D cases. In both these cases, the workability of the code is shown for both the classical and Neumann formulations of HDG, and also for meshes with different types of elements, namely, triangles and quadrilaterals in case of 2-D, while tetrahedrals and hexahedrals in case of 3-D. Finally results obtained using hybrid meshes comprising of multiple element types are also shown.

5.1.1 The 2-D case

In order to illustrate the results of HDG for Stokes equations for a 2-D case, the model problem in (3.1) is solved in $\Omega :=]0,1[\times]0,1[$ with $\Gamma_N = \{(x,y) \in \partial\Omega | y = 0\}$ and $\Gamma_D = \partial\Omega \backslash \Gamma_N$. The source and boundary conditions are taken such that the analytical solution is given by

$$\boldsymbol{u} = \begin{pmatrix} 2ay - b\lambda \exp(-\lambda y)\cos(\lambda x) \\ b\lambda \exp(-\lambda y)\sin(\lambda x) \end{pmatrix}$$

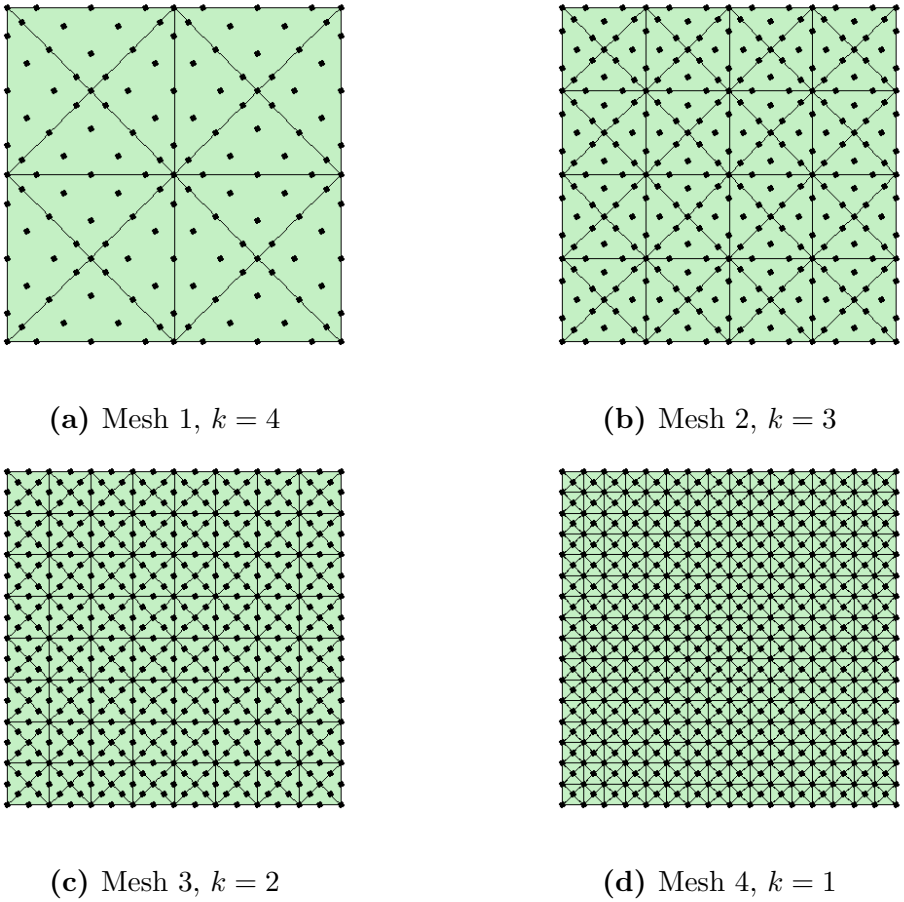
and

$$p = 0.$$

This test case, proposed by [56], is used to test the performance of the HDG for a two dimensional flow that presents a boundary layer type behaviour. The coefficients appearing in the analytical expression of the velocity are set to $a = 1$, $b = 1$ and $\lambda = 10$, leading to large variations of the velocity at the bottom of the

computational domain. The problem is solved in meshes with triangular elements, quadrilateral elements and those containing a mixture of both kinds of elements.

5.1.1.1 Meshes with triangular elements



(a) Mesh 1, $k = 4$ **(b)** Mesh 2, $k = 3$

(c) Mesh 3, $k = 2$ **(d)** Mesh 4, $k = 1$

Figure 5.1: Two dimensional triangular meshes.

Five computational meshes are considered with characteristic element size, h, equal to $1/2\sqrt{2}$, $1/4\sqrt{2}$, $1/8\sqrt{2}$, $1/16\sqrt{2}$ and $1/32\sqrt{2}$ respectively and the order of the approximation, k, in each mesh, is increased from $k = 1$ to $k = 4$. Figure 5.1 shows the first four meshes with the nodal distribution corresponding to a degree of approximation $k = 4$, $k = 3$, $k = 2$ and $k = 1$. The total number of elements and global degrees of freedom in each mesh and for different degree of approximation is summarized in Table 5.1.

Figure 5.2 shows the numerical solution for velocity and pressure computed with quadratic and cubic approximation, $k = 2$ and $k = 3$, in the first and second

of the meshes shown in Figure 5.1. The postprocessed solution, see Section 3.3.3, is also represented in each case. It can be clearly observed that postprocessing offers a significant improvement in the solution.

Table 5.1: Total number of elements and global degrees of freedom in each mesh with triangular and for different degree of approximation

h	$\mathtt{n_{el}}$	$\mathtt{n_{dof}}$ k			
		1	2	3	4
$1/2\sqrt{2}$	16	113	153	193	233
$1/4\sqrt{2}$	64	481	657	833	1009
$1/8\sqrt{2}$	256	1985	2721	3457	4193
$1/16\sqrt{2}$	1024	8065	11073	14081	17089
$1/32\sqrt{2}$	4096	32513	44673	56833	68993

An h-convergence study is performed to validate the implementation. The five meshes described above are considered and the relative error of the numerical solution $\boldsymbol{u}_h$ and the postprocessed solution $\boldsymbol{u}_h^\star$ in the $\mathcal{L}_2(\Omega)$ norm, defined as

$$||e_u||_{\mathcal{L}_2(\Omega)} = \left(\frac{\displaystyle\int_\Omega (\boldsymbol{u_h} - \boldsymbol{u}) \cdot (\boldsymbol{u_h} - \boldsymbol{u}) \, d\Omega}{\displaystyle\int_\Omega \boldsymbol{u} \cdot \boldsymbol{u} \, d\Omega} \right)^{(1/2)} \tag{5.1}$$

and in the $\mathcal{H}^1(\Omega)$ norm, defined as

$$||e_u||_{\mathcal{H}^1(\Omega)} = ||e_u||_{\mathcal{L}_2(\Omega)} + \left(\frac{\displaystyle\int_\Omega (\boldsymbol{\nabla u_h} - \boldsymbol{\nabla u}) : (\boldsymbol{\nabla u_h} - \boldsymbol{\nabla u}) \, d\Omega}{\displaystyle\int_\Omega \boldsymbol{\nabla u} : \boldsymbol{\nabla u} \, d\Omega} \right)^{(1/2)} \tag{5.2}$$

are evaluated in each case.

Figure 5.3 shows the error of the numerical solution and the postprocessed solution as a function of the element size in logarithmic scale for a degree of approximation from $k = 1$ up to $k = 4$, for velocity, postprocessed velocity (5.3c and 5.3d) and pressure (5.3a and 5.3b) in both the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms. The convergence rates are reported in Table 5.2 showing that, in all cases, the optimal rate of convergence for the numerical solution $\boldsymbol{u}_h$ and pressure p_h is obtained, namely $k+1$ in the $\mathcal{L}_2(\Omega)$ norm and k in the $\mathcal{H}^1(\Omega)$ norm. The table also shows

that optimal rate of convergence is obtained for the postprocessed solution $\boldsymbol{u}_h^\star$, namely $k+2$ in the $\mathcal{L}_2(\Omega)$ norm and $k+1$ in the $\mathcal{H}^1(\Omega)$ norm.

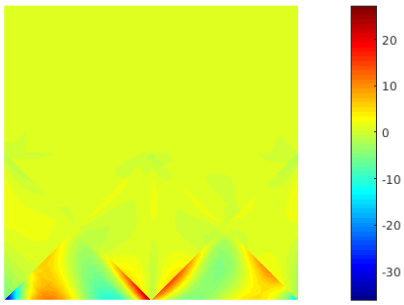
Finally Figure 5.4 shows the error of the numerical solution and the postprocessed solution obtaned using the Neumann formulation of the HDG method as a function of the element size in logarithmic scale for a degree of approximation from $k=1$ up to $k=4$, for velocity, postprocessed velocity (5.4c and 5.4d) and pressure (5.4a and 5.4b) in both the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms. In this case too the convergence rates are reported in Table 5.3 showing that, in all cases, the optimal rate of convergence for the numerical solution $\boldsymbol{u}_h$ and pressure p_h is obtained, namely $k+1$ and k in the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms, respectively and the optimal rate of convergence of $k+2$ and $k+1$ respectively in the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms is obtained for the postprocessed solution $\boldsymbol{u}_h^\star$.

Table 5.2: Rates of convergence for the numerical solution of velocity and pressure and postprocessed solution for 2D Stokes problem for meshes with triangular elements using classical formulation of HDG.

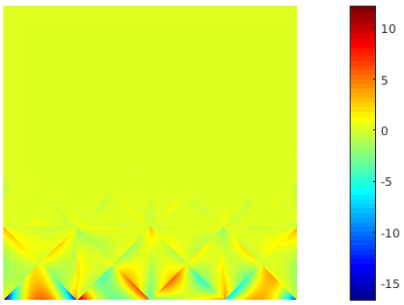
k	$\|\|e\|\|_{\mathcal{L}_2}$			$\|\|e\|\|_{\mathcal{H}^1}$		
	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p
1	2.0	3.0	2.0	1.1	2.0	1.0
2	3.0	4.0	3.0	2.1	3.0	2.0
3	4.0	5.0	4.0	3.1	4.0	3.0
4	5.0	6.0	5.0	4.1	5.0	4.0

Table 5.3: Rates of convergence for the numerical solution of velocity and pressure and postprocessed solution for 2D Stokes problem for meshes with triangular elements using Neumann formulation of HDG.

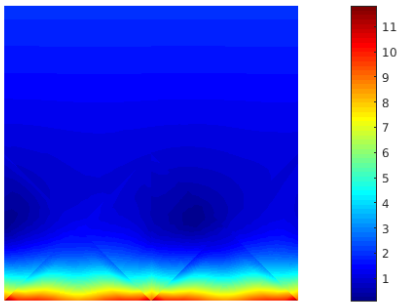
k	$\|\|e\|\|_{\mathcal{L}_2}$			$\|\|e\|\|_{\mathcal{H}^1}$		
	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p
1	2.0	3.0	2.0	1.1	2.0	1.0
2	3.0	4.0	3.0	2.1	3.0	2.0
3	4.0	5.0	4.0	3.1	4.0	3.0
4	5.0	6.0	5.0	4.1	5.0	4.0



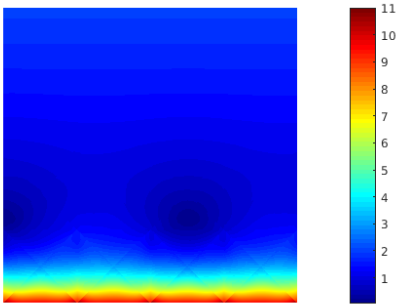
(a) Pressure, Mesh 1, $k = 3$



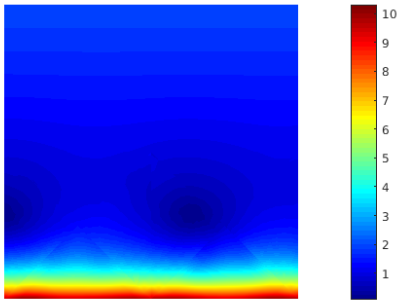
(b) Pressure, Mesh 2, $k = 2$



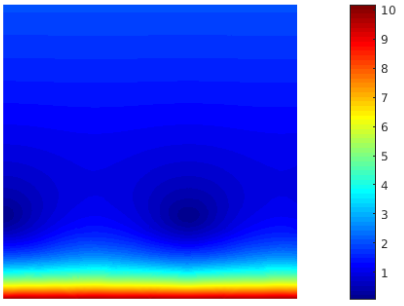
(c) Velocity, Mesh 1, $k = 3$



(d) Velocity, Mesh 2, $k = 2$

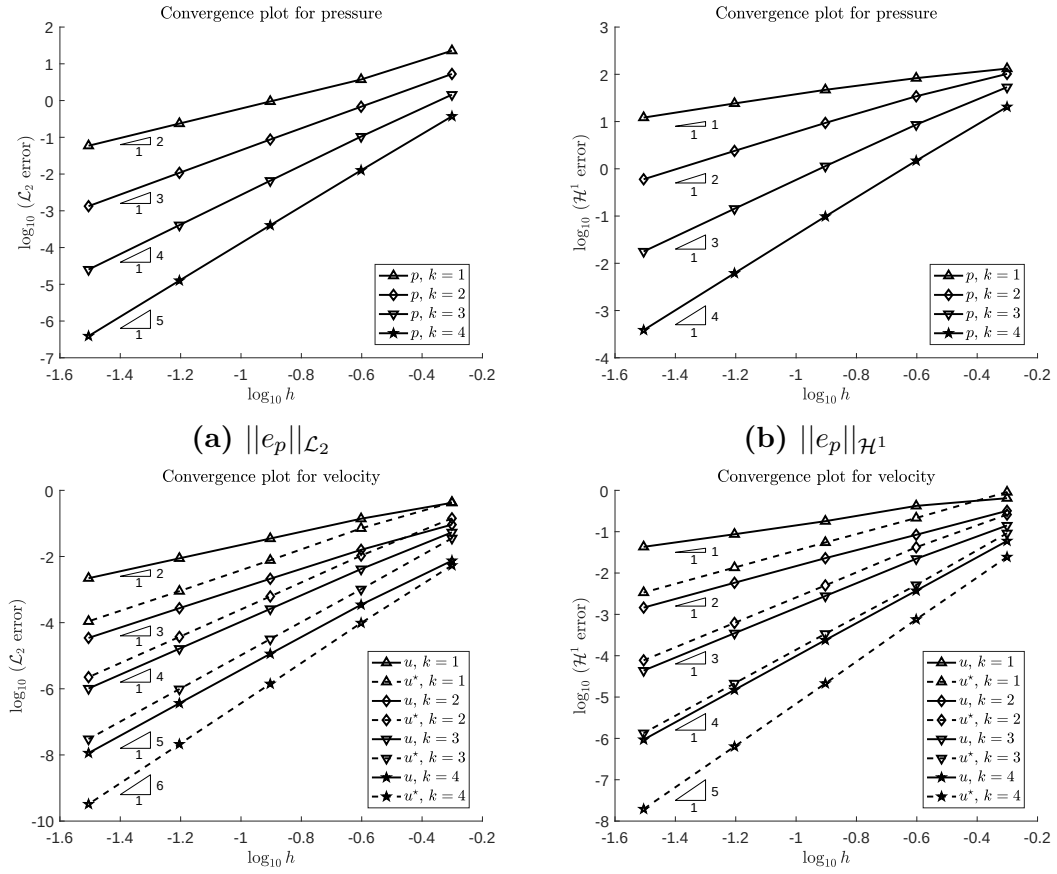


(e) Velocity postprocessed, Mesh 1, $k = 3$



(f) Velocity postprocessed, Mesh 2, $k = 2$

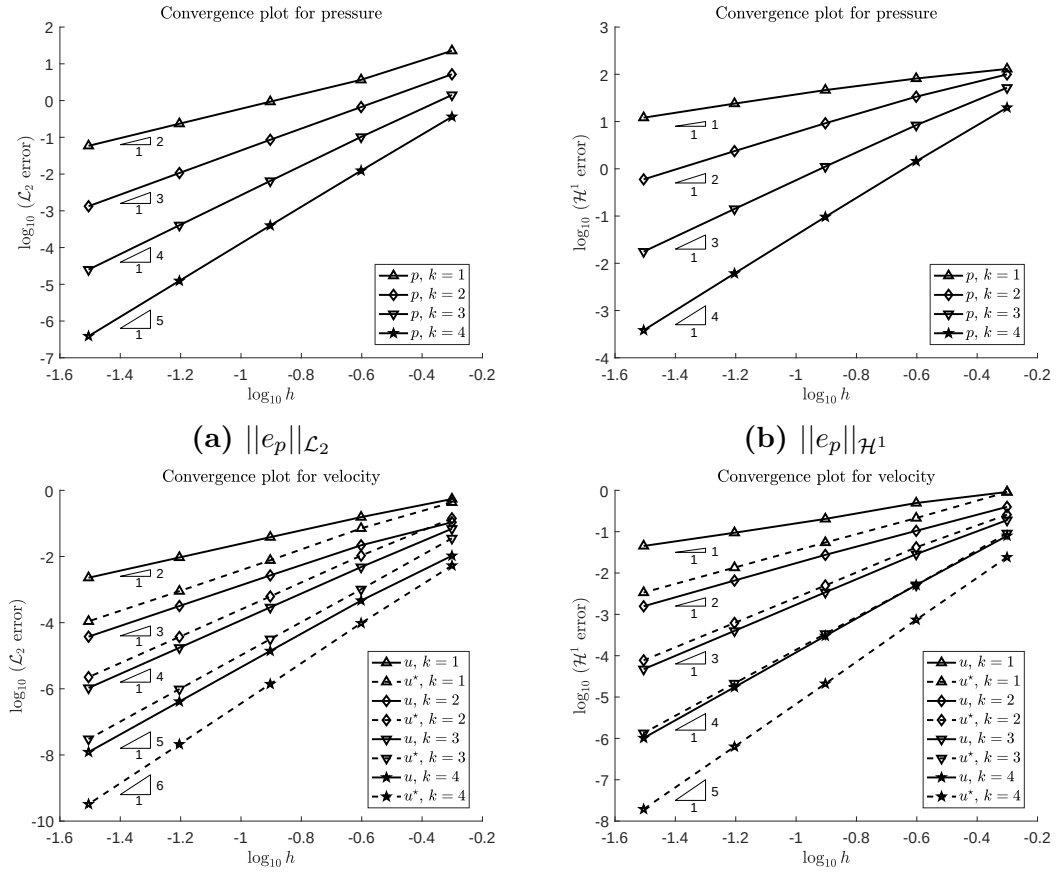
Figure 5.2: Numerical solution for Stokes problem in 2D obtained with two different meshes with triangular elements.



(a) $||e_p||_{\mathcal{L}_2}$ **(b)** $||e_p||_{\mathcal{H}^1}$

(c) $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$ **(d)** $||e_{\boldsymbol{u}}||_{\mathcal{H}^1}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{H}^1}$

Figure 5.3: h-convergence for different degrees of approximation k for 2D Stokes problem for meshes with triangular elements using classical formulation of HDG.

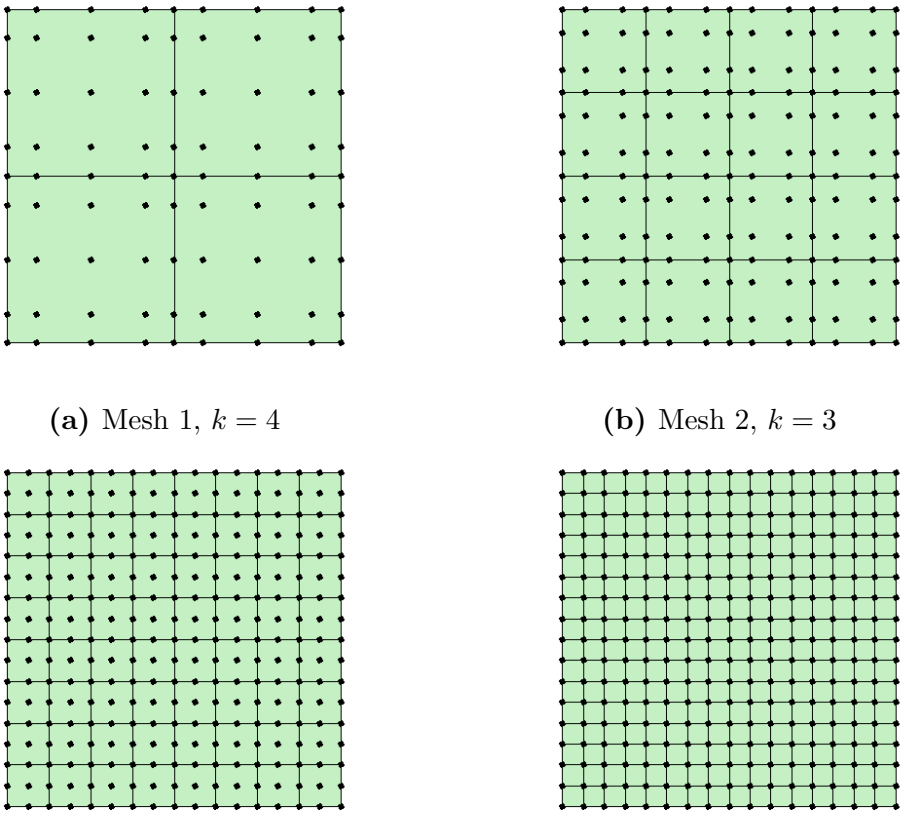


(a) $||e_p||_{\mathcal{L}_2}$ **(b)** $||e_p||_{\mathcal{H}^1}$

(c) $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$ **(d)** $||e_{\boldsymbol{u}}||_{\mathcal{H}^1}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{H}^1}$

Figure 5.4: h-convergence for different degrees of approximation k for 2D Stokes problem for meshes with triangular elements using Neumann formulation of HDG.

5.1.1.2 Meshes with quadrilateral elements



(a) Mesh 1, $k = 4$ **(b)** Mesh 2, $k = 3$

(c) Mesh 3, $k = 2$ **(d)** Mesh 4, $k = 1$

Figure 5.5: Two dimensional quadrilateral meshes.

Seven computational meshes are considered with characteristic element size, h, equal to 1/2, 1/4, 1/8, 1/16, 1/32, 1/64 and 1/128 respectively and the order of the approximation, k, in each mesh, is increased from $k = 1$ to $k = 4$. Figure 5.5 shows the first four meshes with the nodal distribution corresponding to a degree of approximation $k = 4$, $k = 3$, $k = 2$ and $k = 1$. The total number of elements and global degrees of freedom in each mesh and for different degree of approximation is summarized in Table 5.4.

Figure 5.6 shows the numerical solution for velocity and pressure computed with quadratic and cubic approximation, $k = 2$ and $k = 3$, in the first and second of the meshes shown in Figure 5.5. The postprocessed solution is also represented in each case. In this case as well it can be clearly observed that postprocessing offers a significant improvement in the solution.

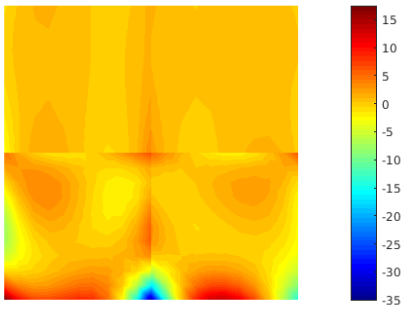
Table 5.4: Total number of elements and global degrees of freedom in each mesh with quadrilateral and for different degree of approximation

h	$\mathtt{n_{el}}$	$\mathtt{n_{dof}}$ k			
		1	2	3	4
1/2	4	25	33	41	49
1/4	16	129	177	225	273
1/8	64	577	801	1025	1249
1/16	256	2433	3393	4353	5313
1/32	1024	9985	13953	17921	21889
1/64	4096	40449	56577	72705	88833
1/128	16384	162817	227841	292865	357889

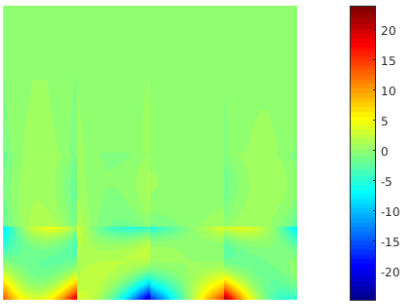
An h-convergence study is performed to validate the implementation. The five meshes described above are considered and the relative error of the numerical solution $\boldsymbol{u}_h$ and the postprocessed solution $\boldsymbol{u}_h^\star$ in the $\mathcal{L}_2(\Omega)$ norm and $\mathcal{H}^1(\Omega)$ norm are evaluated in each case.

Figure 5.7 shows the error of the numerical solution and the postprocessed solution as a function of the element size in logarithmic scale for a degree of approximation from $k = 1$ up to $k = 4$, for velocity, postprocessed velocity (5.7c and 5.7d) and pressure (5.7a and 5.7b) in both the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms. The convergence rates are reported in Table 5.5. From the figure and the table it can be seen that, in all cases, the error in computation is asymptotically approaching the optimal rates of convergence for the numerical solution $\boldsymbol{u}_h$ and pressure p_h, namely $k+1$ and k, respectively in the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms. The table also shows that the error in the computation of the postprocessed solution $\boldsymbol{u}_h^\star$ also asymptotically approaches the optimal rate of convergence, namely $k+2$ and $k+1$ respectively in the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms.

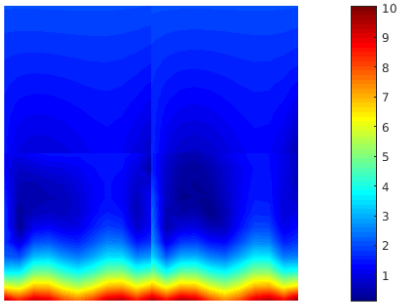
Finally Figure 5.8 shows the error of the numerical solution and the postprocessed solution obtaned using the Neumann formulation of the HDG method as a function of the element size in logarithmic scale for a degree of approximation from $k = 1$ up to $k = 4$, for velocity, postprocessed velocity (5.8c and 5.8d) and pressure (5.8a and 5.8b) in both the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms. In this case too the convergence rates are reported in Table 5.6 showing that, in all cases, the error in the computation of the numerical solution $\boldsymbol{u}_h$ and pressure p_h asymptotically approaches the optimal rate of convergence, namely $k+1$ in the $\mathcal{L}_2(\Omega)$ norm and k in the $\mathcal{H}^1(\Omega)$ norm, while that of the postprocessed solution, $\boldsymbol{u}_h^\star$, approaches its optimal rate of convergence of $k+2$ in the $\mathcal{L}_2(\Omega)$ norm and $k+1$ in the $\mathcal{H}^1(\Omega)$ norm.



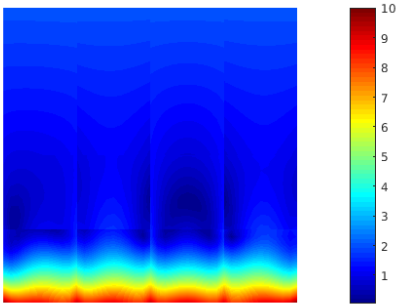
(a) Pressure, Mesh 1, $k = 3$



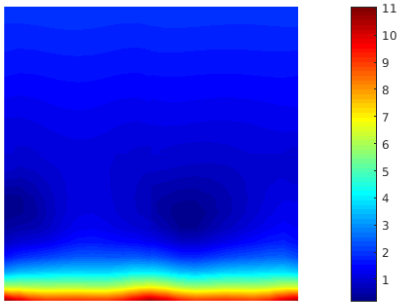
(b) Pressure, Mesh 2, $k = 2$



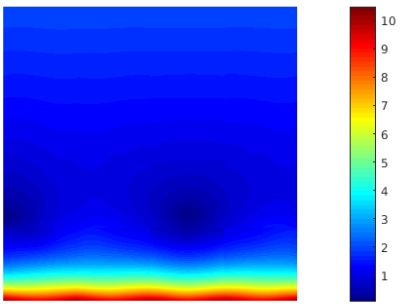
(c) Velocity, Mesh 1, $k = 3$



(d) Velocity, Mesh 2, $k = 2$

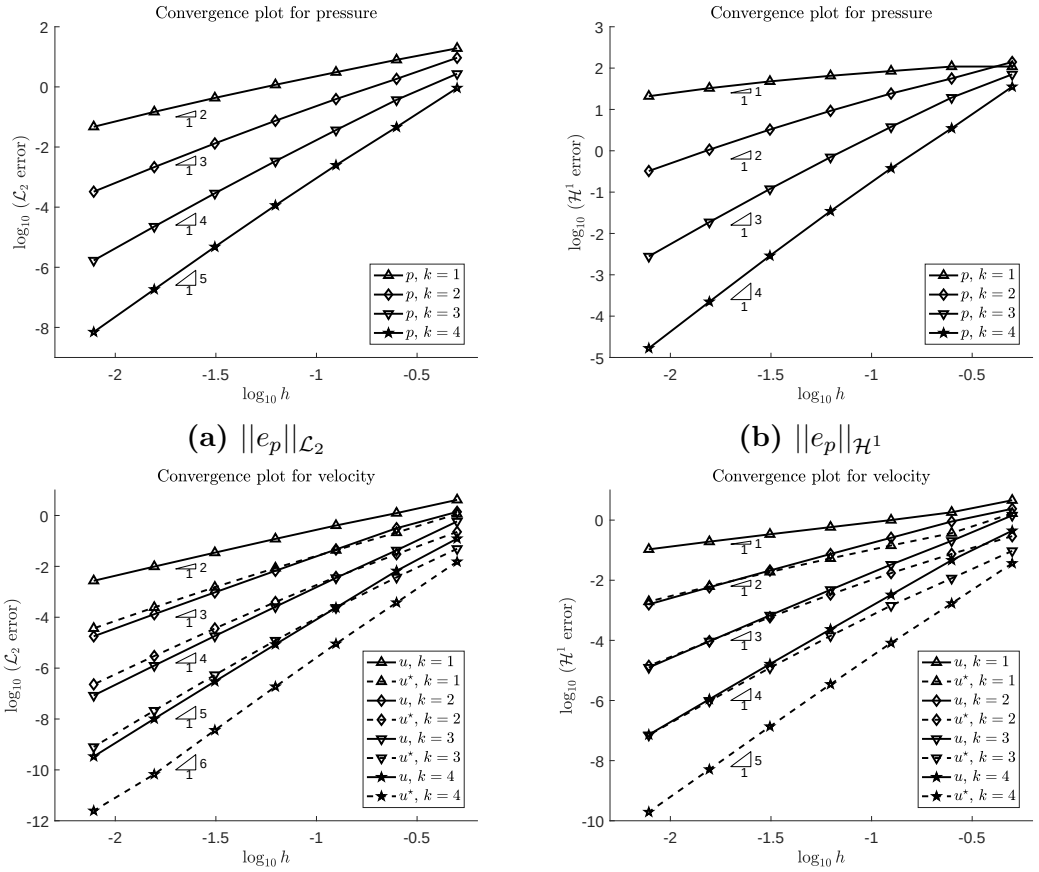


(e) Velocity postprocessed, Mesh 1, $k = 3$



(f) Velocity postprocessed, Mesh 2, $k = 2$

Figure 5.6: Numerical solution for Stokes problem in 2D obtained with two different meshes with quadrilateral elements.



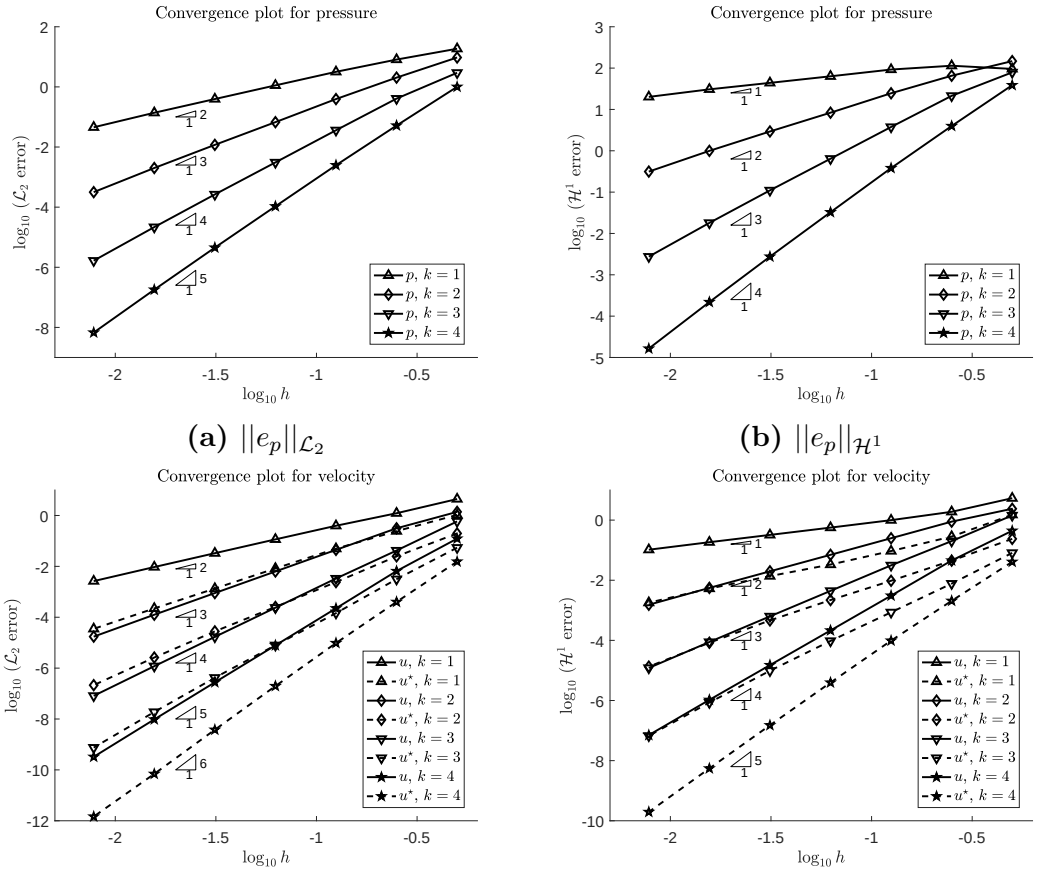
(a) $||e_p||_{\mathcal{L}_2}$ **(b)** $||e_p||_{\mathcal{H}^1}$

(c) $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$ **(d)** $||e_{\boldsymbol{u}}||_{\mathcal{H}^1}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{H}^1}$

Figure 5.7: h-convergence for different degrees of approximation k for 2D Stokes problem for meshes with quadrilateral elements using classical formulation of HDG.

Table 5.5: Rates of convergence for the numerical solution of velocity and pressure and postprocessed solution for 2D Stokes problem for meshes with quadrilateral elements using classical formulation of HDG.

k	$\|\|e\|\|_{\mathcal{L}_2}$			$\|\|e\|\|_{\mathcal{H}^1}$		
	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p
1	1.9	2.7	1.6	0.8	1.7	0.6
2	2.9	3.8	2.7	1.9	2.7	1.7
3	3.9	4.8	3.7	2.9	3.8	2.7
4	4.9	5.8	4.7	3.9	4.8	3.7



(a) $||e_p||_{\mathcal{L}_2}$ **(b)** $||e_p||_{\mathcal{H}^1}$

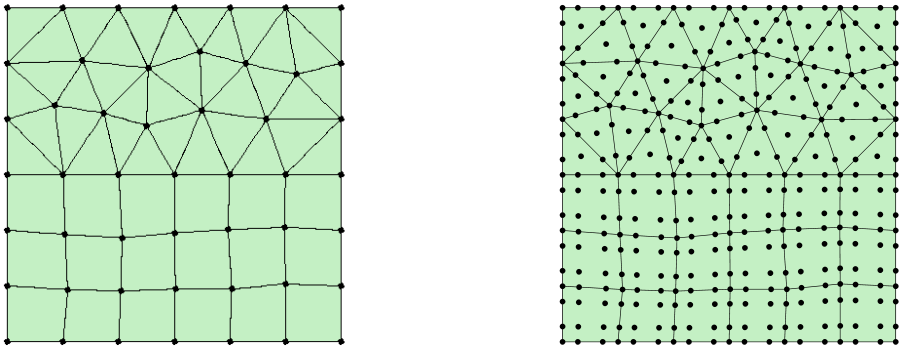
(c) $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$ **(d)** $||e_{\boldsymbol{u}}||_{\mathcal{H}^1}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{H}^1}$

Figure 5.8: h-convergence for different degrees of approximation k for 2D Stokes problem for meshes with quadrilateral elements using Neumann formulation of HDG.

Table 5.6: Rates of convergence for the numerical solution of velocity and pressure and postprocessed solution for 2D Stokes problem for meshes with quadrilateral elements using Neumann formulation of HDG.

k	$\|\|e\|\|_{\mathcal{L}_2}$			$\|\|e\|\|_{\mathcal{H}^1}$		
	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p
1	1.9	2.7	1.6	0.8	1.7	0.6
2	2.9	3.8	2.7	1.9	2.7	1.7
3	3.9	4.8	3.7	2.9	3.8	2.7
4	4.9	5.8	4.7	3.9	4.8	3.7

5.1.1.3 Hybrid mesh



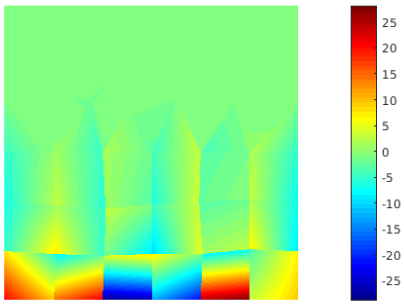
(a) Mesh with $k = 1$ **(b)** Mesh with $k = 3$

Figure 5.9: Two dimensional hybrid meshes.

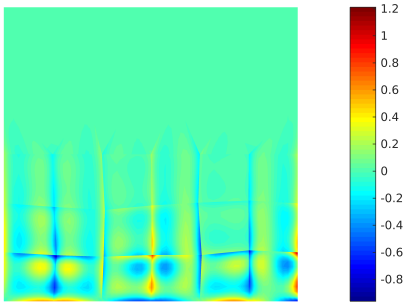
A computational mesh comprising of both quadrilateral and triangular elements is considered in which the order of the approximation, k, is increased from $k = 1$ to $k = 6$. Figure 5.9 shows the two of the meshes with the nodal distribution corresponding to a degree of approximation $k = 1$ and $k = 3$.

Figure 5.10 shows the numerical solution for velocity and pressure computed with linear and cubic approximation, $k = 1$ and $k = 3$, in the hybrid mesh shown in Figure 5.9. The postprocessed solution is also represented in each case. Even with in the case of the hybrid mesh it can be clearly observed that postprocessing offers a significant improvement in the solution.

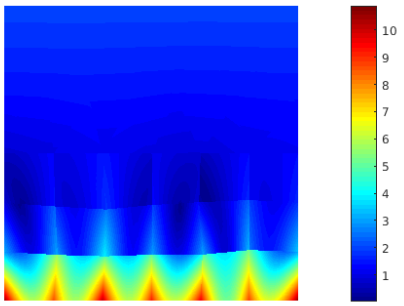
An h-convergence study is not performed here as the implementation of the code has been validated for meshes with only triangles and only quadrilateral elements, as shown in subsections 5.1.1.1 and 5.1.1.2. The solution obtained with these meshes are visually inspected and they show great agreement with the analytical solution.



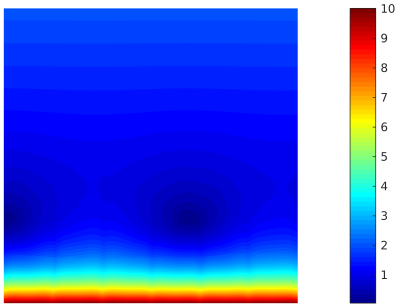
(a) Pressure, $k = 1$



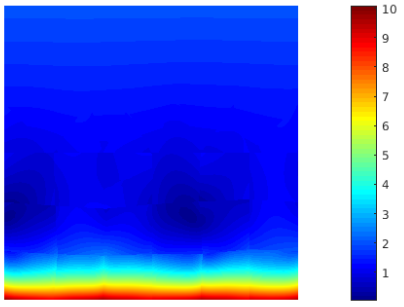
(b) Pressure, $k = 3$



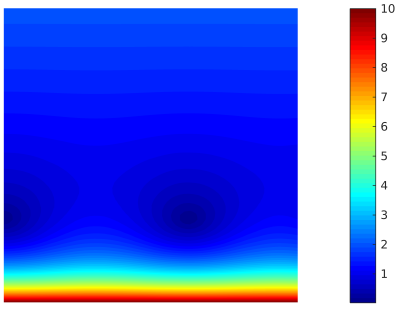
(c) Velocity, $k = 1$



(d) Velocity, $k = 3$



(e) Velocity postprocessed, $k = 1$



(f) Velocity postprocessed, $k = 3$

Figure 5.10: Numerical solution for Stokes problem in 2D obtained with two different degrees of approximation in a mesh with both triangular and quadrilateral elements.

5.1.1.4 Comparison of triangular and quadrilateral meshes

From the discussion in subsection 5.1.1.1 and 5.1.1.2, it could be seen that in both the cases, i.e, meshes with triangular and quadrilateral elements, the error in the computation of the variables was asymptotically converging at the optimal rate of convergence. Also it is worth noting that in the case of meshes with triangles with only four mesh refinements, the optimal rates of convergence were seen. However, in the case of quadrilateral meshes, six levels of mesh refinement were not sufficient to see the error converge at the optimal convergence rate. At face value, this seems to be indicating that triangular meshes perform better than meshes with quadrilateral elements. This is, however, not completely true.

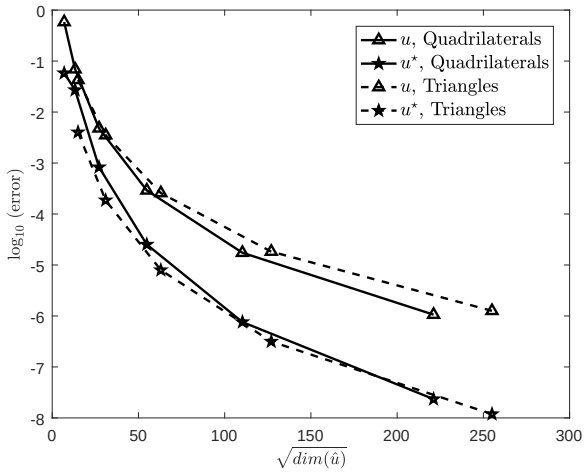


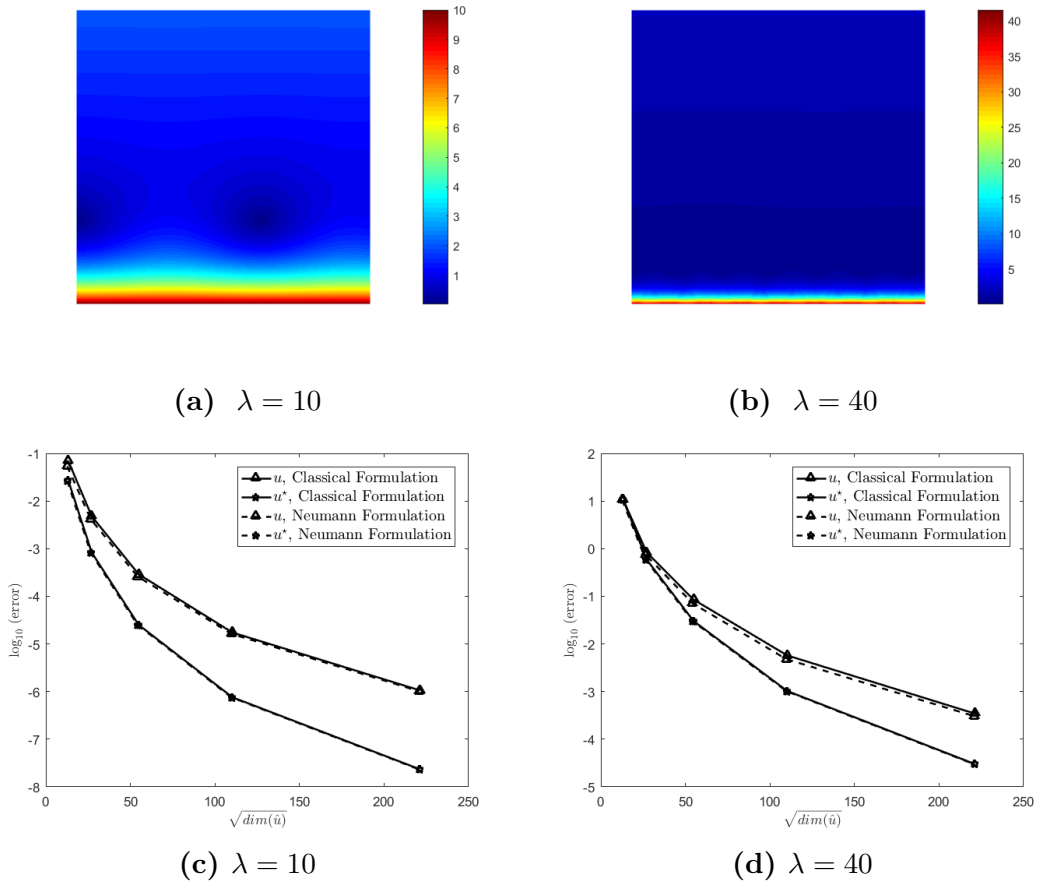
Figure 5.11: Comparison of triangular and quadrilateral meshes: error of $\boldsymbol{u}_h$ and $\boldsymbol{u}_h^\star$ in the $\mathcal{L}_2(\Omega)$ norm as a function of $\sqrt{\dim(\hat{\boldsymbol{u}})}$ for the HDG formulation with triangular and quadrilateral meshes

.

Figure 5.11 shows the evolution of the error of $\boldsymbol{u}_h$ and $\boldsymbol{u}_h^\star$ in the $\mathcal{L}_2(\Omega)$ norm as a function of the square root of the number of degrees of freedom of the global system of equations, i.e. $\mathtt{n_{dof}} = \dim(\hat{\boldsymbol{u}})$. From this plot it can be clearly seen that to compute a solution with a given error, quadrilateral meshes require less degrees of freedom in the global system. This is key in terms of computational time as a smaller global system ensures lesser computational time to solve the system of equations. Thus,in the case of meshes with quadrilateral elements, though the error in the computation of the variables seems to approach the optimal rate of convergence with more levels of mesh refinement than in the case of meshes with triangular elements, it appears that due to the smaller size of global system, quadrilateral meshes are bound to be computationally cheaper than triangular

meshes when the solution has to be computed with a certain level of accuracy.

5.1.1.5 Comparison of classical and Neumann formulations of HDG



(a) $\lambda = 10$ **(b)** $\lambda = 40$

(c) $\lambda = 10$ **(d)** $\lambda = 40$

Figure 5.12: Comparison of classical and Neumann formulations: Analytical solutions (top) and error of $\boldsymbol{u}_h$ and $\boldsymbol{u}_h^\star$ in the $\mathcal{L}_2(\Omega)$ norm as a function of $\sqrt{\dim(\hat{\boldsymbol{u}})}$ (bottom), for different values of λ

Here, a h-convergence study is performed in order to check the optimality of the approximation using the formulation with Neumann local problems and to compare the accuracy of the two HDG formulations considered in this work. Figure 5.12 shows the evolution of the error of $\boldsymbol{u}_h$ and $\boldsymbol{u}_h^\star$ in the $\mathcal{L}_2$ norm as a function of the square root of the number of degrees of freedom of the global system of equations. Two different cases are considered. In these cases the value of coefficient λ appearing in the analytical solution is changed from 10 to 40 to produce a very

thin boundary layer. The degree of approximation is increased in each mesh from $p = 1$ to $p = 4$.

From Figure 5.12 it can be seen that there is a very small improvement in the solution using the Neumann formulation which can be seen in the case when $\lambda = 40$. However, in both the cases there is not much difference in the case of the postprocessed solution. Finally the Neumann formulation comes with slightly lower degrees of freedom of the global variable. It is important to stress that the differences between the formulation with Dirichlet and Neumann local problems are noticed even if a global measure of the error is employed. Obviously, the extra accuracy provided by the formulation with Neumann problems is expected to be more relevant if the output of interest is defined near the Neumann boundary or on the Neumann boundary.

5.1.2 The 3-D case

In order to illustrate the results of HDG for Stokes equations for a 3-D case, the model problem in (3.1) is solved in $\Omega :=]0,1[\times]0,1[\times]0,1[$ with $\Gamma_N = \{(x,y,z) \in \partial\Omega | y = 0\}$ and $\Gamma_D = \partial\Omega \backslash \Gamma_N$. The source and boundary conditions are taken such that the analytical solution is given by

$$\boldsymbol{u} = \begin{pmatrix} b\exp(a(x-z)+b(y-z)) - a\exp(a(z-y)+b(x-y)) \\ b\exp(a(y-x)+b(z-x)) - a\exp(a(x-z)+b(y-z)) \\ b\exp(a(z-y)+b(x-y)) - a\exp(a(y-x)+b(z-x)) \end{pmatrix}$$

and

$$p = \sin(xyz).$$

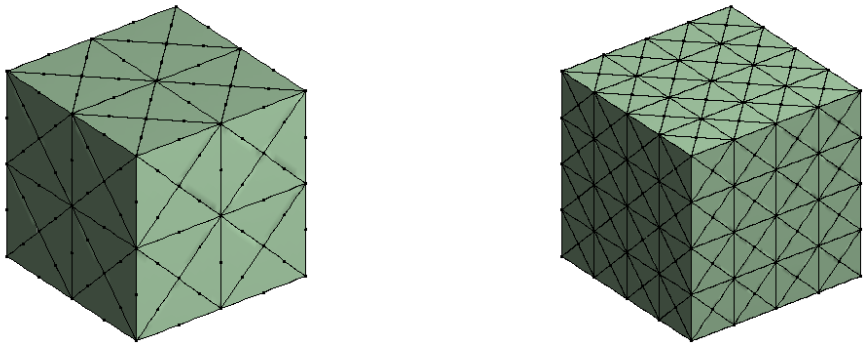
This test case, proposed by [57], is used to test the performance of the HDG for a three dimensional flow. The coefficients appearing in the analytical expression of the velocity are set to $a = 1$ and $b = 0.5$. The problem is solved in meshes with tetrahedral elements, hexahedral elements and those containing a mixture of four kinds of elements, tetrahedrons, hexahedrons, prisms and pyramids.

5.1.2.1 Meshes with tetrahedral elements

Three computational meshes are considered with characteristic element size, h, equal to 1, 1/2 and 1/4 respectively and the order of the approximation, k, in each mesh, is increased from $k = 1$ to $k = 3$. Figure 5.13 shows the second and the third meshes with the nodal distribution corresponding to a degree of approximation $k = 2$ and $k = 1$.

Figure 5.15 shows the numerical solution for velocity and pressure computed with quadratic and cubic approximation, $k = 2$ and $k = 3$, in the meshes shown in

Figure 5.13. The postprocessed solution is also represented in each case. In this case the solution and the postprocessed solution appear to be almost similar and it seems that postprocessing does not offer a significant improvement in the solution. This is because of the nature of the problem chosen which is very simple and hence is being captured very well by a coarse mesh with low degree of approximation.



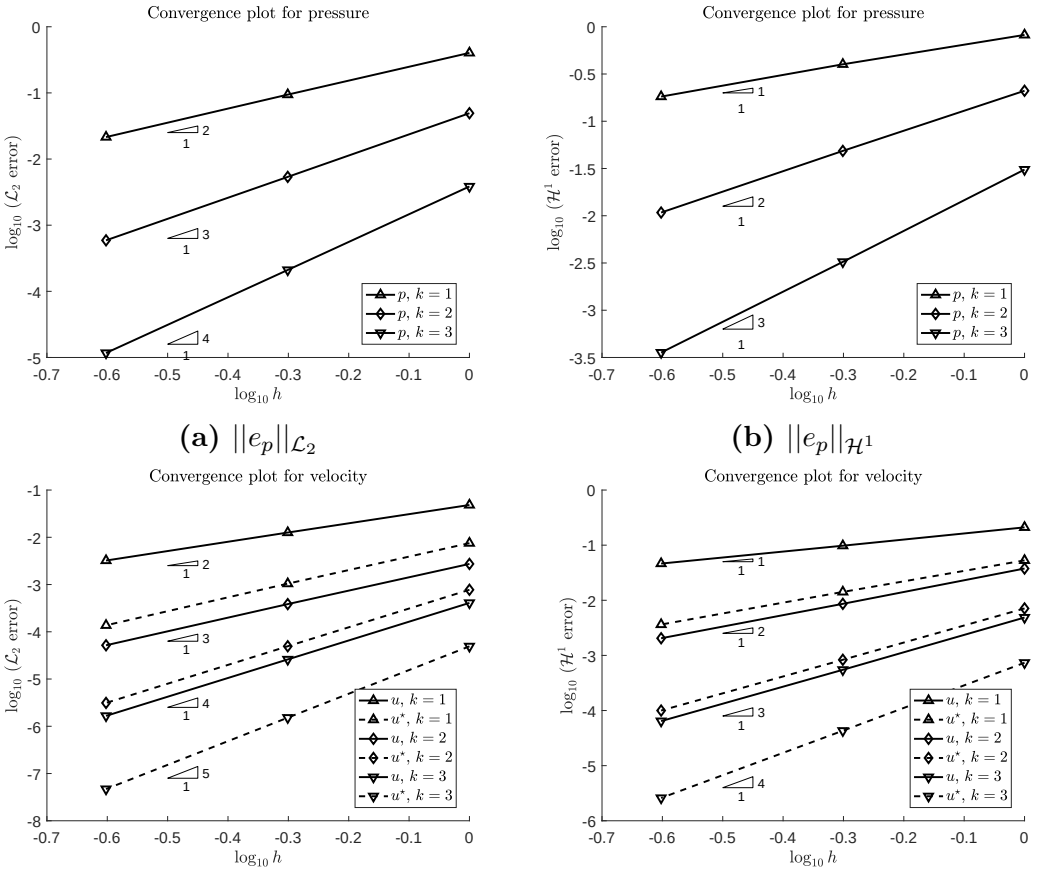
(a) Mesh 2, $k = 2$ **(b)** Mesh 3, $k = 1$

Figure 5.13: Three dimensional tetrahedral meshes.

An h-convergence study is performed to validate the implementation. The three meshes described above are considered and the relative error of the numerical solution $\boldsymbol{u}_h$ and the postprocessed solution $\boldsymbol{u}_h^\star$ in the $\mathcal{L}_2(\Omega)$ norm and $\mathcal{H}^1(\Omega)$ norm are evaluated in each case.

Figure 5.14 shows the error of the numerical solution and the postprocessed solution as a function of the element size in logarithmic scale for a degree of approximation from $k = 1$ up to $k = 3$, for velocity, postprocessed velocity (5.14c and 5.14d) and pressure (5.14a and 5.14b) in both the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms. The convergence rates are reported in Table 5.7 showing that, in all cases, the optimal rate of convergence for the numerical solution $\boldsymbol{u}_h$ and pressure p_h is obtained, namely $k + 1$ and k, respectively, in the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms. The table also shows that optimal rate of convergence is obtained for the postprocessed solution $\boldsymbol{u}_h^\star$, namely $k + 2$ and $k + 1$ respectively, in the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms.

Finally Figure 5.16 shows the error of the numerical solution and the postprocessed solution obtaned using the Neumann formulation of the HDG method as a function of the element size in logarithmic scale for a degree of approximation from $k = 1$ up to $k = 3$, for velocity, postprocessed velocity (5.16c and 5.16d) and pressure (5.16a and 5.16b) in both the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms. In this case too the convergence rates are reported in Table 5.8 showing that, in all cases, the optimal rate of convergence for the numerical solution $\boldsymbol{u}_h$ and pressure p_h is ob-

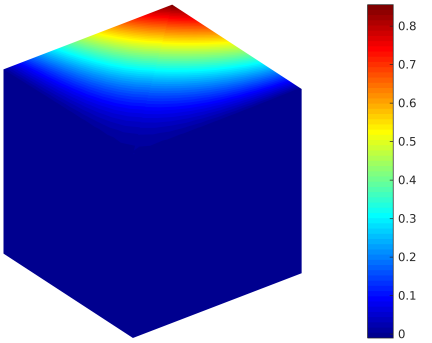


(a) $||e_p||_{\mathcal{L}_2}$ **(b)** $||e_p||_{\mathcal{H}^1}$

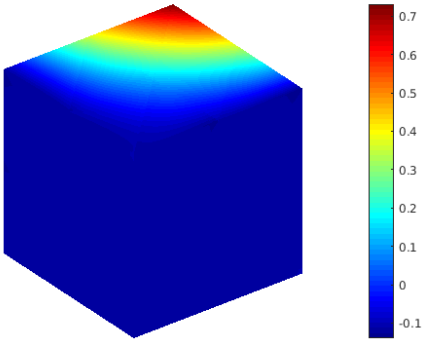
(c) $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$ **(d)** $||e_{\boldsymbol{u}}||_{\mathcal{H}^1}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{H}^1}$

Figure 5.14: h-convergence for different degrees of approximation k for 3D Stokes problem for meshes with tetrahedral elements using classical formulation of HDG.

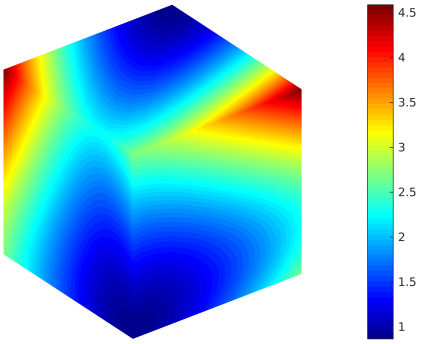
tained, namely $k+1$ and k in the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms and the optimal rate of convergence of $k+2$ and $k+1$ is obtained for the postprocessed solution $\boldsymbol{u}_h^\star$, respectively in the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms.



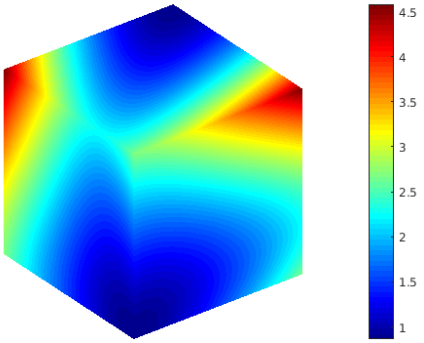
(a) Pressure, Mesh 1, $k = 3$



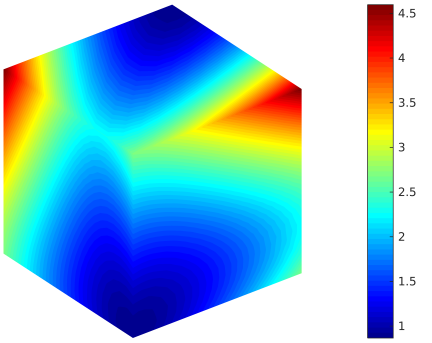
(b) Pressure, Mesh 2, $k = 2$



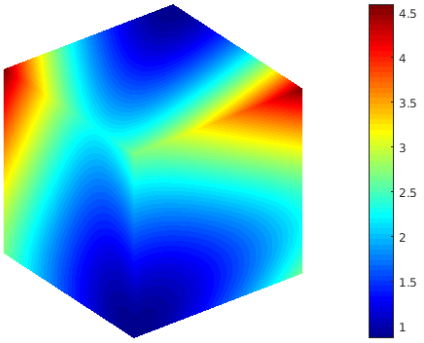
(c) Velocity, Mesh 1, $k = 3$



(d) Velocity, Mesh 2, $k = 2$



(e) Velocity postprocessed, Mesh 1, $k = 3$

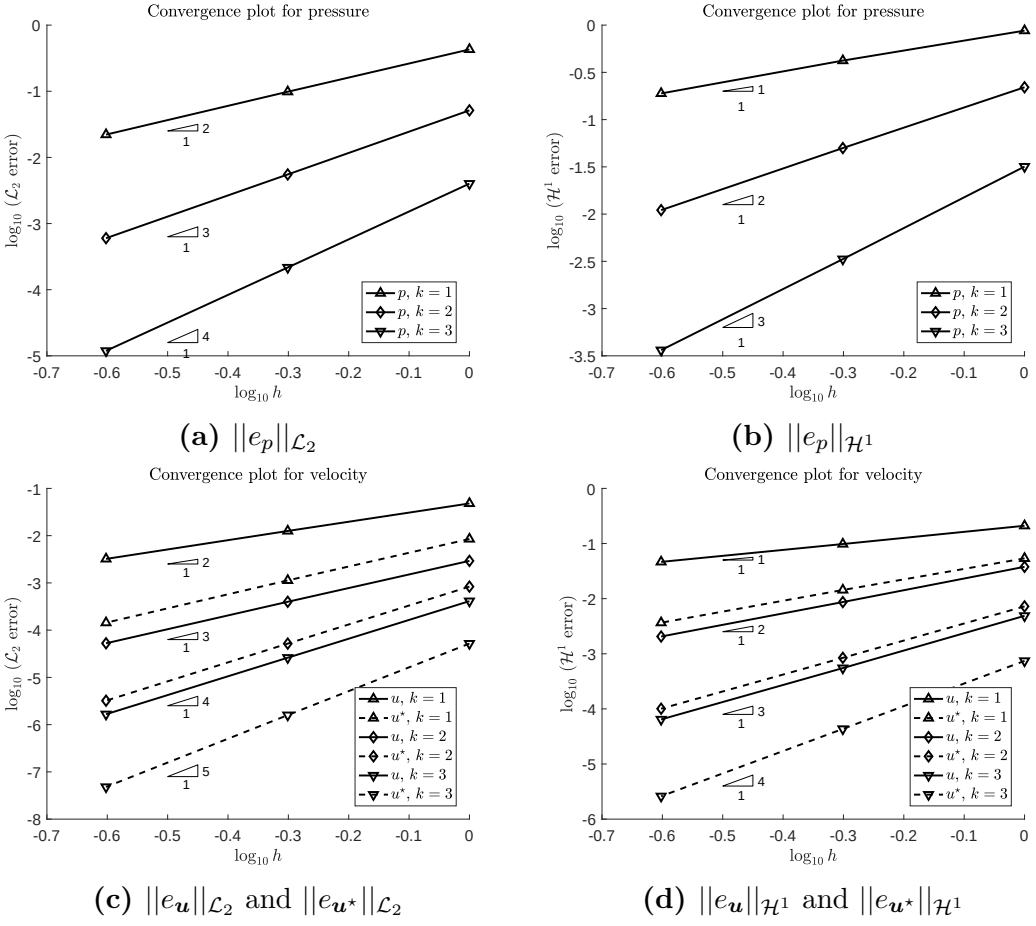


(f) Velocity postprocessed, Mesh 2, $k = 2$

Figure 5.15: Numerical solution for Stokes problem in 3D obtained with two different meshes with tetrahedral elements.

Table 5.7: Rates of convergence for the numerical solution of velocity and pressure and postprocessed solution for 3D Stokes problem for meshes with tetrahedral elements using classical formulation of HDG.

k	$\|\|e\|\|_{\mathcal{L}_2}$			$\|\|e\|\|_{\mathcal{H}^1}$		
	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p
1	2.0	3.0	2.1	1.1	2.0	1.2
2	2.9	4.0	3.2	2.1	3.1	2.2
3	4.0	5.0	4.2	3.1	4.1	3.2



(a) $||e_p||_{\mathcal{L}_2}$ **(b)** $||e_p||_{\mathcal{H}^1}$

(c) $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$ **(d)** $||e_{\boldsymbol{u}}||_{\mathcal{H}^1}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{H}^1}$

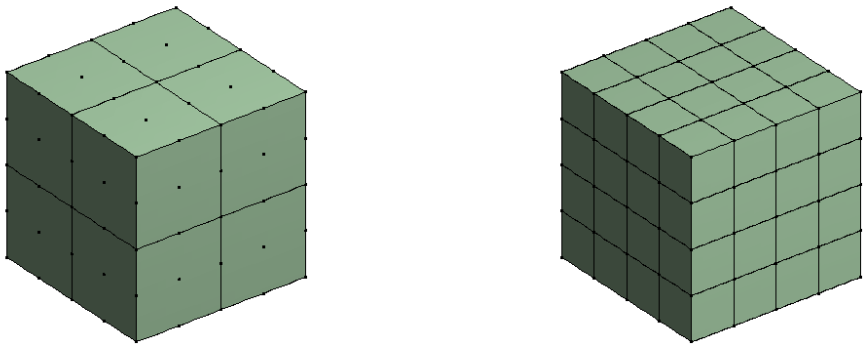
Figure 5.16: h-convergence for different degrees of approximation k for 3D Stokes problem for meshes with tetrahedral elements using Neumann formulation of HDG.

Table 5.8: Rates of convergence for the numerical solution of velocity and pressure and postprocessed solution for 3D Stokes problem for meshes with tetrahedral elements using Neumann formulation of HDG.

k	$\|\|e\|\|_{\mathcal{L}_2}$			$\|\|e\|\|_{\mathcal{H}^1}$		
	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p
1	2.0	3.0	2.1	1.1	2.0	1.2
2	2.9	4.0	3.2	2.1	3.1	2.2
3	4.0	5.0	4.2	3.1	4.1	3.2

5.1.2.2 Meshes with hexahedral elements

Four computational meshes are considered with characteristic element size, h, equal to 1, 1/2, 1/4 and 1/8 respectively and the order of the approximation, k, in each mesh, is increased from $k = 1$ to $k = 3$. Figure 5.17 shows the second and the third meshes with the nodal distribution corresponding to a degree of approximation $k = 2$ and $k = 1$.



(a) Mesh 2, $k = 2$ **(b)** Mesh 3, $k = 1$

Figure 5.17: Three dimensional hexahedral meshes.

Figure 5.18 shows the numerical solution for velocity and pressure computed with quadratic and cubic approximation, $k = 2$ and $k = 3$, in the meshes shown in Figure 5.17. The postprocessed solution is also represented in each case. Similar to the results presented in the previous section with tetrahedral elements, in this case too, the solution and the postprocessed solution appear to be almost similar and it seems that postprocessing does not offer a significant improvement in the solution. As stated earlier this is because of the simple nature of the problem chosen which is being captured very well by a coarse mesh with low degree of approximation.

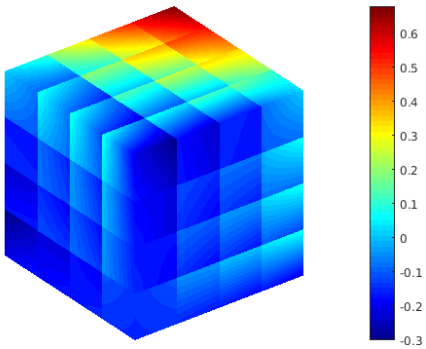
An h-convergence study is performed to validate the implementation. The four meshes described above are considered and the relative error of the numerical solution $\boldsymbol{u}_h$ and the postprocessed solution $\boldsymbol{u}_h^\star$ in the $\mathcal{L}_2(\Omega)$ norm and $\mathcal{H}^1(\Omega)$ norm are evaluated in each case.

Figure 5.19 shows the error of the numerical solution and the postprocessed solution as a function of the element size in logarithmic scale for a degree of approximation from $k = 1$ up to $k = 3$, for velocity, postprocessed velocity (5.19c and 5.19d) and pressure (5.19a and 5.19b) in both the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms. The convergence rates are reported in Table 5.9. From the figure and the table it can be seen that, in all cases, the error in computation is asymptotically approaching the optimal rates of convergence for the numerical solution $\boldsymbol{u}_h$ and pressure p_h, namely $k+1$ and k, respectively in the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms. The table also shows that the error in the computation of the postprocessed solution $\boldsymbol{u}_h^\star$ also asymptotically approaches the optimal rate of convergence, namely $k+2$ and $k+1$ respectively in the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms.

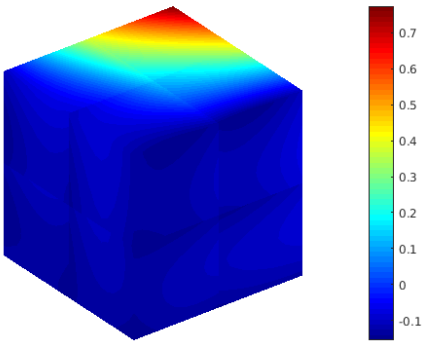
Finally Figure 5.20 shows the error of the numerical solution and the postprocessed solution obtaned using the Neumann formulation of the HDG method as a function of the element size in logarithmic scale for a degree of approximation from $k = 1$ up to $k = 3$, for velocity, postprocessed velocity (5.20c and 5.20d) and pressure (5.20a and 5.20b) in both the $\mathcal{L}_2(\Omega)$ and $\mathcal{H}^1(\Omega)$ norms. In this case too the convergence rates are reported in Table 5.10 showing that, in all cases, the error in the computation of the numerical solution $\boldsymbol{u}_h$ and pressure p_h asymptotically approaches the optimal rate of convergence, namely $k+1$ in the $\mathcal{L}_2(\Omega)$ norm and k in the $\mathcal{H}^1(\Omega)$ norm, while that of the postprocessed solution, $\boldsymbol{u}_h^\star$, approaches its optimal rate of convergence of $k+2$ in the $\mathcal{L}_2(\Omega)$ norm and $k+1$ in the $\mathcal{H}^1(\Omega)$ norm.

Table 5.9: Rates of convergence for the numerical solution of velocity and pressure and postprocessed solution for 3D Stokes problem for meshes with hexahedral elements using classical formulation of HDG.

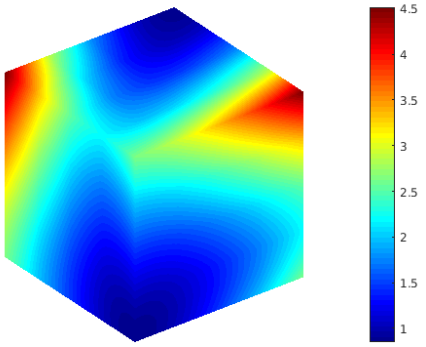
k	$\|\|e\|\|_{\mathcal{L}_2}$			$\|\|e\|\|_{\mathcal{H}^1}$		
	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p
1	1.6	2.5	1.5	0.8	1.5	0.6
2	2.7	3.7	2.6	1.8	2.6	1.6
3	3.7	4.7	3.6	2.8	3.7	2.6



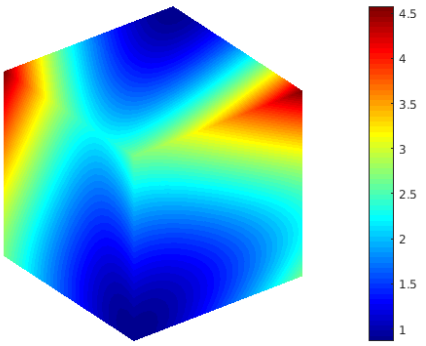
(a) Pressure, Mesh 3, $k = 1$



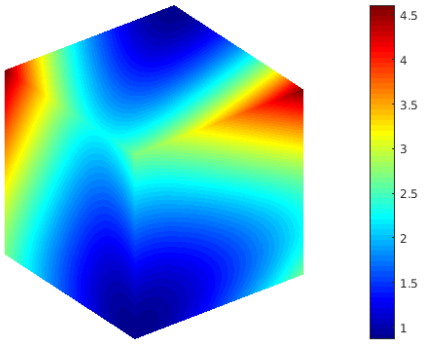
(b) Pressure, Mesh 2, $k = 2$



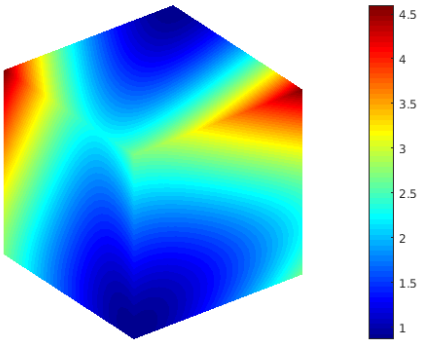
(c) Velocity, Mesh 3, $k = 1$



(d) Velocity, Mesh 2, $k = 2$

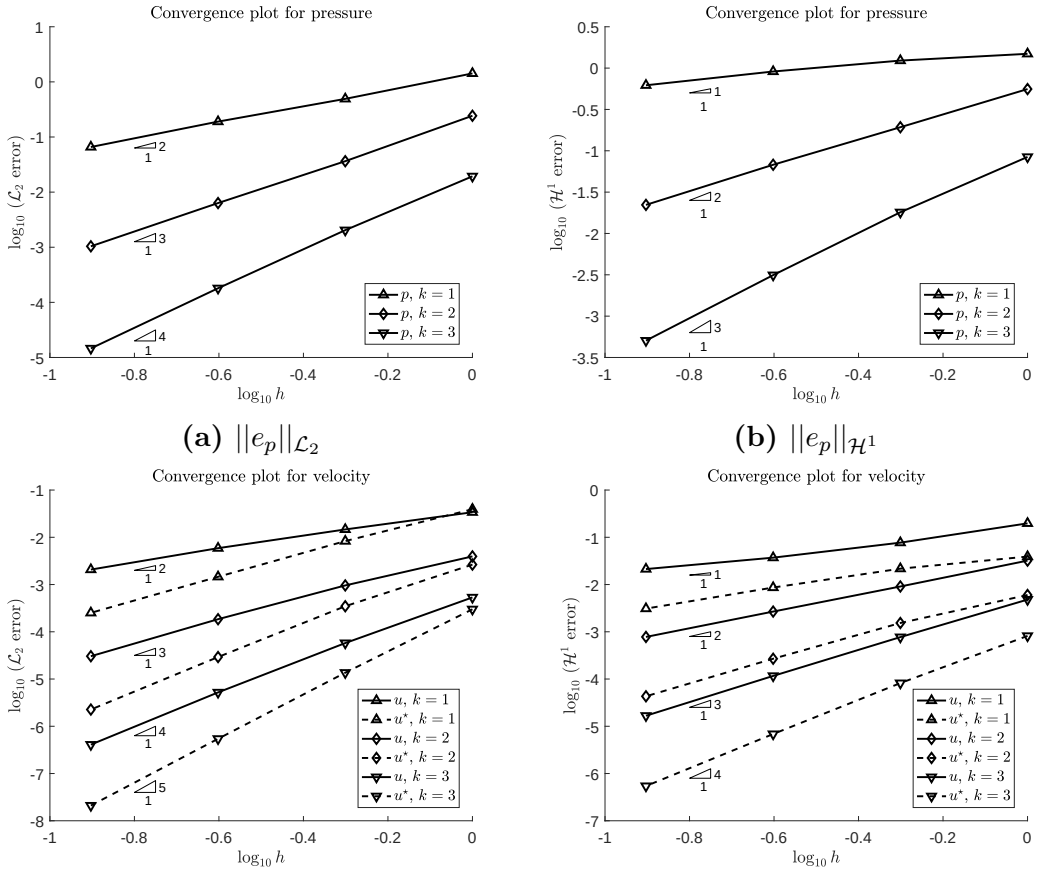


(e) Velocity postprocessed, Mesh 3, $k = 1$



(f) Velocity postprocessed, Mesh 2, $k = 2$

Figure 5.18: Numerical solution for Stokes problem in 3D obtained with two different meshes with hexahedral elements.



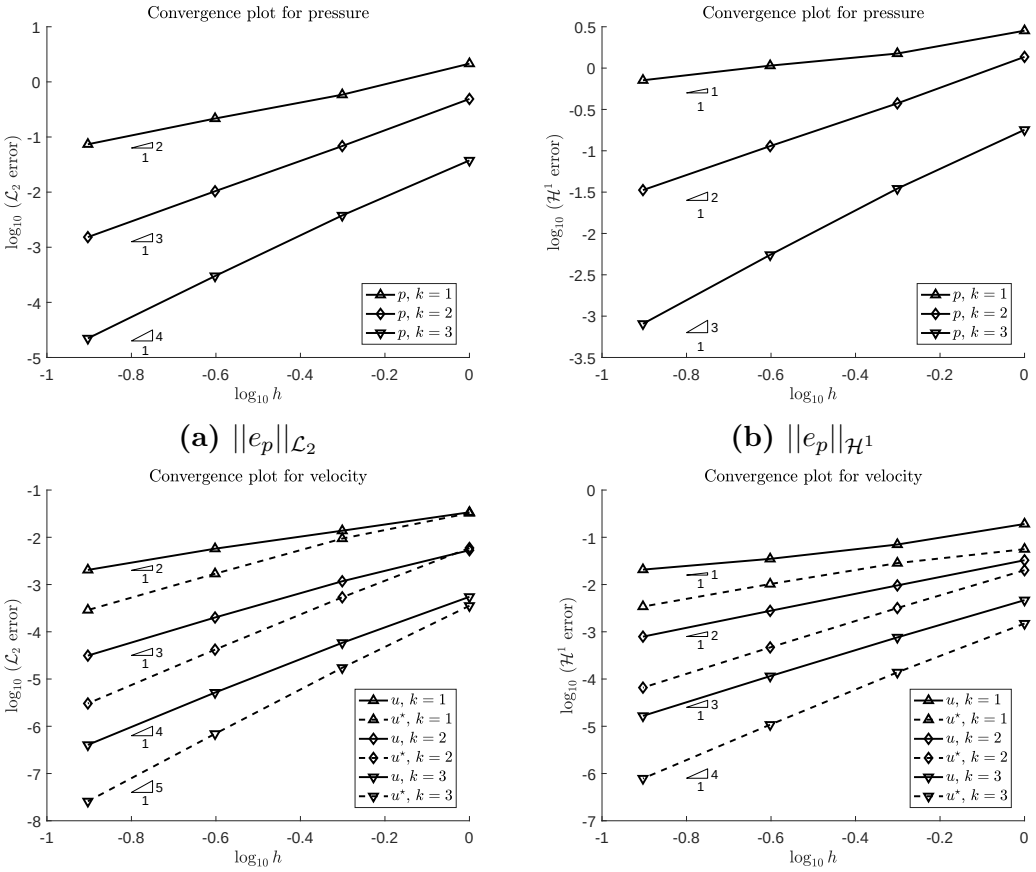
(a) $||e_p||_{\mathcal{L}_2}$ **(b)** $||e_p||_{\mathcal{H}^1}$

(c) $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$ **(d)** $||e_{\boldsymbol{u}}||_{\mathcal{H}^1}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{H}^1}$

Figure 5.19: h-convergence for different degrees of approximation k for 3D Stokes problem for meshes with hexahedral elements using classical formulation of HDG.

Table 5.10: Rates of convergence for the numerical solution of velocity and pressure and postprocessed solution for 3D Stokes problem for meshes with hexahedral elements using Neumann formulation of HDG.

k	$\|\|e\|\|_{\mathcal{L}_2}$			$\|\|e\|\|_{\mathcal{H}^1}$		
	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p
1	1.6	2.5	1.5	0.8	1.5	0.6
2	2.7	3.7	2.6	1.8	2.6	1.6
3	3.7	4.7	3.6	2.8	3.7	2.6



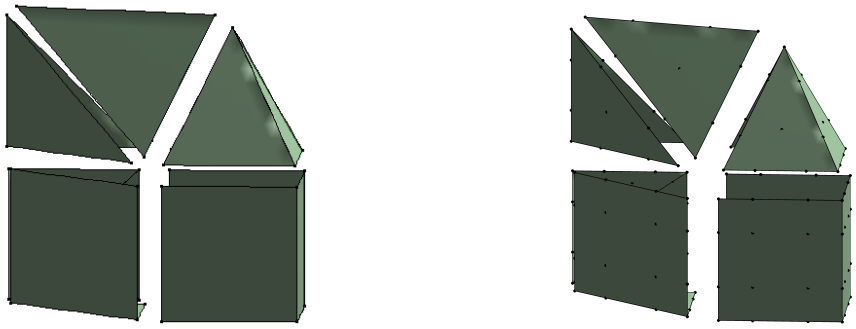
(a) $||e_p||_{\mathcal{L}_2}$ **(b)** $||e_p||_{\mathcal{H}^1}$

(c) $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$ **(d)** $||e_{\boldsymbol{u}}||_{\mathcal{H}^1}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{H}^1}$

Figure 5.20: h-convergence for different degrees of approximation k for 3D Stokes problem for meshes with hexahedral elements using Neumann formulation of HDG.

5.1.2.3 Hybrid mesh

Similar to the case in 2D, a hybrid computational mesh comprising of tetrahedra, hexahedral, prismatic and pyramid elements is considered in which the order of the approximation, k, is increased from $k = 1$ to $k = 4$. Figure 5.21 shows two of the meshes with the nodal distribution corresponding to a degree of approximation $k = 1$ and $k = 3$.



(a) Mesh with $k = 1$ **(b)** Mesh with $k = 3$

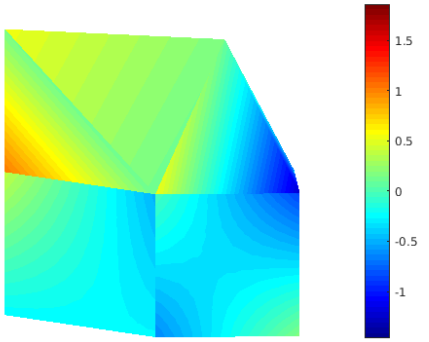
Figure 5.21: Three dimensional hybrid meshes.

Figure 5.22 shows the numerical solution for velocity and pressure computed with linear and cubic approximation, $k = 1$ and $k = 3$, in the hybrid mesh shown in Figure 5.21. Again, since the problem is very simple with a very smooth solution field, the postprocessed solution is represented in each case does not offer a significant improvement in the solution.

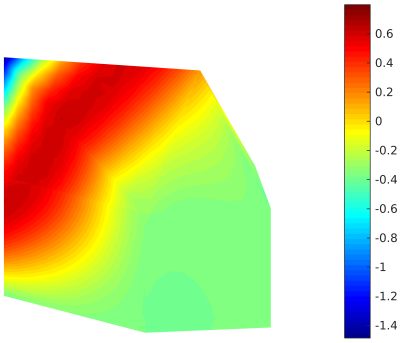
An h-convergence study is not performed here as the implementation of the code has been validated for meshes with only tetrahedral and only hexahedral elements, as shown in subsections 5.1.2.1 and 5.1.2.2. The solution obtained with these meshes are visually inspected and they show great agreement with the analytical solution.

5.1.2.4 Comparison of tetrahedral and hexahedral meshes

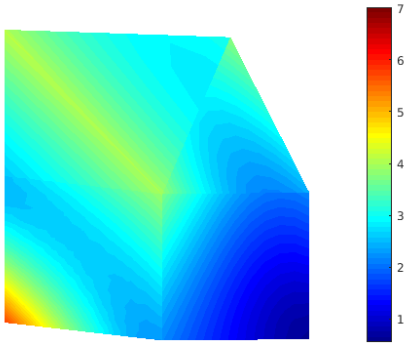
From the discussion in subsection 5.1.2.1 and 5.1.2.2, it could be seen that in both the cases, i.e, meshes with tetrahedral and hexahedral elements, the error in the computation of the variables was asymptotically converging at the optimal rate of convergence. Also it is worth noting that in the case of meshes with tetrahedrons, with only two mesh refinements, the optimal rates of convergence were seen. However, in the case of hexahedral meshes, three levels of mesh refinement were not sufficient to see the error converge at the optimal convergence rate. Again, this seems to be indicating that tetrahedral meshes perform better than meshes with hexahedral elements. This is, however, not completely true.



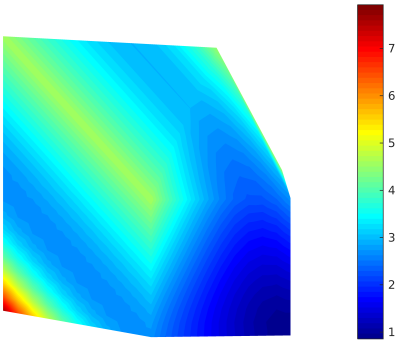
(a) Pressure, $k = 1$



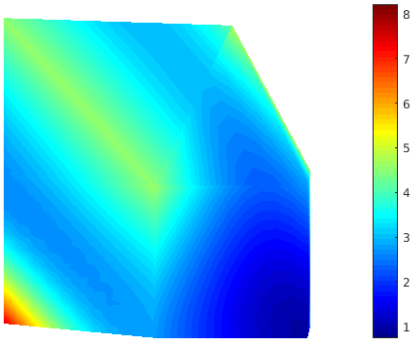
(b) Pressure, $k = 3$



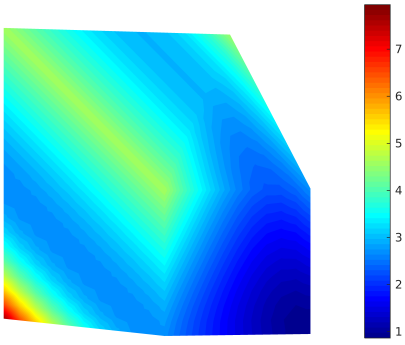
(c) Velocity, $k = 1$



(d) Velocity, $k = 3$



(e) Velocity postprocessed, $k = 1$



(f) Velocity postprocessed, $k = 3$

Figure 5.22: Numerical solution for Stokes problem in 3D obtained with two different degrees of approximation in a mesh with tetrahedral, hexahedral, prismatic and pyramid elements.

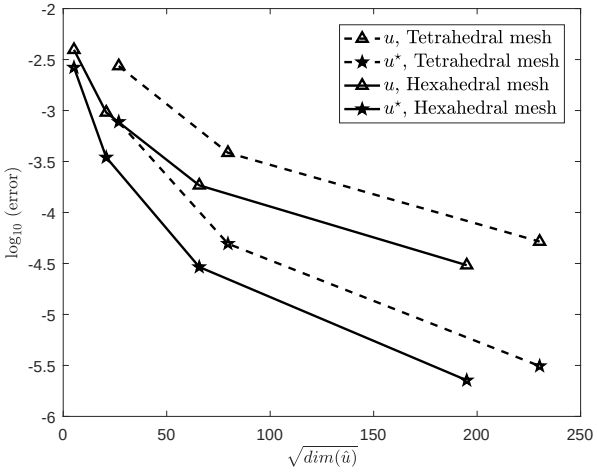


Figure 5.23: Comparison of tetrahedral and hexahedral meshes: error of $\boldsymbol{u}_h$ and $\boldsymbol{u}_h^\star$ in the $\mathcal{L}_2(\Omega)$ norm as a function of $\sqrt{\dim(\hat{\boldsymbol{u}})}$ for the HDG formulation with tetrahedral and hexahedral meshes

.

Figure 5.23 shows the evolution of the error of $\boldsymbol{u}_h$ and $\boldsymbol{u}_h^\star$ in the $\mathcal{L}_2(\Omega)$ norm as a function of the square root of the number of degrees of freedom of the global system of equations, i.e. $\mathtt{n_{dof}} = \dim(\hat{\boldsymbol{u}})$. From this plot it can be clearly seen that to compute a solution with a given error, tetrahedral meshes require less degrees of freedom in the global system. This is key in terms of computational time as a smaller global system ensures lesser computational time to solve the system of equations. Thus, in the case of meshes with hexahedral elements, though the error in the computation of the variables seems to approach the optimal rate of convergence with more levels of mesh refinement than in the case of meshes with tetrahedral elements, it appears that due to the smaller size of global system, hexahedral meshes are bound to be computationally cheaper than hexahedral meshes when the solution has to be computed with a certain level of accuracy.

5.2 Validation of HDG for Stokes equations using meshes with non uniform degree of approximation

These examples are performed in order to validate the code written to solve Stokes' equations using the HDG method using meshes with varying degree of approximation within the elements. Validation of the code is done using convergence plots for both the 2-D and 3-D cases. In this case validation is done by ensuring that

the errors in the computation of the variables converge at optimal rates both when seen from a global point of view and from a local element-wise point of view.

5.2.1 The 2-D case

In order to illustrate the results of HDG for Stokes equations for a 2-D case with varying degree of approximation in the elements, the model problem in (3.1) is solved in $\Omega :=]0,1[\times]0,1[$ with $\Gamma_N = \{(x,y) \in \partial\Omega | y = 0\}$ and $\Gamma_D = \partial\Omega \backslash \Gamma_N$. The source and boundary conditions are taken such that the analytical solution is given by

$$\boldsymbol{u} = \begin{pmatrix} 2ay - b\lambda \exp(-\lambda y)\cos(\lambda x) \\ b\lambda \exp(-\lambda y)\sin(\lambda x) \end{pmatrix}$$

and

$$p = 0.$$

The coefficients appearing in the analytical expression of the velocity are set to $a = 1$, $b = 1$ and $\lambda = 10$.

Five computational meshes are considered with characteristic element size, h, equal to $1/2\sqrt{2}$, $1/4\sqrt{2}$, $1/8\sqrt{2}$, $1/16\sqrt{2}$ and $1/32\sqrt{2}$ respectively and the order of the approximation, k, in each mesh, is non uniform. Figure 5.24a shows the distribution of the degree of approximation, k, in the different regions of the domain. Figures 5.24b, 5.24c and 5.24d show three levels of refinement of the mesh with coloured elements being the element of interest for element-wise h-convergence study.

An a priori estimate for an elliptic problem stands [58]

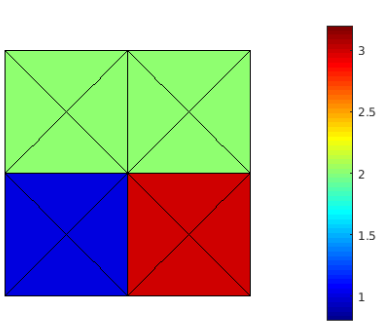
$$||e||_{\mathcal{L}_2} \leq \alpha h^{k_{\min}+1}, \tag{5.3}$$

where $k_{\min}$ is related with the minimum degree of the complete polynomials included in the interpolation. The constant α is independent of the element size h but it is unknown. This represents a global measure, where h is a characteristic element size. A local a priori error estimate stands [55]

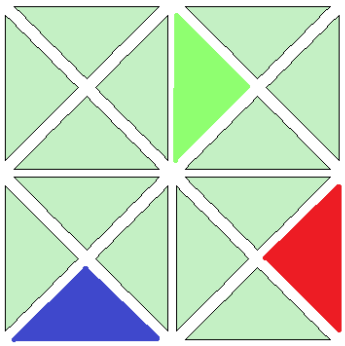
$$||e_i||_{\mathcal{L}_2} \leq \alpha h_i^{k_i+1+\mathtt{n_{sd}}/2}, \tag{5.4}$$

where h_i and k_i, respectively, are the characteristic size of and degree of approximation in the element Ω_i and $\mathtt{n_{sd}}$ is the dimension of the problem (two for bidimensional problems, three for tridimensional ones).

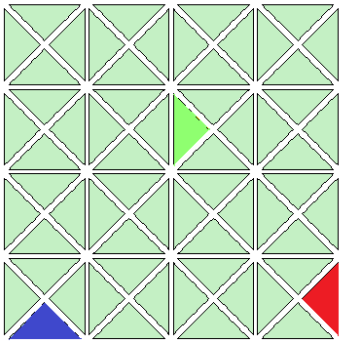
Figure 5.25 shows both the local and global error of the numerical solution and the postprocessed solution as a function of the element size in logarithmic scale



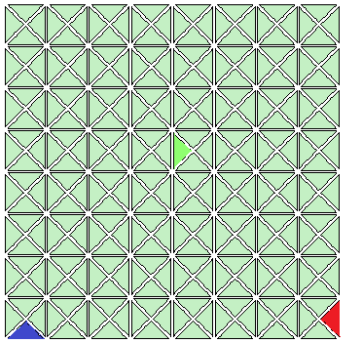
(a) k in different regions of the domain



(b) Mesh 1



(c) Mesh 2



(d) Mesh 3

Figure 5.24: Two dimensional meshes with variable degrees of approximation, $k = 1, 2, 3$ within the elements. Coloured elements are the elements of interest considered for element-wise h-convergence study with the colours blue, green and red respectively representing $k = 1$, $k = 2$ and $k = 3$

Table 5.11: Rates of convergence for the numerical solution of velocity and pressure and postprocessed solution for 2D Stokes problem for meshes with non uniform degree of approximation in the elements.

Type of error estimate	k	$\|\|e_{\boldsymbol{u}}\|\|_{\mathcal{L}_2}$	$\|\|e_{\boldsymbol{u}^\star}\|\|_{\mathcal{L}_2}$	$\|\|e_p\|\|_{\mathcal{L}_2}$
Global	1,2,3	2.0	2.9	2.0
	2,3,4	3.0	4.1	3.0
	3,4,5	4.0	5.1	4.0
Local	1	2.9	4.0	2.8
	2	4.1	4.9	3.9
	3	4.8	5.8	4.7

for varying degrees of approximation. The element-wise error estimation is shown for elements with degrees of approximation, $k = 1, 2, 3$. The convergence rates are reported in Table 5.11 showing that, in all cases, the optimal rate of convergence for the numerical solution $\boldsymbol{u}_h$ and pressure p_h is obtained, namely $k_{\min} + 1$ in the $\mathcal{L}_2(\Omega)$ norm for the global error, while $k_i + 1 + {}^{\mathrm{n_{sd}}}/_2$ in the $\mathcal{L}_2(\Omega)$ norm for the local element-wise error. The table also shows that optimal rate of convergence is obtained for the postprocessed solution $\boldsymbol{u}_h^\star$, namely $k_{\min} + 2$ in the $\mathcal{L}_2(\Omega)$ norm for the global error, while $k_i + 2 + {}^{\mathrm{n_{sd}}}/_2$ in the $\mathcal{L}_2(\Omega)$ norm for the local element-wise error.

5.2.2 The 3-D case

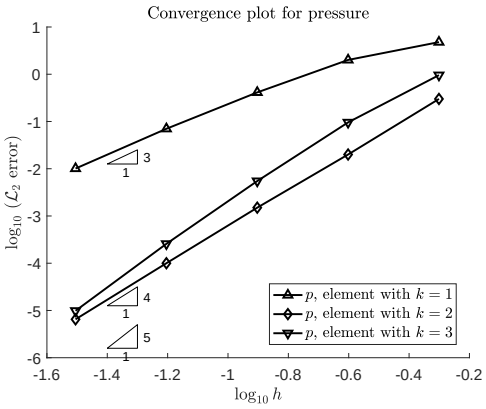
In order to illustrate the results of HDG for Stokes equations for a 3-D case, the model problem in (3.1) is solved in $\Omega :=]0,1[\times]0,1[\times]0,1[$ with $\Gamma_N = \{(x,y,z) \in \partial\Omega | y = 0\}$ and $\Gamma_D = \partial\Omega \backslash \Gamma_N$. The source and boundary conditions are taken such that the analytical solution is given by

$$\boldsymbol{u} = \begin{pmatrix} b\exp(a(x-z)+b(y-z)) - a\exp(a(z-y)+b(x-y)) \\ b\exp(a(y-x)+b(z-x)) - a\exp(a(x-z)+b(y-z)) \\ b\exp(a(z-y)+b(x-y)) - a\exp(a(y-x)+b(z-x)) \end{pmatrix}$$

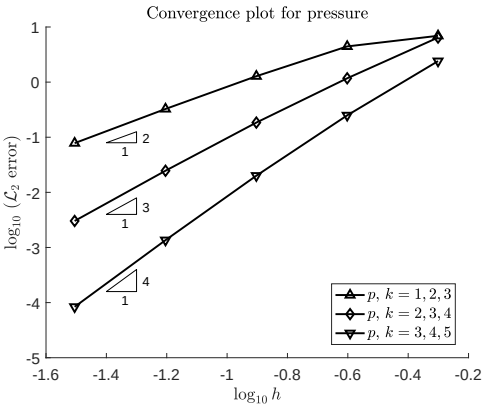
and

$$p = \sin(xyz).$$

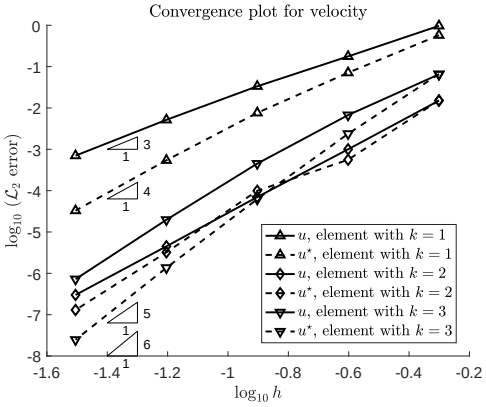
The coefficients appearing in the analytical expression of the velocity are set to $a = 1$ and $b = 0.5$.



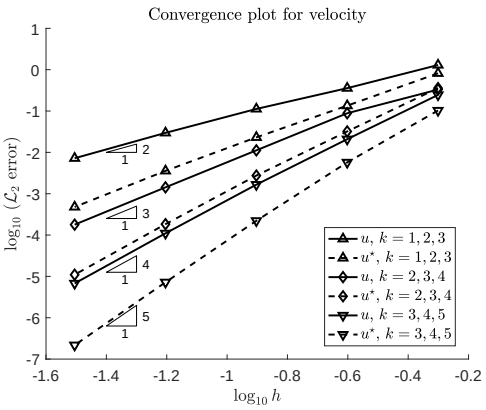
(a) element-wise error $||(e_p)_i||_{\mathcal{L}_2}$



(b) global error $||e_p||_{\mathcal{L}_2}$

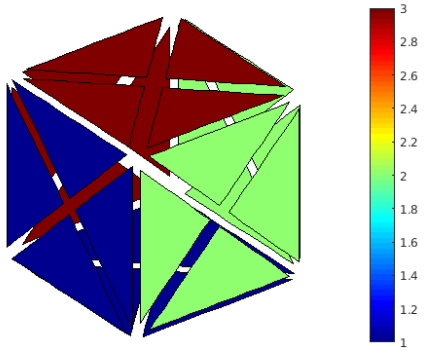


(c) element-wise error $||(e_{\boldsymbol{u}})_i||_{\mathcal{L}_2}$ and $||(e_{\boldsymbol{u}^\star})_i||_{\mathcal{L}_2}$

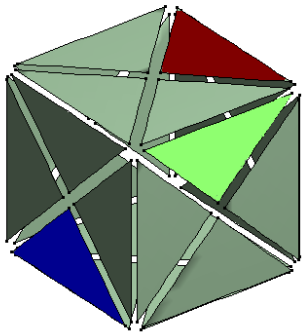


(d) global error $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$

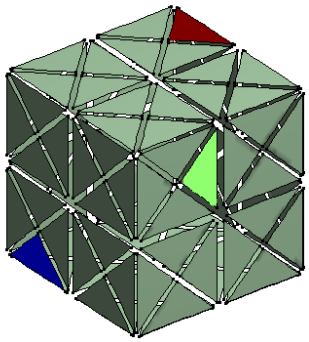
Figure 5.25: h-convergence for different combinations of non uniform degrees of approximation k in the elements of the meshes for 2D Stokes problem.



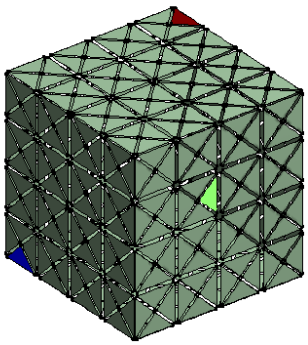
(a) k in different regions of the domain



(b) Mesh 1



(c) Mesh 2



(d) Mesh 3

Figure 5.26: Three dimensional meshes with variable degrees of approximation, $k = 1, 2, 3$ within the elements. Coloured elements are the elements of interest considered for element-wise h-convergence study with the colours blue, green and red respectively representing $k = 1$, $k = 2$ and $k = 3$

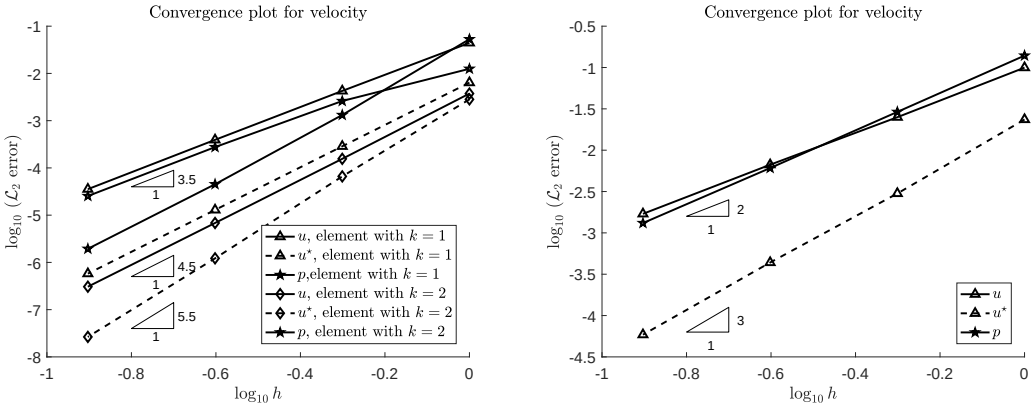
Four computational meshes are considered with characteristic element size, h, equal to 1, 1/2, 1/4 and 1/8, respectively and the order of the approximation, k, in each mesh, is non uniform. Figure 5.26a shows the distribution of the degree of approximation, k, in the different regions of the domain. Figures 5.26b, 5.26c and 5.26d show three levels of refinement of the mesh with coloured elements being the element of interest for element-wise h-convergence study.

Figure 5.27 shows both the local and global error of the numerical solution and the postprocessed solution as a function of the element size in logarithmic scale for varying degrees of approximation. The element-wise error estimation is shown for elements with degrees of approximation, $k = 1, 2, 3$. The convergence rates are reported in Table 5.12 showing that, in all cases, the optimal rate of convergence

for the numerical solution $\boldsymbol{u}_h$ and pressure p_h is obtained, namely $k_{\min}+1$ in the $\mathcal{L}_2(\Omega)$ norm for the global error, while $k_i+1+{}^{\mathtt{n_{sd}}}/_2$ in the $\mathcal{L}_2(\Omega)$ norm for the local element-wise error. The table also shows that optimal rate of convergence is obtained for the postprocessed solution $\boldsymbol{u}_h^\star$, namely $k_{\min}+2$ in the $\mathcal{L}_2(\Omega)$ norm for the global error, while $k_i+2+{}^{\mathtt{n_{sd}}}/_2$ in the $\mathcal{L}_2(\Omega)$ norm for the local element-wise error.

Table 5.12: Rates of convergence for the numerical solution of velocity and pressure and postprocessed solution for 3D Stokes problem for meshes with non uniform degree of approximation in the elements.

Type of error estimate	k	$\|\|e_{\boldsymbol{u}}\|\|_{\mathcal{L}_2}$	$\|\|e_{\boldsymbol{u}^\star}\|\|_{\mathcal{L}_2}$	$\|\|e_p\|\|_{\mathcal{L}_2}$
	1,2,3	2.0	2.9	2.2
Global	1,2,3	3.0	3.9	3.1
	1	3.47	4.48	3.46
Local	2	4.48	5.52	4.47
	3	5.50	6.51	5.48



(a) element-wise error $||(e_p)_i||_{\mathcal{L}_2}$, $||(e_{\boldsymbol{u}})_i||_{\mathcal{L}_2}$ and $||(e_{\boldsymbol{u}^\star})_i||_{\mathcal{L}_2}$

(b) global error $||e_p||_{\mathcal{L}_2}$, $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$

Figure 5.27: h-convergence for different combinations of non uniform degrees of approximation k in the elements of the meshes for 3D Stokes problem.

5.3 Validation of NEFEM-HDG for Stokes equations

This example is performed in order to validate the code written to solve Stokes' equations using the NEFEM-HDG method. Validation of the code is done using convergence plots using a problem that has an analytical solution. The workability of the code is shown for both the classical and Neumann formulations of HDG

In order to illustrate the results of NEFEM-HDG for Stokes equations for a 2-D case, the model problem in (3.1) is solved in a domain bounded by the lines $x = 0$, $y = 1$ and $x = 1$ on the left right and top and by the curve $y = \big(1 + \cos(5\pi x)\big)/10$ with $\Gamma_N = \{(x, y) \in \partial\Omega | y = \big(1 + \cos(5\pi x)\big)/10\}$ and $\Gamma_D = \partial\Omega \backslash \Gamma_N$. The source and boundary conditions are taken such that the analytical solution is given by

$$\boldsymbol{u} = \begin{pmatrix} a\big(y - f(x)\big) - b\lambda \exp\Big(-\lambda\big(y - f(x)\big)\Big) \\ \Big(a\big(y - f(x)\big) - b\lambda \exp\Big(-\lambda\big(y - f(x)\big)\Big)\Big) f'(x) \end{pmatrix}$$

with

$$f(x) = \frac{\big(1 + \cos(5\pi x)\big)}{10}; \quad f'(x) = -\frac{\pi}{2}\sin(5\pi x)$$

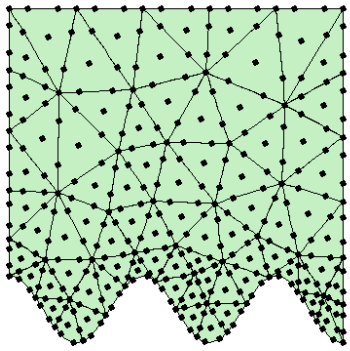
and

$$p = 0.$$

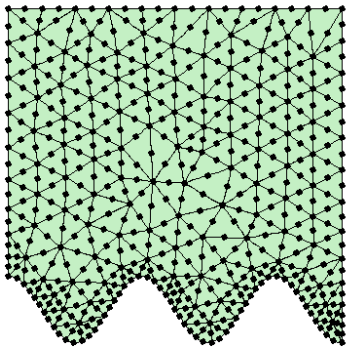
This test case, is a modification of the one proposed by [56], is used to test the performance of the HDG for a two dimensional flow. The modification is done to ensure that the boundary layer follows the curved boundary. The coefficients appearing in the analytical expression of the velocity are set to $a = 1$, $b = 1$ and $\lambda = 10$, leading to large variations of the velocity along the curved boundary. The problem is solved in meshes with triangular elements.

Five computational meshes are considered with characteristic element size, h, equal to 1.45×10^{-1}, 7.32×10^{-2}, 3.69×10^{-2}, 1.83×10^{-2} and $9.20E \times 10^{-3}$ respectively and the order of the approximation, k, in each mesh, is increased from $k = 1$ to $k = 3$. Figure 5.28 shows the first three meshes with the nodal distribution corresponding to a degree of approximation $k = 3$, $k = 2$ and $k = 1$. The total number of elements and global degrees of freedom in each mesh and for different degree of approximation is summarized in Table 5.13. .

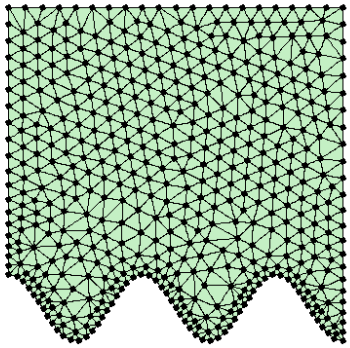
Figure 5.29 shows the numerical solution for velocity and pressure computed with quadratic and cubic approximation, $k = 2$ and $k = 3$, in the first and second of the meshes shown in Figure 5.28. The postprocessed solution is also represented



(a) Mesh 1, $k = 3$



(b) Mesh 2, $k = 2$



(c) Mesh 3, $k = 1$

Figure 5.28: NEFEM-HDG validation: Two dimensional triangular meshes.

Table 5.13: Total number of elements and global degrees of freedom in each mesh with quadrilateral and for different degree of approximation

h	$\mathtt{n_{el}}$	$\mathtt{n_{dof}}$ k		
		1	2	3
1.45E-01	74	533	725	917
7.32E-02	253	1907	2607	3307
3.69E-02	884	6833	9365	11897
1.83E-02	3489	27431	37657	47883
9.20E-03	13609	107907	148251	188595

in each case. It can be clearly observed that postprocessing offers a significant improvement in the solution.

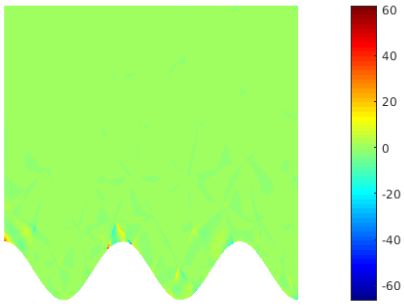
An h-convergence study is performed to validate the implementation. The five meshes described above are considered and the relative error of the numerical solution $\boldsymbol{u}_h$ and the postprocessed solution $\boldsymbol{u}_h^\star$ in the $\mathcal{L}_2(\Omega)$ norm is evaluated in each case.

Figure 5.30 shows the error of the numerical solution and the postprocessed solution as a function of the element size in logarithmic scale for a degree of approximation from $k = 1$ up to $k = 3$, for velocity, postprocessed velocity (5.30b) and pressure (5.30a) in the $\mathcal{L}_2(\Omega)$ norm. The convergence rates are reported in Table 5.14 showing that, in all cases, the optimal rate of convergence for the numerical solution $\boldsymbol{u}_h$ and pressure p_h is obtained, namely $k + 1$ in the $\mathcal{L}_2(\Omega)$ norm. The table also shows that optimal rate of convergence is obtained for the postprocessed solution $\boldsymbol{u}_h^\star$, namely $k + 2$ in the $\mathcal{L}_2(\Omega)$ norm.

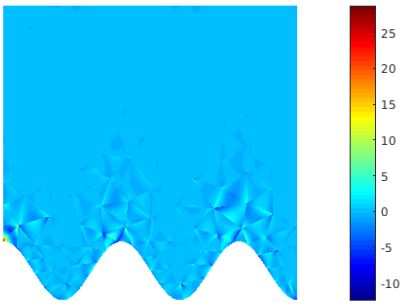
Finally Figure 5.31 shows the error of the numerical solution and the postprocessed solution obtaned using the Neumann formulation of the HDG method as a function of the element size in logarithmic scale for a degree of approximation from $k = 1$ up to $k = 3$, for velocity, postprocessed velocity (5.31b) and pressure (5.31a) in the $\mathcal{L}_2(\Omega)$ norm. In this case too the convergence rates are reported in Table 5.14 showing that, in all cases, the optimal rate of convergence for the numerical solution $\boldsymbol{u}_h$ and pressure p_h is obtained, namely $k+1$ in the $\mathcal{L}_2(\Omega)$ norm, and the optimal rate of convergence of $k+2$ in the $\mathcal{L}_2(\Omega)$ norm is obtained for the postprocessed solution $\boldsymbol{u}_h^\star$.

Table 5.14: Rates of convergence for the numerical solution of velocity and pressure and postprocessed solution in the $\mathcal{L}_2$ norm for 2D Stokes problem for meshes with triangular elements using classical and Neumann formulations of NEFEM-HDG method.

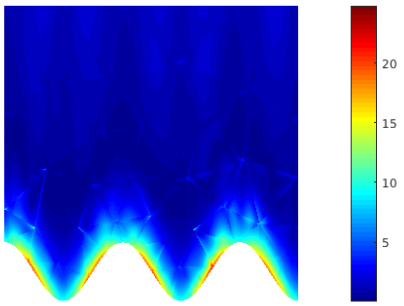
	Formulation					
k	Classical			Neumann		
	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p	$\boldsymbol{u}$	$\boldsymbol{u}^\star$	p
1	2.0	3.1	2.0	2.1	3.1	2.0
2	3.1	4.2	3.0	3.2	4.1	3.0
3	4.1	5.3	4.1	4.1	5.2	4.1



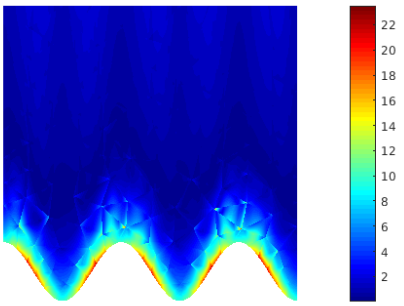
(a) Pressure, Mesh 1, $k = 3$



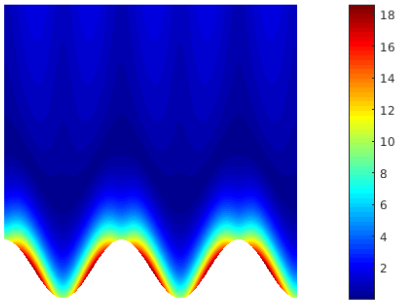
(b) Pressure, Mesh 2, $k = 2$



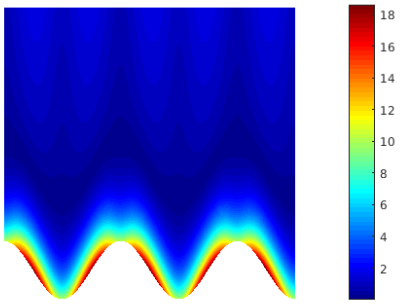
(c) Velocity, Mesh 1, $k = 3$



(d) Velocity, Mesh 2, $k = 2$

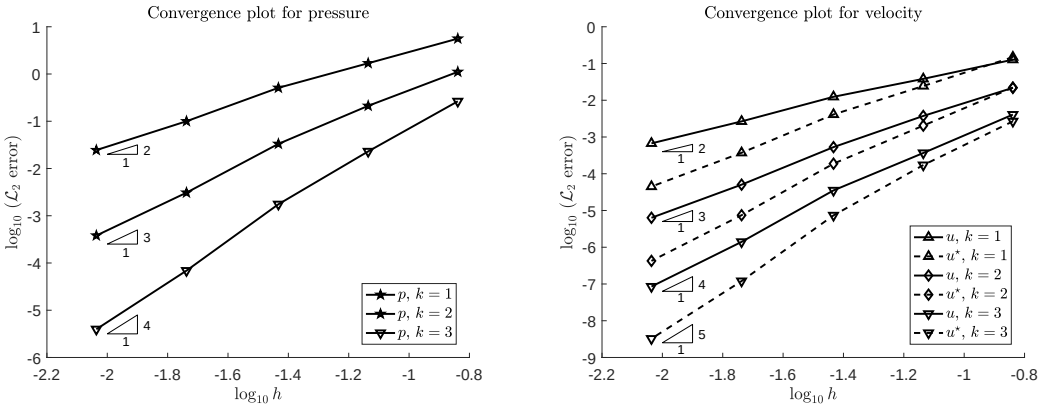


(e) Velocity postprocessed, Mesh 1, $k = 3$



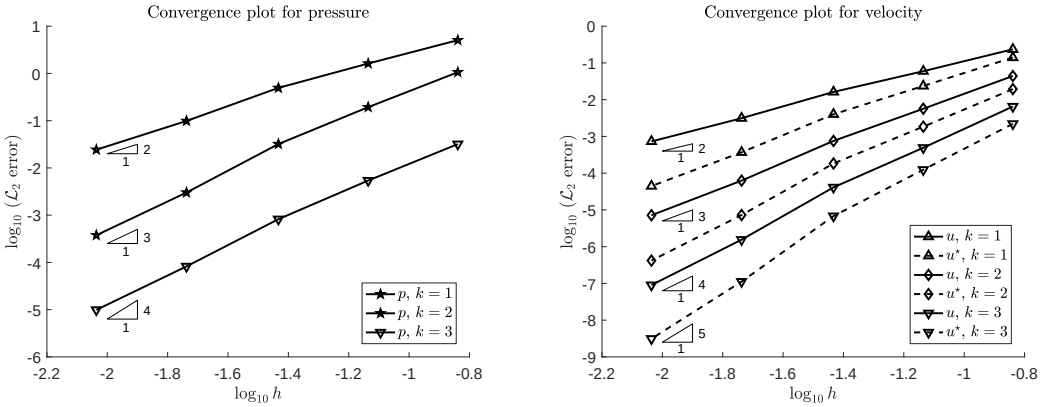
(f) Velocity postprocessed, Mesh 2, $k = 2$

Figure 5.29: Numerical solution for Stokes problem in 2D obtained with two different meshes with triangular elements solved using NEFEM formulation.



(a) $||e_p||_{\mathcal{L}_2}$ **(b)** $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$

Figure 5.30: h-convergence for different degrees of approximation k for 2D Stokes problem for meshes with triangular elements using classical formulation of NEFEM-HDG.



(a) $||e_p||_{\mathcal{L}_2}$ **(b)** $||e_{\boldsymbol{u}}||_{\mathcal{L}_2}$ and $||e_{\boldsymbol{u}^\star}||_{\mathcal{L}_2}$

Figure 5.31: h-convergence for different degrees of approximation k for 2D Stokes problem for meshes with triangular elements using Neumann formulation of NEFEM-HDG.

5.4 Degree adaptivite HDG for Stokes equations

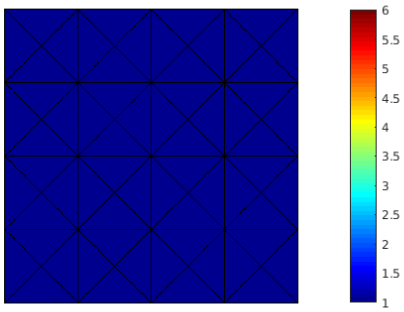
The last set of examples are performed to show the working of degree adaptive HDG method to solve Stokes' equations. One example each for isoparametric formulation and NEFEM formulation are shown in case of 2-D while an example using isoparametric formulation is shown in the 3-D case. A comparison of the

performance of the degree adaptive method is done with the cases of uniform k and h mesh refinement.

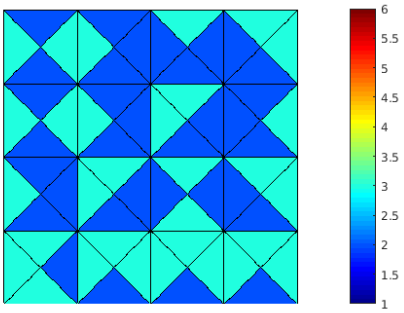
5.4.1 The 2-D case

5.4.1.1 Isoparametric Formulation

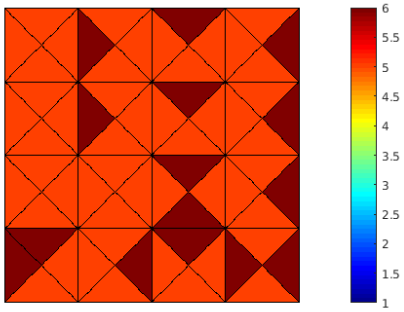
In order to illustrate the results of degree adaptive HDG for Stokes equations for a 2-D case using isoparametric formulation, the model problem in (3.1) is solved in $\Omega :=]0,1[\times]0,1[$ with $\Gamma_N = \{(x,y) \in \partial\Omega | y = 0\}$ and $\Gamma_D = \partial\Omega \backslash \Gamma_N$. The source and boundary conditions are taken such that the analytical solution is that proposed by [56], with the coefficients appearing in the analytical expression of the velocity to set to $a = 1$, $b = 1$ and $\lambda = 10$. The problem is solved in meshes with triangular elements.



(a) Initial configuration



(b) $\varepsilon = 0.5 \times 10^{-2}$, after final iteration



(c) $\varepsilon = 0.5 \times 10^{-4}$, after final iteration

Figure 5.32: Distribution of k in the elements for different tolerance values before and after degree adaptive mesh refinement

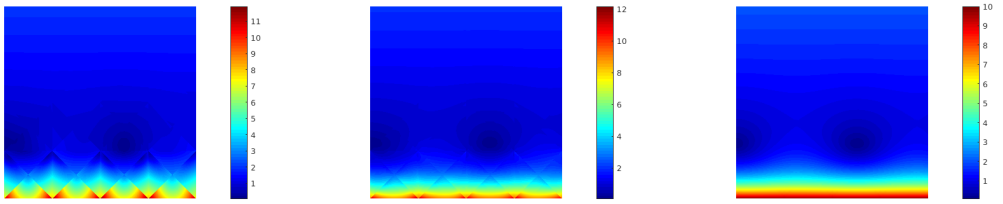
The desired tolerance of error is set to two different values of $\varepsilon = 0.5 \times 10^{-2}$ and $\varepsilon = 0.5 \times 10^{-4}$ (i.e. two and four significant digits, see [59], which cover the usual engineering accuracy needs). The element wise error limit is set to 0.5ε. Figure 5.32a shows the initial distribution of k for a mesh with 64 elements. Figure 5.32b shows the distribution of k after the error in the computation is less than the set tolerance of $\varepsilon = 0.5 \times 10^{-2}$ following the degree adaptive mesh refinement process, while 5.32c shows a similar distribution of k after the error in the computation is less than the set tolerance of $\varepsilon = 0.5 \times 10^{-4}$.

Figure 5.33 shows the pressure and velocity profiles after first and final iterations of degree adaptive mesh refined formulation for both the values of the set error tolerance. These results are obtained with the methodology proposed in Section 4.2. From the figure it is evident that there is a considerable improvement in the solution after the degree adaptive process. However, visual inspection does not provide a concrete proof of the performance of the degree adaptive method.

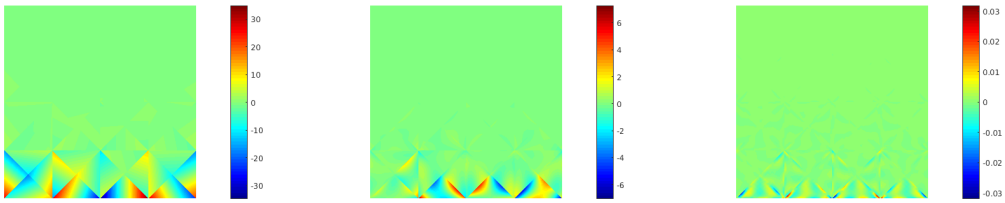
In order to compare the performance of the degree adaptive method, error of $\boldsymbol{u}_h$ in the $\mathcal{L}_2(\Omega)$ norm plotted as a function of $\sqrt{\dim(\hat{\boldsymbol{u}})}$ in Figure 5.34 for three different cases, degree adaptive mesh refinement strategy, uniform k and h refinements. The initial mesh has 64 elements with a characteristic size of $h = 1/4\sqrt{2}$ and uniform degree of approximation of 1 in all its elements. In the case of uniform k refinement, the value of k is increased from 1 to 4, whereas in the case of uniform h refinement, the characteristic size is decreased linearly from $h = 1/4\sqrt{2}$ to $h = 1/32\sqrt{2}$. From the figure it can be clearly seen that adaptive mesh refinement technique outperforms the uniform h refinement strategy while it is on par with the uniform k refinement strategy.

5.4.1.2 NEFEM Formulation

Next, in order to illustrate the results of degree adaptive HDG for Stokes equations for a 2-D case using NEFEM formulation, the model problem in (3.1) is solved in a domain bounded by the lines $x = 0$, $y = 1$ and $x = 1$ on the left right and top and by the curve $y = \big(1+\cos(5\pi x)\big)/10$ with $\Gamma_N = \{(x, y) \in \partial\Omega | y = \big(1+\cos(5\pi x)\big)/10\}$ and $\Gamma_D = \partial\Omega \backslash \Gamma_N$. The source and boundary conditions are taken such that the analytical solution is similar to that proposed by [56]. It is modified, however, in order to ensure that the boundary layer follows the curved boundary. The coefficients appearing in the analytical expression of the velocity to set to $a = 1$, $b = 1$ and $\lambda = 10$. The problem is solved in meshes with triangular elements.



(a) Velocity, after first iteration **(b)** Velocity, after final iteration, with $\varepsilon = 0.5 \times 10^{-2}$ **(c)** Velocity, after final iteration, with $\varepsilon = 0.5 \times 10^{-4}$



(d) Pressure, after first iteration **(e)** Pressure, after final iteration, with $\varepsilon = 0.5 \times 10^{-2}$ **(f)** Pressure, after final iteration, with $\varepsilon = 0.5 \times 10^{-4}$

Figure 5.33: Results obtained from degree adaptive HDG for 2-D case using isoparametric formulation.

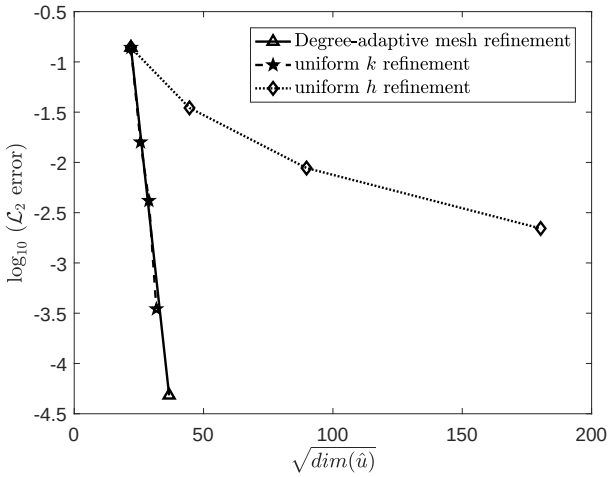
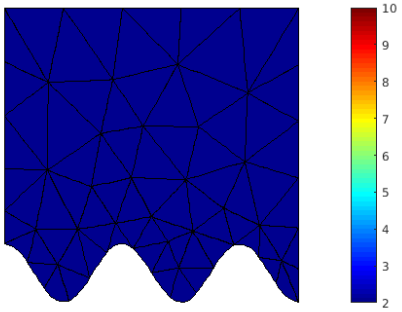
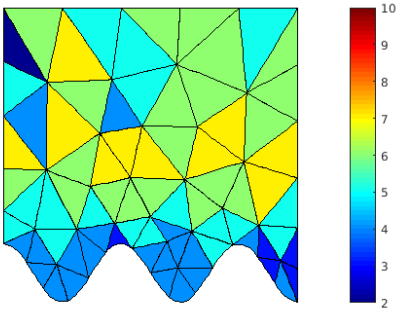


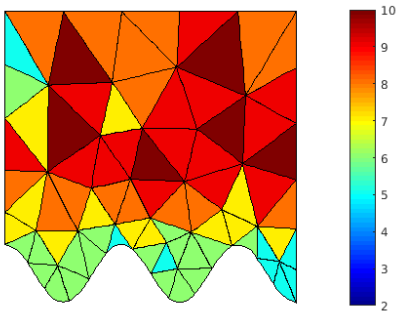
Figure 5.34: Comparison of performance of degree adaptive mesh refinement and uniform k and h refinement for 2-D case of HDG using isoparametric formulation: error of $\boldsymbol{u}_h$ in the $\mathcal{L}_2(\Omega)$ norm as a function of $\sqrt{\dim(\hat{\boldsymbol{u}})}$.



(a) Initial configuration



(b) $\varepsilon = 0.5 \times 10^{-2}$, after final iteration



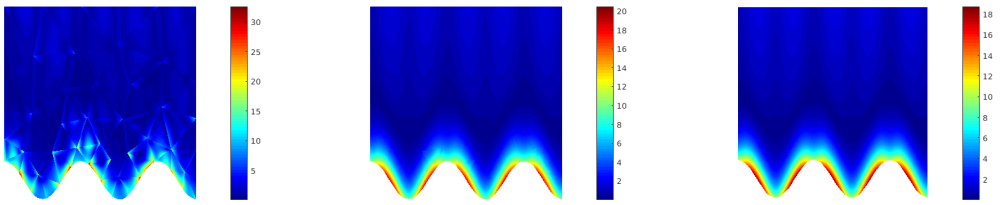
(c) $\varepsilon = 0.5 \times 10^{-4}$, after final iteration

Figure 5.35: Distribution of k in the elements for different tolerance values before and after degree adaptive mesh refinement

The desired tolerance of error is set to two different values of $\varepsilon = 0.5 \times 10^{-2}$ and $\varepsilon = 0.5 \times 10^{-4}$, with the element wise error limit set to 0.5ε. The initial distribution of k for a mesh with 64 elements is shown in Figure 5.35a. Figure 5.35b shows the distribution of k after the error in the computation is less than the set tolerance of $\varepsilon = 0.5 \times 10^{-2}$ following the degree adaptive mesh refinement process, while 5.35c shows a similar distribution of k after the error in the computation is less than the set tolerance of $\varepsilon = 0.5 \times 10^{-4}$.

Figure 5.36 shows the pressure and velocity profiles after first and final iterations of degree adaptive mesh refined formulation for both the values of the set error tolerance. These results are obtained with the methodology proposed in Section 4.2. From the figure it is evident that there is a considerable improvement in the solution after the degree adaptive process. Again, visual inspection does not provide a enough evidence of the performance of the degree adaptive method.

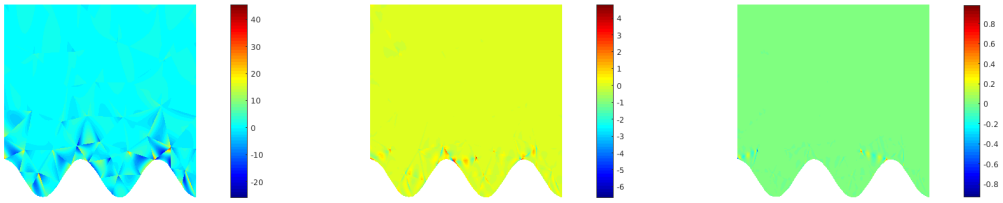
In order to compare the performance of the degree adaptive method, error of $\boldsymbol{u}_h$ in the $\mathcal{L}_2(\Omega)$ norm plotted as a function of $\sqrt{\dim(\hat{\boldsymbol{u}})}$ in Figure 5.37 for three different cases, degree adaptive mesh refinement strategy, uniform k and h refinements. The initial mesh has 64 elements with a characteristic size of $h = 1/4\sqrt{2}$ and uniform degree of approximation of 1 in all its elements. In the case of uniform k refinement, the value of k is increased from 1 to 4, whereas in the case of uniform h refinement, the characteristic size is decreased linearly from $h = 1/4\sqrt{2}$ to $h = 1/32\sqrt{2}$. From the figure it can be clearly seen that adaptive mesh refinement technique outperforms the uniform h refinement strategy while it is on par with the uniform k refinement strategy.



(a) Velocity, after first iteration

(b) Velocity, after final iteration, with $\varepsilon = 0.5 \times 10^{-2}$

(c) Velocity, after final iteration, with $\varepsilon = 0.5 \times 10^{-4}$



(d) Pressure, after first iteration

(e) Pressure, after final iteration, with $\varepsilon = 0.5 \times 10^{-2}$

(f) Pressure, after final iteration, with $\varepsilon = 0.5 \times 10^{-4}$

Figure 5.36: Results obtained from degree adaptive HDG for 2-D case using isoparametric formulation.

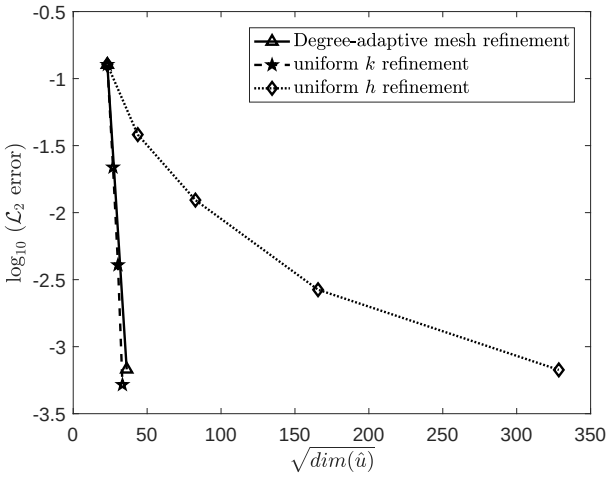


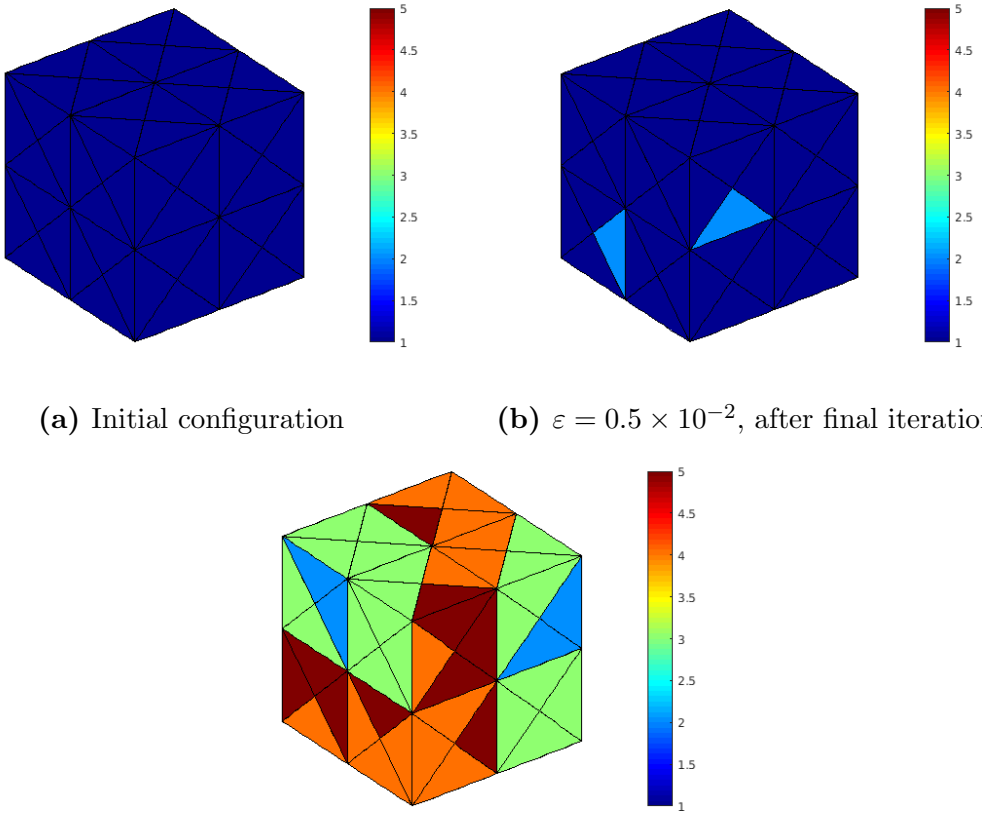
Figure 5.37: Comparison of performance of degree adaptive mesh refinement and uniform k and h refinement for 2-D case of HDG using isoparametric formulation: error of $\boldsymbol{u}_h$ in the $\mathcal{L}_2(\Omega)$ norm as a function of $\sqrt{\dim(\hat{\boldsymbol{u}})}$.

5.4.2 The 3-D case

In this section the results of degree adaptive HDG for Stokes equations for a 2-D case using isoparametric formulation are illustrated, the model problem in (3.1) is solved in $\Omega :=]0,1[\times]0,1[\times]0,1[$ with $\Gamma_N = \{(x,y,z) \in \partial\Omega | y = 0\}$ and $\Gamma_D = \partial\Omega \backslash \Gamma_N$. The source and boundary conditions are taken such that the analytical solution is that proposed by [57], with the coefficients appearing in the analytical expression of the velocity to set to $a = 1$ and $b = 0.5$. The problem is solved in meshes with tetrahedral elements.

Similar to the 2-D case, the desired tolerance of error is set to two different values of $\varepsilon = 0.5 \times 10^{-2}$ and $\varepsilon = 0.5 \times 10^{-4}$. The element wise error limit is set to 0.5ε. Figure 5.38a shows the initial distribution of k for a mesh with 24 elements. Figure 5.38b shows the distribution of k after the error in the computation is less than the set tolerance of $\varepsilon = 0.5 \times 10^{-2}$ following the degree adaptive mesh refinement process, while 5.38c shows a similar distribution of k after the error in the computation is less than the set tolerance of $\varepsilon = 0.5 \times 10^{-4}$.

Figure 5.39 shows the pressure and velocity profiles after first and final iterations of degree adaptive mesh refined formulation for both the values of the set error tolerance. As mentioned previously, these results are obtained with the degree adaptivity methodology proposed in Section 4.2. From the figure it is evident that there is a considerable improvement in the solution after the degree adaptive process. This is however not sufficient to provide a tangible proof of the performance of the degree adaptive method.



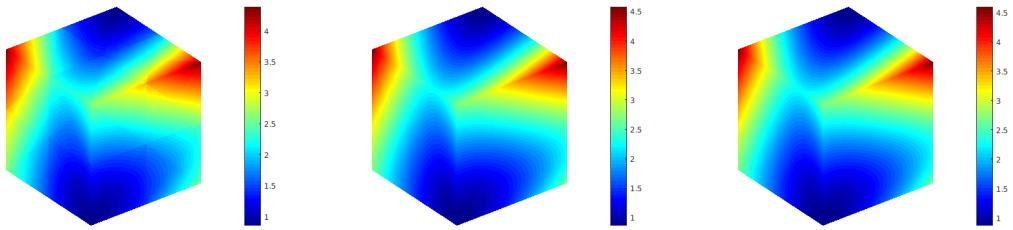
(a) Initial configuration

(b) $\varepsilon = 0.5 \times 10^{-2}$, after final iteration

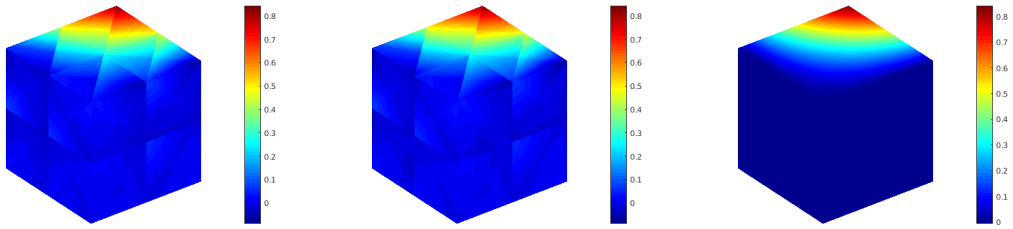
(c) $\varepsilon = 0.5 \times 10^{-4}$, after final iteration

Figure 5.38: Distribution of k in the elements for different tolerance values before and after degree adaptive mesh refinement

In order to compare the performance of the degree adaptive method, error of $\boldsymbol{u}_h$ in the $\mathcal{L}_2(\Omega)$ norm plotted as a function of $\sqrt{\dim(\hat{\boldsymbol{u}})}$ in Figure 5.40 for three different cases, degree adaptive mesh refinement strategy, uniform k and h refinements. The initial mesh has 24 elements with a characteristic size of $h = 1/2$ and uniform degree of approximation of 1 in all its elements. In the case of uniform k refinement, the value of k is increased from 1 to 4, whereas in the case of uniform h refinement, the characteristic size is decreased linearly from $h = 1/4\sqrt{2}$ to $h = 1/32\sqrt{2}$. From the figure it can be clearly seen that adaptive mesh refinement technique outperforms the uniform h refinement strategy while it is on par with the uniform k refinement strategy.



(a) Velocity, after first iteration **(b)** Velocity, after final iteration, with $\varepsilon = 0.5 \times 10^{-2}$ **(c)** Velocity, after final iteration, with $\varepsilon = 0.5 \times 10^{-4}$



(d) Pressure, after first iteration **(e)** Pressure, after final iteration, with $\varepsilon = 0.5 \times 10^{-2}$ **(f)** Pressure, after final iteration, with $\varepsilon = 0.5 \times 10^{-4}$

Figure 5.39: Results obtained from degree adaptive HDG for 2-D case using isoparametric formulation.

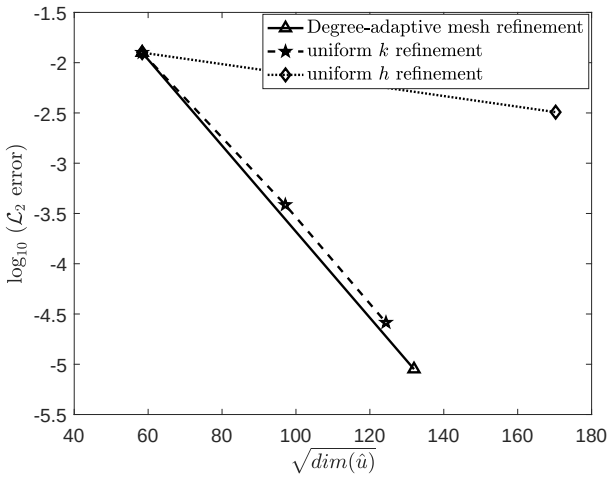


Figure 5.40: Comparison of performance of degree adaptive mesh refinement and uniform k and h refinement for 2-D case of HDG using isoparametric formulation: error of $\boldsymbol{u}_h$ in the $\mathcal{L}_2(\Omega)$ norm as a function of $\sqrt{\dim(\hat{\boldsymbol{u}})}$.

Chapter 6

Conclusions and future developments

The main purpose of this work was to gain a working understanding of the HDG method and NEFEM, two relatively new approaches in the area of computational methods based on finite elements, in order to integrate these two cutting-edge techniques seamlessly into one working code for the analysis of problems of interest to the aerodynamic aircraft design industry. A HDG degree adaptive technique is proposed for the solution of Stokes flow. The proposed adaptive technique provides uniform error distributions below a user defined tolerance, automatically placing the correct polynomial degree in each element. Adaptivity is driven by an error estimator derived exploiting the superconvergent properties of HDG, and involving only elemental computations. Thus, the error estimation is an inexpensive computation compared to the solution.

This study presents numerical examples in two and three dimensions for the Stokes problems solved using both uniform and non-uniform degrees of approximation in its elements, using both the isoparametric and NEFEM formulations. In all the examples, the implementation of the codes developed during this work are tested for up to order 4. In particular, for any order of approximation, optimal rate of convergence for both the primal and dual variables are consistently obtained. The postprocessing technique that enables to compute a solution that converges with a rate $k+2$ for a k-th degree approximation is detailed. This is a unique feature of the HDG formulation considered in this research and only requires the solution of an element-by-element problem. An example each is shown to show the advantages of degree adaptive HDG technique using isoparametric formulation in 2-D and 3-D and NEFEM formulation in 2D.

NEFEM integrated to HDG presents a case of a high-order method that promises to be devoid of the issues that are preventing the industry from embracing high-order methods. However, there is a lot of research that needs to done in order to

prove the competency of the method. The extension of degree adaptive NEFEM to 3D, proving the competency of the degree adaptive method through more practical applications are some of the work that need immediate attention and will be the areas of focus in the work that will follow this thesis.

The research presented here is an intermediate step towards the development of a 3D DG solver able to tackle realistic flow problems. The incorporation of the nonlinear terms corresponding to the Navier-Stokes equations are required as well as the adoption of efficient methods and preconditioners for the numerical solution of the linear system that arise in each Newton-Raphson iteration during the solution of the non-linear systems of equations. For the compressible Navier-Stokes equations it is necessary to implement efficient and robust shock capturing techniques that are able to resolve shocks within an element without mesh refinement in order to fully exploit the advantages of high-order approximations. Finally, the implementation of a turbulence model is mandatory in order to solve problems of industrial interest.

Bibliography

[1] Kroll, N., Bieler, H., Deconinck, H., Couaillier, V., van der Ven, H., Sorensen,K. (editors): ADIGMA- A European initiative on the development of adaptive higher-order variational methods for aerospace applications. *Notes on Numerical Fluid Mechanics and Multidisciplinary Design*, Volume 113, Springer, 2010 1, 2, 3

[2] Wang, Z.J., Fidkowski, K., Abgrall, R., Bassi, F., Caraeni, D., Cary, A., Deconinck, H., Hartmann, R., Hillewaert, K., Huynh, H.T., Kroll, N., May, G., Persson, P., van Leer, B., Visbal, M.: High-Order CFD Methods: Current Status and Perspective. *International Journal for Numerical Methods in Fluids* 2012; 00: 1-42. 1

[3] Martin, M.P., Taylor, E.M., Wu, M., Weirs, V.G.: A bandwidth-optimized WENO scheme for the effective direct numerical simulation of compressible turbulence. *Journal of Computational Physics* 2006; 220(1): 270–289. 1

[4] Karniadakis, G.E., Cockburn, B., Shu C.W.: The development of discontinuous Galerkin method: *Lecture Notes in Computational Science and Engineering* 2000, vol. 11, Springer. 1

[5] Ekaterinaris, J.A.: High-order accurate, low numerical diffusion methods for aerodynamics: *Progress in Aerospace Sciences* 2005; 41 (3–4): 192–300. 1

[6] Wang, Z.: High-order methods for the Euler and Navier–Stokes equations on unstructured grids: *Progress in Aerospace Sciences* 2007; 43 (1-3): 1-41. 1

[7] Cockburn, B., Shu C.W.: The Runge–Kutta discontinuous Galerkin method for conservation laws V: Multidimensional systems. *Journal of Computational Physics* 1998; 141(2): 199–224. 1

[8] Montlaur, A., Fernández-Méndez, S., Huerta, A.: Discontinuous Galerkin methods for the Stokes equations using divergence-free approximations. *In-*

ternational Journal for Numerical Methods in Fluids 2008; 57: 1071-1092. 1, 15

[9] Arnold, D. N.: An interior penalty finite element method with discontinuous elements. *SIAM Journal on Numerical Analysis* 1982; 19(4), 742–760. 2

[10] Peraire, J., Persson P.O.: The compact discontinuous Galerkin (CDG) method for elliptic problems. *SIAM Journal on Scientific Computing* 2008: 30(4), 1806–1824. 2

[11] Nguyen, N.C., Peraire J., Cockburn B.: An implicit high-order hybridizable discontinuous Galerkin method for the incompressible Navier–Stokes equations. *Journal of Computational Physics* 2011; 230(4): 1147–70. 2

[12] Cockburn, B., Gopalakrishnan, J., Lazarov, R.: Unified hybridization of discontinuous Galerkin, mixed, and continuous Galerkin methods for second order elliptic problems. *SIAM Journal on Numerical Analysis* 2009; 47(2): 1319–65. 2, 17, 19

[13] Cockburn, B., Dong, B., Guzmán, J,: A superconvergent LDG-hybridizable Galerkin method for second-order elliptic problems. *Mathematics of Computation* 2008; 77(264) :1887–916. 2, 17, 19, 23

[14] Nguyen, N.C., Peraire, J., Cockburn, B.: An implicit high-order hybridizable discontinuous Galerkin method for linear convection–diffusion equations. *Journal of Computational Physics* 2009; 228(9): 3232–54. 2, 17, 19

[15] Nguyen, N.C., Peraire, J., Cockburn, B.: An implicit high-order hybridizable discontinuous Galerkin method for nonlinear convection–diffusion equations. *Journal of Computational Physics* 2009; 228(23): 8841–55. 2, 17, 19

[16] Nguyen, N.C., Peraire, J., Cockburn, B.: A hybridizable discontinuous Galerkin method for Stokes flow. *Computational Methods Applied Mechanics and Engineering* 2010; 199(9–12): 582–97. 2, 17, 19

[17] Nguyen, N. C., Peraire, J., and Cockburn B.: High-order implicit hybridizable discontinuous Galerkin methods for acoustics and elastodynamics. *Journal of Computational Physics* 2011; 230(10), 3695–3718. 2

[18] Nguyen, N. C., Peraire, J., and Cockburn B.: Hybridizable discontinuous Galerkin methods for the time-harmonic Maxwell's equations. Journal of Computational Physics 2011; 230(19), 7151–7175. 2

[19] Nguyen, N. C., Peraire, J., and Cockburn B.: An implicit high-order hybridizable discontinuous Galerkin method for the incompressible Navier-Stokes equations. *Journal of Computational Physics* 2011: 230(4), 1147–1170. 2, 17, 19

[20] Peraire, J., Nguyen, N. C., and Cockburn B.: A hybridizable discontinuous Galerkin method for the compressible Euler and Navier-Stokes equations. In *In Proceedings of the 48th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA-2010, Orlando, Florida,. 2

[21] Kirby, R., Sherwin, S.,J., and Cockburn B.: To CG or to HDG: A comparative study. *Journal of Scientific Computing* 2011: 51(1), 183–212. 2

[22] Arnold, D. N., Brezzi, F.: Mixed and nonconforming finite element methods: implementation, postprocessing and error estimates. *RAIRO – Modélisation mathématique et analyse numérique* 1985: 19(1), 7–32. 2

[23] Giorgiani, G., Fernándes-Méndez, S., Huerta, A.: Hybridizable Discontinuous Galerkin with degree adaptivity for the incompressible Navier–Stokes equations. *Computers and Fluids- Journal*, 2014; 01(11) 2

[24] Sevilla, R., Fernándes-Méndez, S., Huerta, A.: NURBS-Enhanced Finite Element Method (NEFEM). A seamless bridge between CAD and FEM. *Archives of Computational Methods in Engineering*,2011; 18 (4); 441-484 v, 3, 26, 27, 29, 30, 31, 32, 37

[25] Sevilla, R., Fernández-Méndez, S., Huerta, A.: Comparison of high-order curved finite elements. *International Journal for Numerical Methods in Engineering*, 2011; 87(8), 719–734 3, 37

[26] Kirby, B.J.: Micro- and Nanoscale Fluid Mechanics: Transport in Microfluidic Devices. *Cambridge University Press*, 2010. 5

[27] Dusenbery, D.B.: Living at Micro Scale. *Harvard University Press*, Cambridge, 2009. 5

[28] Vogel, S.: Life in Moving Fluids. *Princeton University Press*, 1994. 5

[29] Donea, J., Huerta, A.: Finite Element Methods for Flow Problems. Chichester: *John Wiley & Sons*, 2003. 6, 8, 9, 12, 13

[30] Sevilla, R., Huerta, A.: Tutorial on Hybridizable Discontinuous Galerkin (HDG) for Second-Order Elliptic Problems. *Advanced Finite Element Technologies, CISM International Centre for Mechanical Sciences*, 2016, 566:105-129. 17

[31] Giorgiani, G., Modesto, D., Fernandés-Méndez, S., Huerta, A.: High-order continuous and discontinuous Galerkin methods for wave problems. *International Journal for Numerical Methods in Fluids*, 2013, 73(10): 883–903. 19, 32

[32] Giorgiani, G., Fernandés-Méndez, S., Huerta, A.: Hybridizable discontinuous Galerkin p-adaptivity for wave problems. *International Journal for Numerical Methods in Fluids*, 2013, 72(12): 1244–1262. 19, 32

[33] Hesthaven, J.S., Warburton, T.: Nodal discontinuous Galerkin methods. Algorithms, analysis and applications. *Springer*, 2008.

[34] Cockburn, B., Shu, C.W.: The local discontinuous Galerkin method for time-dependent convection-diffusion systems. *SIAM Journal on Numerical Analysis*, 1998, 35(6): 2440-2463.

[35] Arnold, D.N., Brezzi, F., Cockburn, B., Marini, L.D.: Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis*, 2002, 39(5): 1749-1779.

[36] Cockburn, B., Guzmán, J., Wang, H.: Superconvergent discontinuous Galerkin methods for second-order elliptic problems. *Mathematics of Computation*, 2009, 78(265): 1–24. 23

[37] Cockburn, B., Fu, G.: A note on the devising of superconvergent HDG methods for Stokes flow by M-decompositions. *IMA Journal of Numerical Analysis*, 2016, 00: 1–20. 23

[38] Ergatoudis, J., Irons, B.M., Zienkiewicz, O.C.: Curved isoparametric 'quadrilateral' elements for finite element analysis. *International Journal of Solids and Structures*, 1968, 4(1): 31–42. 26

[39] Sevilla, R., Hassan, O., Morgan, K.: The use of hybrid meshes to improve the efficiency of a discontinuous Galerkin method for the solution of Maxwell's equations *Computers and Structures*, 2014, 137: 2-13. viii, 27

[40] Bergot, M., Cohen, G., Duruflé, M.: Higher-order finite elements for hybrid meshes using new nodal pyramidal elements. *Journal of Scientific Computing*, 2010, 42(3): 345–81. 27

[41] Taylor, M.A., Wingate, B.A., Vincent, R.E.: An algorithm for computing Fekete points in the triangle. *SIAM Journal on Numerical Analysis*, 2000, 38(5): 1707–20. 27

[42] Luo, H., Pozrikidis, C.: A Lobatto interpolation grid in the tetrahedron. *IMA Journal of Applied Mathematics*, 2006 71(2): 298–313. 27

[43] Zhong, Q.X., Sevilla, R., Hassan, O., Morgan, K.: The generation of arbitrary order curved meshes for 3D finite element analysis. *Journal of Computational Mechanics*, 51(3):361-374, 2013 37

[44] Zienkiewicz O.C., Morgan K.: Finite elements and approximation. *Dover, Mineola*, 1983. 37

[45] Zienkiewicz O.C., Taylor R.L.: The Finite Element Method, vol 1. The basis, 5th edn. *Butterwort-Heinemann, Oxford*, 2000. 37

[46] Ainsworth, M., Oden, J.T.: A posteriori error estimation in finite element analysis. *Pure and Applied Mathematics (New York)*. New York: Wiley-Interscience [John Wiley & Sons], 2000. 33

[47] Huerta, A., Rodriguez-Ferran, A., Díez, P., Sarrate, J.: Adaptive finite element strategies based on error assessment. *International Journal for Numerical Methods in Engineering*, 1999, 46(10): 1803–1818. 33

[48] Díez, P., Arroyo,M., Huerta, A.: Adaptivity based on error estimation for viscoplastic softening materials. *Mechanics of Cohesive-frictional Materials*, 2000, 5(2): 87–112. 33

[49] Sauer-Budge, A. M., Bonet, J., Huerta, A., Peraire J.: Computing bounds for linear functionals of exact weak solutions to Poisson's equation. *SIAM Journal on Numerical Analysis*, 2004, 42(4): 1610–1630. 33

[50] Parés, N., Díez, P., Huerta, A.: Subdomain-based flux-free a posteriori error estimators. Cover image Computer Methods in Applied Mechanics and Engineering *Computer Methods in Applied Mechanics and Engineering*, 2006, 195(4-6): 297–323. 33

[51] Parés, N., Díez, P., Huerta, A.: Bounds of functional outputs for parabolic problems. II. Bounds of the exact solution. *Computer Methods in Applied Mechanics and Engineering*, 2008, 197(19-20): 1661–1679. 33

[52] Parés, N., Díez, P., Huerta, A.: Exact bounds for linear outputs of the advection-diffusion-reaction equation using flux-free error estimates. *SIAM Journal on Scientific Computing*, 2009, 31(4): 3064–3089. 33

[53] Díez, P., Rodenas, J.J., Zienkiewicz, O.C.: Equilibrated patch recovery error estimates: simple and accurate upper bounds of the error. *International Journal for Numerical Methods in Engineering*, 2007, 69(10): 2075–2098. 33

[54] Ladevéze, P., Pelle, J.,P.: Mastering calculations in linear and nonlinear mechanics. *Mechanical Engineering Series. New York: Springer-Verlag*, 2005. Translated from the 2001 French original by Theofanis Strouboulis. 33

[55] Díez, P., Huerta, A.: A unified approach to remeshing strategies for finite element h-adaptivity. *Computer Methods in Applied Mechanics and Engineering*, 1999, 176: 215-229 33, 34, 67

[56] Wang, C.Y.: Exact solutions of the steady state Navier Stokes equations. *Annual Review of Fluid Mechanics*, 1991, 23:159(177). 38, 73, 78, 79

[57] Ethier, C.R., Steinman, D.A.: Exact fully 3-D Navier-Stokes solutions for benchmarking. *International Journal for Numerical Methods in Fluids*, 1991, 19(5): 369-375 54, 83

[58] Hughes, T.J.R.: The Finite Element Method. *Prentice Hall International*, Stanford, 1987. 67

[59] Higham, N. J.: Accuracy and stability of numerical algorithms (Second ed.). *Philadelphia, PA: Society for Industrial and Applied Mathematics* (SIAM), 2002. 79

[60] Piegl, W., Tiller, W.: The NURBS book. *Springer*, 1995. 95

Appendices

Appendix A

Curves

The two most common ways of describing curves in geometric modeling are implicit equations and parametric functions [60]. A curve lying in the xy plane can be represented using the implicit equation of the form $f(x, y) = 0$. On the other hand, in parametric form, each of the coordinates of a point on the curve is represented separately as an explicit function of an independent parameter

$$\boldsymbol{C}(\lambda) = (x(\lambda), y(\lambda)) \quad a \leq \lambda \leq b.$$

A.1 Power basis form of a curve

A great variety of curves are obtained by allowing the coordinate functions $x(u)$ and $y(u)$ to be arbitrary. However, while implementing a geometric modeling system, there are trade-offs. A class of functions that are capable of precisely representing all the curves the users of the system need and are easily, efficiently, and accurately processed in a computer are preferred over the others. A widely used class of functions is the polynomials. Power basis and Bézier are the two common methods of expressing polynomial functions.

An nth-degree power basis curve is given by

$$\boldsymbol{C}(\lambda) = (x(\lambda), y(\lambda), z(\lambda)) = \sum_{i=0}^{n} \boldsymbol{a}_i \lambda^i \quad 0 \leq \lambda \leq 1. \tag{A.1}$$

The $\boldsymbol{a}_i = (x_i, y_i, z_i)$ are vectors, hence

$$x(\lambda) = \sum_{i=0}^{n} x_i \lambda^i \quad y(\lambda) = \sum_{i=0}^{n} y_i \lambda^i \quad z(\lambda) = \sum_{i=0}^{n} z_i \lambda^i.$$

A.2 Bézier curves

Bézier curve is yet another polynomial curve. Since both the power basis and Bézier use polynomials for their coordinate functions, these forms are mathematically equivalent. The shortcomings of the power basis form are remedied by the Bézier method. The coefficients $\{\boldsymbol{a}_i\}$ of the power basis form convey very little geometrical insight about the shape of the curve. Also algorithms for processing power basis polynomials have an algebraic rather than a geometric essence.

An nth-degree Bézier curve is defined by

$$\boldsymbol{C}(\lambda) = \sum_{i=0}^{n} N_{i_n}(\lambda)\boldsymbol{P}_i \quad 0 \leq \lambda \leq 1. \tag{A.2}$$

The basis functions, $\{N_{i,n}(\lambda)\}$, are the classical nth-degree Bernstein polynomials given by

$$N_{i,n}(\lambda) = \frac{n!}{i!(n-1)!}\lambda^i(1-\lambda)^{n-i}. \tag{A.3}$$

The geometric coefficients $\{\boldsymbol{P}_i\}$ are the control points.

A.3 Rational Bézier curves

Although polynomials offer many advantages, there exist a number of important curves such as, circles, ellipses and hyperbolas, to name a few, which cannot be represented precisely using polynomials. It is known from classical mathematics that all the conic curves, including the circle can be represented using rational functions, which are defined as the ratio of two polynomials.

An nth-degree rational Bézier curve is defined by

$$\boldsymbol{C}(\lambda) = \frac{\sum_{i=0}^{n} N_{i,n}(\lambda)w_i\boldsymbol{P}_i}{\sum_{i=0}^{n} N_{i,n}(\lambda)w_i} \quad 0 \leq \lambda \leq 1. \tag{A.4}$$

The $\boldsymbol{P}_i = (x_i, y_i, z_i)$ and $N_{i,n}(\lambda)$ are as before; the w_i are weights.

A.4 B-spline basis functions and curves

Curves consisting of just one polynomial or rational segment are often inadequate as they have a number of shortcomings. While a high degree is required in order to

satisfy a large number of constraints and to accurately fit some complex shaped, these curves are inefficient to process and are numerically unstable. Also single-segment curves are not well suited to interactive shape design as the control is not sufficiently local. The solution is to use curves which are piecewise polynomials, or piecewise rational.

In order to define a B-spline basis function, it is first requires to describe a nondecreasing sequence of real numbers, $\Lambda = \{\lambda_0, ..., \lambda_m\}$ where $\lambda_i \leq \lambda_{i+1}, i = 0, ..., m-1$. The λ_i are called knots, and Λ is the knot vector. The ith B-spline basis function of p-degree, denoted by $B_{i,p}(\lambda)$, is defined as

$$B_{i,0}(\lambda) = \begin{cases} 1 & \text{if } \lambda_i \leq \lambda < \lambda_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$B_{i,p}(\lambda) = \frac{\lambda - \lambda_i}{\lambda_{i+p} - \lambda_i} B_{i,p-1}(\lambda) + \frac{\lambda_{i+p+1} - \lambda}{\lambda_{i+p+1} - \lambda_{i+1}} B_{i+1,p-1}(\lambda). \tag{A.5}$$

A pth-degree B-spline curve is defined by

$$\boldsymbol{C}(\lambda) = \sum_{i=0}^{n} B_{i,p}(\lambda) \boldsymbol{P}_i \quad a \leq \lambda \leq b, \tag{A.6}$$

where the $\{\boldsymbol{P}_i\}$ are the control points, and the $\{B_{i,p}(\lambda)\}$ are the pth-degree B-spline basis function, see Equation (A.5) defined on the nonperiodic (and nonuniform) knot vector

$$\Lambda = \{\underbrace{a, ..., a}_{p+1}, \lambda_{p+1}, ..., \lambda_{m-p-1}, \underbrace{b, ..., b}_{p+1}\},$$

(m+1 knots). The polygon formed by the $\{\boldsymbol{P}_i\}$ is the control polygon.

A.5 Non-Uniform Rational B-Splines (NURBS)

A pth-degree NURBS curve is a piecewise rational function defined in parametric form as

$$\boldsymbol{C}(\lambda) = \frac{\sum_{i=0}^{n} B_{i,p}(\lambda) w_i \boldsymbol{P}_i}{\sum_{i=0}^{n} B_{i,p}(\lambda) w_i} \quad a \leq \lambda \leq b, \tag{A.7}$$

where the $\{\boldsymbol{P}_i\}$, the $\{w_i\}$ and the $\{B_{i,p}(\lambda)\}$ are as before.

Appendix B

Implementational Details

This appendix is devoted to the detailed presentation of the matrices and vectors appearing in the discrete version of both the local and global problems induced by the HDG method.

B.1 Isoparametric formulation

The following compact form of the interpolation functions is introduced

$$\begin{gathered}
\mathbf{N} = [N_1 \;\; N_2 \;\; ... \;\; N_{\mathtt{n_{en}}}]^T, \quad \widehat{\mathbf{N}} = [\widehat{N}_1 \;\; \widehat{N}_2 \;\; ... \;\; \widehat{N}_{\mathtt{n_{fn}}}]^T, \\
\mathbf{N}_{\boldsymbol{n}} = [N_1\boldsymbol{n} \;\; N_2\boldsymbol{n} \;\; ... \;\; N_{\mathtt{n_{en}}}\boldsymbol{n}]^T, \quad \widehat{\mathbf{N}}_{\boldsymbol{n}} = [\widehat{N}_1\boldsymbol{n} \;\; \widehat{N}_2\boldsymbol{n} \;\; ... \;\; \widehat{N}_{\mathtt{n_{fn}}}\boldsymbol{n}]^T, \\
\boldsymbol{\nabla}\mathbf{N} = [(\boldsymbol{J}^{-1}\boldsymbol{\nabla}N_1)^T \;\; (\boldsymbol{J}^{-1}\boldsymbol{\nabla}N_2)^T \;\; ... \;\; (\boldsymbol{J}^{-1}\boldsymbol{\nabla}N_{\mathtt{n_{en}}})^T]^T, \\
\mathbf{N}_{\mathtt{n_{sd}}} = [N_1\boldsymbol{I}_{\mathtt{n_{sd}}} \;\; N_2\boldsymbol{I}_{\mathtt{n_{sd}}} \;\; ... \;\; N_{\mathtt{n_{en}}}\boldsymbol{I}_{\mathtt{n_{sd}}}]^T, \\
\mathbf{N}_{\mathtt{n_{sd}^2}} = [N_1\boldsymbol{I}_{\mathtt{n_{sd}^2}} \;\; N_2\boldsymbol{I}_{\mathtt{n_{sd}^2}} \;\; ... \;\; N_{\mathtt{n_{sd}^2}}\boldsymbol{I}_{\mathtt{n_{sd}^2}}]^T,
\end{gathered}$$

where $\boldsymbol{n} = (n_1, ..., n_{\mathtt{n_{sd}}})$ denotes the outward unit normal vector to an edge/face and $\boldsymbol{I}_{\mathtt{n_{sd}}}$ and $\boldsymbol{I}_{\mathtt{n_{sd}^2}}$ are identity matrices of dimension $\mathtt{n_{sd}}$ and $\mathtt{n_{sd}^2}$, respectively.

The matrices and vectors appearing in Equations (3.11) and (3.17), computed for each element $i = 1, ..., \mathtt{n_{en}}$, can be expressed as

$$[\mathbf{A}_{LL}]_i = \sum_{\mathtt{g}=1}^{\mathtt{n_{ip}^e}} \mathbf{N}_{\mathtt{n_{sd}^2}}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}})\mathbf{N}_{\mathtt{n_{sd}^2}}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}})w_{\mathtt{g}}^{\mathtt{e}}$$

$$[\mathbf{A}_{Lu}]_i = -\sum_{\mathtt{g}=1}^{\mathtt{n_{ip}^e}} \boldsymbol{\nabla}\mathbf{N}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}})\mathbf{N}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}})w_{\mathtt{g}}^{\mathtt{e}}$$

$$[\mathbf{A}_{uL}]_i = \nu \sum_{\mathtt{g}=1}^{\mathtt{n_{ip}^e}} \mathbf{N}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}})\boldsymbol{\nabla}\mathbf{N}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}})w_{\mathtt{g}}^{\mathtt{e}} - \nu \sum_{\partial\Omega_i} \sum_{\mathtt{g}=1}^{\mathtt{n_{ip}^f}} \mathbf{N}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}})\mathbf{N}_{\boldsymbol{n}}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}})w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{uu}]_i = -\sum_{\partial\Omega_i} \tau_i \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}_{\mathtt{n}_{\mathtt{sd}}}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) \mathbf{N}_{\mathtt{n}_{\mathtt{sd}}}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{up}]_i = \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{e}}} \boldsymbol{\nabla}\mathbf{N}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}}) \mathbf{N}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}}) w_{\mathtt{g}}^{\mathtt{e}} - \sum_{\partial\Omega_i} \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}_{\boldsymbol{n}}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) \mathbf{N}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{pu}]_i = \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{e}}} \mathbf{N}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}}) \boldsymbol{\nabla}\mathbf{N}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}}) w_{\mathtt{g}}^{\mathtt{e}}$$

$$[\mathbf{a}_{\rho p}]_i = \sum_{\partial\Omega_i} \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{L\hat{u}}]_i = -\sum_{\partial\Omega_i \setminus \Gamma_D} \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}_{\boldsymbol{n}}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) \widehat{\mathbf{N}}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{u\hat{u}}]_i = -\sum_{\partial\Omega_i \setminus \Gamma_D} \tau_i \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) \widehat{\mathbf{N}}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{p\hat{u}}]_i = \sum_{\partial\Omega_i \setminus \Gamma_D} \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) \widehat{\mathbf{N}}_{\boldsymbol{n}}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{f}_L]_i = \sum_{\partial\Omega_i \cap \Gamma_D} \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}_{\boldsymbol{n}}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) \boldsymbol{u}_D^T\big(\boldsymbol{x}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}})\big) w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{f}_u]_i = -\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{e}}} \mathbf{N}_{\mathtt{n}_{\mathtt{sd}}}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}}) \boldsymbol{s}^T\big(\boldsymbol{x}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{e}})\big) w_{\mathtt{g}}^{\mathtt{e}} - \sum_{\partial\Omega_i \cap \Gamma_D} \tau_i \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}_{\mathtt{n}_{\mathtt{sd}}}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) \boldsymbol{u}_D^T\big(\boldsymbol{x}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}})\big) w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{f}_p]_i = \sum_{\partial\Omega_i \cap \Gamma_D} \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) \boldsymbol{n}\big(\boldsymbol{x}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}})\big) \boldsymbol{u}_D^T\big(\boldsymbol{x}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}})\big) w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{\hat{u}L}]_i = \sum_{\partial\Omega_i \setminus \Gamma_D} \nu \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \widehat{\mathbf{N}}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) \mathbf{N}_{\boldsymbol{n}}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{\hat{u}u}]_i = \sum_{\partial\Omega_i \setminus \Gamma_D} \tau_i \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \widehat{\mathbf{N}}(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) \mathbf{N}^T(\boldsymbol{\xi}_{\mathtt{g}}^{\mathtt{f}}) w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{\hat{u}p}]_i = \sum_{\partial\Omega_i\setminus\Gamma_D}\sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}} \widehat{\mathbf{N}}_{\boldsymbol{n}}(\boldsymbol{\xi}^{\mathtt{f}}_{\mathtt{g}})\mathbf{N}^T(\boldsymbol{\xi}^{\mathtt{f}}_{\mathtt{g}})w^{\mathtt{f}}_{\mathtt{g}}$$

$$[\mathbf{A}_{\hat{u}\hat{u}}]_i = -\sum_{\partial\Omega_i\setminus\Gamma_D}\tau_i\sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}} \widehat{\mathbf{N}}(\boldsymbol{\xi}^{\mathtt{f}}_{\mathtt{g}})\widehat{\mathbf{N}}^T(\boldsymbol{\xi}^{\mathtt{f}}_{\mathtt{g}})w^{\mathtt{f}}_{\mathtt{g}}$$

$$[\mathbf{f}_{\hat{u}}]_i = -\sum_{\partial\Omega_i\cap\Gamma_N}\sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}} \widehat{\mathbf{N}}(\boldsymbol{\xi}^{\mathtt{f}}_{\mathtt{g}})\boldsymbol{t}^T\big(\boldsymbol{x}(\boldsymbol{\xi}^{\mathtt{f}}_{\mathtt{g}})\big)w^{\mathtt{f}}_{\mathtt{g}}$$

$$[\boldsymbol{a}_{\rho\hat{u}}]_i = \sum_{\partial\Omega_i\setminus\Gamma_D}\sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}} \widehat{\mathbf{N}}^T_{\boldsymbol{n}}(\boldsymbol{\xi}^{\mathtt{f}}_{\mathtt{g}})w^{\mathtt{f}}_{\mathtt{g}}$$

$$[\mathbf{f}_{\rho}]_i = -\sum_{\partial\Omega_i\cap\Gamma_D}\sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}} \boldsymbol{n}\big(\boldsymbol{x}(\boldsymbol{\xi}^{\mathtt{f}}_{\mathtt{g}})\big)\boldsymbol{u}^T_D\big(\boldsymbol{x}(\boldsymbol{\xi}^{\mathtt{f}}_{\mathtt{g}})\big)w^{\mathtt{f}}_{\mathtt{g}}$$

In the above expressions, $\boldsymbol{\xi}^{\mathtt{e}}_{\mathtt{g}}$ and $w^{\mathtt{e}}_{\mathtt{g}}$ are the $\mathtt{n}^{\mathtt{e}}_{\mathtt{ip}}$ integration points and weights defined on the reference element and $\boldsymbol{\xi}^{\mathtt{f}}_{\mathtt{g}}$ and $w^{\mathtt{f}}_{\mathtt{g}}$ are the $\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}$ integration points and weights defined on the reference edge/face.

B.2 NEFEM formulation

The following compact form of the interpolation functions is introduced

$$\begin{gathered}
\mathbf{N}^{\mathbf{x}} = [N^x_1 \;\; N^x_2 \;\; ... \;\; N^x_{\mathtt{n_{en}}}]^T, \quad \widehat{\mathbf{N}^{\mathbf{x}}} = [\widehat{N^x}_1 \;\; \widehat{N^x}_2 \;\; ... \;\; \widehat{N^x}_{\mathtt{n_{fn}}}]^T,\\
\mathbf{N}^{\mathbf{x}}{}_{\boldsymbol{n}} = [N^x_1\boldsymbol{n} \;\; N^x_2\boldsymbol{n} \;\; ... \;\; N^x_{\mathtt{n_{en}}}\boldsymbol{n}]^T, \quad \widehat{\mathbf{N}^{\mathbf{x}}}_{\boldsymbol{n}} = [\widehat{N^x}_1\boldsymbol{n} \;\; \widehat{N^x}_2\boldsymbol{n} \;\; ... \;\; \widehat{N^x}_{\mathtt{n_{fn}}}\boldsymbol{n}]^T,\\
\boldsymbol{\nabla}\mathbf{N}^{\mathbf{x}} = [(\boldsymbol{\nabla}N^x_1)^T \;\; (\boldsymbol{\nabla}N^x_2)^T \;\; ... \;\; (\boldsymbol{\nabla}N^x_{\mathtt{n_{en}}})^T]^T,\\
\mathbf{N}^{\mathbf{x}}{}_{\mathtt{n_{sd}}} = [N^x_1\boldsymbol{I}_{\mathtt{n_{sd}}} \;\; N^x_2\boldsymbol{I}_{\mathtt{n_{sd}}} \;\; ... \;\; N^x_{\mathtt{n_{en}}}\boldsymbol{I}_{\mathtt{n_{sd}}}]^T,\\
\mathbf{N}^{\mathbf{x}}{}_{\mathtt{n^2_{sd}}} = [N^x_1\boldsymbol{I}_{\mathtt{n^2_{sd}}} \;\; N^x_2\boldsymbol{I}_{\mathtt{n^2_{sd}}} \;\; ... \;\; N^x_{\mathtt{n^2_{sd}}}\boldsymbol{I}_{\mathtt{n^2_{sd}}}]^T,
\end{gathered}$$

where $\boldsymbol{n} = (n_1, ..., n_{\mathtt{n_{sd}}})$ denotes the outward unit normal vector to an edge/face and $\boldsymbol{I}_{\mathtt{n_{sd}}}$ and $\boldsymbol{I}_{\mathtt{n^2_{sd}}}$ are identity matrices of dimension $\mathtt{n_{sd}}$ and $\mathtt{n^2_{sd}}$, respectively.

The matrices and vectors appearing in Equations (3.11) and (3.17), computed for each element $i = 1, ..., \mathtt{n_{en}}$, can be expressed as

$$[\mathbf{A}_{LL}]_i = \sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{e}}_{\mathtt{ip}}} \mathbf{N}^{\mathbf{x}}{}_{\mathtt{n^2_{sd}}}(\boldsymbol{x}^{\mathtt{e}}_{\mathtt{g}})\mathbf{N}^{\mathbf{x}T}{}_{\mathtt{n^2_{sd}}}(\boldsymbol{x}^{\mathtt{e}}_{\mathtt{g}})w^{\mathtt{e}}_{\mathtt{g}}$$

$$[\mathbf{A}_{Lu}]_i = -\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{e}}} \boldsymbol{\nabla}\mathbf{N}^{\mathbf{x}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{e}})\mathbf{N}^{\mathbf{x}T}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{e}})w_{\mathtt{g}}^{\mathtt{e}}$$

$$[\mathbf{A}_{uL}]_i = \nu\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{e}}} \mathbf{N}^{\mathbf{x}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{e}})\boldsymbol{\nabla}\mathbf{N}^{\mathbf{x}T}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{e}})w_{\mathtt{g}}^{\mathtt{e}} - \nu\sum_{\partial\Omega_i}\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^{\mathbf{x}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})\mathbf{N}^{\mathbf{x}T}_{\boldsymbol{n}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{uu}]_i = -\sum_{\partial\Omega_i}\tau_i\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^{\mathbf{x}}{}_{\mathtt{n}_{\mathtt{sd}}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})\mathbf{N}^{\mathbf{x}T}_{\mathtt{n}_{\mathtt{sd}}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{up}]_i = \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{e}}} \boldsymbol{\nabla}\mathbf{N}^{\mathbf{x}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{e}})\mathbf{N}^{\mathbf{x}T}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{e}})w_{\mathtt{g}}^{\mathtt{e}} - \sum_{\partial\Omega_i}\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^{\mathbf{x}}{}_{\boldsymbol{n}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})\mathbf{N}^{\mathbf{x}T}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{pu}]_i = \sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{e}}} \mathbf{N}^{\mathbf{x}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{e}})\boldsymbol{\nabla}\mathbf{N}^{\mathbf{x}T}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{e}})w_{\mathtt{g}}^{\mathtt{e}}$$

$$[\mathbf{a}_{\rho p}]_i = \sum_{\partial\Omega_i}\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^{\mathbf{x}T}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{L\hat{u}}]_i = -\sum_{\partial\Omega_i\setminus\Gamma_D}\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^{\mathbf{x}}{}_{\boldsymbol{n}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})\widehat{\mathbf{N}^{\mathbf{x}}}^T\big(\boldsymbol{x}(\xi_{\mathtt{g}}^{\mathtt{f}})\big)w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{u\hat{u}}]_i = -\sum_{\partial\Omega_i\setminus\Gamma_D}\tau_i\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^{\mathbf{x}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})\widehat{\mathbf{N}^{\mathbf{x}}}^T\big(\boldsymbol{x}(\xi_{\mathtt{g}}^{\mathtt{f}})\big)w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{p\hat{u}}]_i = \sum_{\partial\Omega_i\setminus\Gamma_D}\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^{\mathbf{x}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})\widehat{\mathbf{N}^{\mathbf{x}}}^T_{\boldsymbol{n}}\big(\boldsymbol{x}(\xi_{\mathtt{g}}^{\mathtt{f}})\big)w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{f}_L]_i = \sum_{\partial\Omega_i\cap\Gamma_D}\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^{\mathbf{x}}{}_{\boldsymbol{n}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})\boldsymbol{u}_D^T(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{f}_u]_i = -\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{e}}} \mathbf{N}^{\mathbf{x}}{}_{\mathtt{n}_{\mathtt{sd}}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{e}})\boldsymbol{s}^T(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{e}})w_{\mathtt{g}}^{\mathtt{e}} - \sum_{\partial\Omega_i\cap\Gamma_D}\tau_i\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^{\mathbf{x}}{}_{\mathtt{n}_{\mathtt{sd}}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})\boldsymbol{u}_D^T(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{f}_p]_i = \sum_{\partial\Omega_i\cap\Gamma_D}\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \mathbf{N}^{\mathbf{x}T}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})\boldsymbol{n}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})\boldsymbol{u}_D^T(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{\hat{u}L}]_i = \sum_{\partial\Omega_i\setminus\Gamma_D}\nu\sum_{\mathtt{g}=1}^{\mathtt{n}_{\mathtt{ip}}^{\mathtt{f}}} \widehat{\mathbf{N}^{\mathbf{x}}}\big(\boldsymbol{x}(\xi_{\mathtt{g}}^{\mathtt{f}})\big)\mathbf{N}^{\mathbf{x}T}_{\boldsymbol{n}}(\boldsymbol{x}_{\mathtt{g}}^{\mathtt{f}})w_{\mathtt{g}}^{\mathtt{f}}$$

$$[\mathbf{A}_{\hat{u}u}]_i = \sum_{\partial\Omega_i\setminus\Gamma_D} \tau_i \sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}} \widehat{\mathbf{N}^{\mathbf{x}}}\big(\boldsymbol{x}(\xi^{\mathtt{f}}_{\mathtt{g}})\big)\mathbf{N}^{\mathbf{x}T}(\boldsymbol{x}^{\mathtt{f}}_{\mathtt{g}})w^{\mathtt{f}}_{\mathtt{g}}$$

$$[\mathbf{A}_{\hat{u}p}]_i = \sum_{\partial\Omega_i\setminus\Gamma_D} \sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}} \widehat{\mathbf{N}^{\mathbf{x}}}_{\boldsymbol{n}}\big(\boldsymbol{x}(\xi^{\mathtt{f}}_{\mathtt{g}})\big)\mathbf{N}^{\mathbf{x}T}(\boldsymbol{x}^{\mathtt{f}}_{\mathtt{g}})w^{\mathtt{f}}_{\mathtt{g}}$$

$$[\mathbf{A}_{\hat{u}\hat{u}}]_i = -\sum_{\partial\Omega_i\setminus\Gamma_D} \tau_i \sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}} \widehat{\mathbf{N}^{\mathbf{x}}}\big(\boldsymbol{x}(\xi^{\mathtt{f}}_{\mathtt{g}})\big)\widehat{\mathbf{N}^{\mathbf{x}}}^T\big(\boldsymbol{x}(\xi^{\mathtt{f}}_{\mathtt{g}})\big)w^{\mathtt{f}}_{\mathtt{g}}$$

$$[\mathbf{f}_{\hat{u}}]_i = -\sum_{\partial\Omega_i\cap\Gamma^x_N} \sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}} \widehat{\mathbf{N}^{\mathbf{x}}}\big(\boldsymbol{x}(\xi^{\mathtt{f}}_{\mathtt{g}})\big)\boldsymbol{t}^T(\boldsymbol{x}^{\mathtt{f}}_{\mathtt{g}})w^{\mathtt{f}}_{\mathtt{g}}$$

$$[\boldsymbol{a}_{\rho\hat{u}}]_i = \sum_{\partial\Omega_i\setminus\Gamma_D} \sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}} \widehat{\mathbf{N}^{\mathbf{x}}}^T_{\boldsymbol{n}}\big(\boldsymbol{x}(\xi^{\mathtt{f}}_{\mathtt{g}})\big)w^{\mathtt{f}}_{\mathtt{g}}$$

$$[\mathbf{f}_{\rho}]_i = -\sum_{\partial\Omega_i\cap\Gamma_D} \sum_{\mathtt{g}=1}^{\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}} \boldsymbol{n}(\boldsymbol{x}^{\mathtt{f}}_{\mathtt{g}})\boldsymbol{u}^T_D(\boldsymbol{x}^{\mathtt{f}}_{\mathtt{g}})w^{\mathtt{f}}_{\mathtt{g}}$$

In the above expressions, $\boldsymbol{x}^{\mathtt{e}}_{\mathtt{g}}$ and $w^{\mathtt{e}}_{\mathtt{g}}$ are the $\mathtt{n}^{\mathtt{e}}_{\mathtt{ip}}$ integration points and weights defined in the physical space of the element and $\boldsymbol{x}^{\mathtt{f}}_{\mathtt{g}}$ and $w^{\mathtt{f}}_{\mathtt{g}}$ are the $\mathtt{n}^{\mathtt{f}}_{\mathtt{ip}}$ integration points and weights defined on the edge/face.