Hybridisable discontinuous Galerkin for second-order elliptic problems

Rubén Sevilla
Zienkiewicz Centre for Computational Engineering,
College of Engineering, Swansea University, Bay Campus, SA1 8EN, Wales, UK

DG Summer School - Barcelona - July 2017

Contents

1 Introduction 1

2 Problem statement 2

3 The Hybridizable Discontinuous Galerkin 2
- 3.1 Weak formulation . . . 3
- 3.2 Spatial discretisation . . . 4
- 3.3 Superconvergence by post-processing . . . 5
- 3.4 Computational aspects . . . 5
  - 3.4.1 Computation of elemental matrices . . . 5
  - 3.4.2 Storage of relevant information to solve the local problems . . . 7
  - 3.4.3 Accounting for isoparametric curved elements . . . 7
  - 3.4.4 Implementation details . . . 7

4 Numerical Example 9

5 Exercise 9

1 Introduction

These notes are aimed to complement the computer lab on hybridisable discontinuous Galerkin for second-order elliptic problems. They are accompanied by a Matlab academic code for the solution of the Poisson equation in two dimensions using triangular elements. All the content of these notes has been extracted from [1], where a more detailed and general description of the HDG method for second-order elliptic problems can be found.

2 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 form of the model problem is

$$\begin{cases} -\boldsymbol{\nabla}\cdot\boldsymbol{\nabla} u = f & \text{in } \Omega, \\ u = u_D & \text{on } \partial\Omega, \end{cases} \tag{1}$$

where f is a source term.

The domain Ω is assumed partitioned in $\mathtt{n_{el}}$ disjoint subdomains Ω_e

$$\overline{\Omega} = \bigcup_{i=1}^{\mathtt{n_{el}}} \overline{\Omega}_e, \quad \Omega_e \cap \Omega_l = \emptyset \text{ for } e \neq l,$$

with boundaries $\partial\Omega_e$, which define an internal interface Γ

$$\Gamma := \left[\bigcup_{e=1}^{\mathtt{n_{el}}} \partial\Omega_e\right] \setminus \partial\Omega. \tag{2}$$

An equivalent strong form of the second-order elliptic problem can be written in mixed form over the *broken* computational domain as

$$\begin{cases} \boldsymbol{\nabla}\cdot\boldsymbol{q} = f & \text{in } \Omega_e, \text{ and for } e = 1,\dots,\mathtt{n_{el}}, \\ \boldsymbol{q} + \boldsymbol{\nabla} u = \boldsymbol{0} & \text{in } \Omega_e, \text{ and for } e = 1,\dots,\mathtt{n_{el}}, \\ u = u_D & \text{on } \partial\Omega, \\ [\![u\boldsymbol{n}]\!] = \boldsymbol{0} & \text{on } \Gamma, \\ [\![\boldsymbol{n}\cdot\boldsymbol{q}]\!] = 0 & \text{on } \Gamma. \end{cases} \tag{3}$$

where the two last equations correspond to the imposition of the continuity of the primal variable u and the normal fluxes respectively along the internal interface Γ. Note that the *jump* $[\![\cdot]\!]$ operator has been introduced following the definition by [2]. That is, along each portion of the interface Γ it sums the values from the left and right of say, Ω_e and Ω_l, namely

$$[\![\odot]\!] = \odot_e + \odot_l.$$

3 The Hybridizable Discontinuous Galerkin

The HDG formulation solves problem (3) in two phases, see [3–8]. First, an element-by-element problem is defined with $(\boldsymbol{q}, u)$ as unknowns. The local problem determines $\boldsymbol{q}_e := \boldsymbol{q}|_{\Omega_e}$ and $u_e := u|_{\Omega_e}$ for $e = 1,\dots,\mathtt{n_{el}}$ with a new variable $\hat{u}$ along the interface Γ acting as a Dirichlet boundary condition, namely

$$\begin{cases} \boldsymbol{\nabla}\cdot\boldsymbol{q}_e = f & \text{in } \Omega_e, \\ \boldsymbol{q}_e + \boldsymbol{\nabla} u_e = \boldsymbol{0} & \text{in } \Omega_e, \\ u_e = u_D & \text{on } \partial\Omega_e \cap \partial\Omega, \\ u_e = \hat{u} & \text{on } \partial\Omega_e \setminus \partial\Omega, \end{cases} \tag{4}$$

for $e = 1, \dots, \mathtt{n_{el}}$. Note that this approach assumes $\hat{u}$ given. In each element Ω_e this problem produces an element-by-element solution $\boldsymbol{q}_e$ and u_e as a function of the unknown $\hat{u}$.

Second, a global problem is defined to determine $\hat{u}$. It corresponds to the imposition of the so-called *transmission conditions*, see [3]

$$[\![\boldsymbol{n} \cdot \boldsymbol{q}]\!] = 0 \quad \text{on } \Gamma, \tag{5}$$

Note that the continuity of u along Γ is automatically satisfied because $u = \hat{u}$ on Γ, as imposed by the local problems (4), and $\hat{u}$ is unique for adjacent elements.

3.1 Weak formulation

The weak formulation for each element equivalent to (4) is, for $e = 1, \dots, \mathtt{n_{el}}$,

$$\begin{aligned} -(\boldsymbol{\nabla} v, \boldsymbol{q}_e)_{\Omega_e} + < v, \boldsymbol{n}_e \cdot \widehat{\boldsymbol{q}}_e >_{\partial\Omega_e} &= (v, f)_{\Omega_e} \\ -(\boldsymbol{w}, \boldsymbol{q}_e)_{\Omega_e} + (\boldsymbol{\nabla} \cdot \boldsymbol{w}, u_e)_{\Omega_e} &= < \boldsymbol{n}_e \cdot \boldsymbol{w}, u_D >_{\partial\Omega_e \cap \partial\Omega} + < \boldsymbol{n}_e \cdot \boldsymbol{w}, \hat{u} >_{\partial\Omega_e \setminus \partial\Omega}, \end{aligned}$$

where the numerical traces of the fluxes $\widehat{\boldsymbol{q}}_e$ are defined element-by-element as

$$\boldsymbol{n}_e \cdot \widehat{\boldsymbol{q}}_e := \begin{cases} \boldsymbol{n}_e \cdot \boldsymbol{q}_e + \tau_e (u_e - u_D) & \text{on } \partial\Omega_e \cap \partial\Omega, \\ \boldsymbol{n}_e \cdot \boldsymbol{q}_e + \tau_e (u_e - \hat{u}) & \text{elsewhere}, \end{cases} \tag{6}$$

with τ_e being a stabilization parameter, see for instance [3–8] and $(\cdot,\cdot)_D$ and $< \cdot,\cdot >_S$ denote the $\mathcal{L}_2$ scalar product in any subdomain $D \subset \Omega$ and in any domain $S \subset \Gamma \cup \partial\Omega$ respectively.

With the definition of the numerical fluxes given by (6), the weak problem becomes

$$\begin{aligned} < v, \tau_e\, u_e >_{\partial\Omega_e} - (\boldsymbol{\nabla} v, \boldsymbol{q}_e)_{\Omega_e} + < v, \boldsymbol{n}_e \cdot \boldsymbol{q}_e >_{\partial\Omega_e} & \\ = (v, f)_{\Omega_e} + < v, \tau_e\, u_D >_{\partial\Omega_e \cap \partial\Omega} + < v, \tau_e\, \hat{u} >_{\partial\Omega_e \setminus \partial\Omega}, & \end{aligned} \tag{7a}$$

$$\begin{aligned} -(\boldsymbol{w}, \boldsymbol{q}_e)_{\Omega_e} + (\boldsymbol{\nabla} \cdot \boldsymbol{w}, u_e)_{\Omega_e} & \\ = < \boldsymbol{n}_e \cdot \boldsymbol{w}, u_D >_{\partial\Omega_e \cap \partial\Omega} + < \boldsymbol{n}_e \cdot \boldsymbol{w}, \hat{u} >_{\partial\Omega_e \setminus \partial\Omega} . & \end{aligned} \tag{7b}$$

The weak form (7) for the *local problem* is equivalent to the strong form described by (4).

To symmetrize the local problem, an integration by parts is performed on the second term of the l.h.s. in (7a), leading to

$$\begin{aligned} < v, \tau_e\, u_e >_{\partial\Omega_e} + (v, \boldsymbol{\nabla} \cdot \boldsymbol{q}_e)_{\Omega_e} & \\ = (v, f)_{\Omega_e} + < v, \tau_e\, u_D >_{\partial\Omega_e \cap \partial\Omega} + < v, \tau_e\, \hat{u} >_{\partial\Omega_e \setminus \partial\Omega}, & \end{aligned} \tag{8a}$$

$$\begin{aligned} (\boldsymbol{\nabla} \cdot \boldsymbol{w}, u_e)_{\Omega_e} - (\boldsymbol{w}, \boldsymbol{q}_e)_{\Omega_e} & \\ = < \boldsymbol{n}_e \cdot \boldsymbol{w}, u_D >_{\partial\Omega_e \cap \partial\Omega} + < \boldsymbol{n}_e \cdot \boldsymbol{w}, \hat{u} >_{\partial\Omega_e \setminus \partial\Omega} . & \end{aligned} \tag{8b}$$

Once the weak form for the *local problem* is presented, the *global problem* (5) is of interest. The weak form equivalent to (5) is simply

$$\sum_{i=1}^{\mathtt{n_{el}}} < \mu, \boldsymbol{n}_e \cdot \widehat{\boldsymbol{q}}_e >_{\partial\Omega_e \setminus \partial\Omega} = 0.$$

Then, replacing (6) in the previous equation results in the global weak problem

$$\sum_{i=1}^{\mathtt{n_{el}}} \Big\{ < \mu, \tau_e \, u_e >_{\partial\Omega_e \setminus \partial\Omega} + < \mu, \boldsymbol{n}_e \cdot \boldsymbol{q}_e >_{\partial\Omega_e \setminus \partial\Omega} - < \mu, \tau_e \, \hat{u} >_{\partial\Omega_e \setminus \partial\Omega} \Big\} = 0. \tag{9}$$

Note that both u_e and $\boldsymbol{q}_e$ are known functions of $\hat{u}$.

3.2 Spatial discretisation

The variables u and $\boldsymbol{q}$ are approximated in a reference element as

$$u(\boldsymbol{\xi}) \simeq u^h(\boldsymbol{\xi}) = \sum_{j=1}^{\mathtt{n_{en}}} \mathrm{u}_j N_j(\boldsymbol{\xi}), \tag{10a}$$

$$\boldsymbol{q}(\boldsymbol{\xi}) \simeq \boldsymbol{q}^h(\boldsymbol{\xi}) = \sum_{j=1}^{\mathtt{n_{en}}} \mathbf{q}_j N_j(\boldsymbol{\xi}), \tag{10b}$$

where u_j and $\mathbf{q}_j$ denote the value of the corresponding variable at node j, N_j is the shape function associated with node j and $\mathtt{n_{en}}$ is the total number of nodes on the element.

Similarly, the hybrid variable $\hat{u}$ is approximated in a reference face as

$$\hat{u}(\boldsymbol{\zeta}) \simeq \hat{u}^h(\boldsymbol{\zeta}) = \sum_{j=1}^{\mathtt{n_{fn}}} \hat{\mathrm{u}}_j \hat{N}_j(\boldsymbol{\zeta}). \tag{11}$$

where $\hat{\mathrm{u}}_j$ denotes the value of $\hat{u}$ at node j, $\hat{N}_j$ is the shape function associated with node j and $\mathtt{n_{fn}}$ is the total number of nodes on the face.

Introducing the approximation of u, $\boldsymbol{q}$ and $\hat{u}$ given in Equations (10) and (11) in the weak form of the local problem given by Equation (8) gives rise to the following system of equations for each element Ω_e (i.e., for $e = 1, \ldots, \mathtt{n_{el}}$)

$$\begin{bmatrix} \mathbf{A}_{uu} & \mathbf{A}_{uq} \\ \mathbf{A}_{uq}^T & \mathbf{A}_{qq} \end{bmatrix}_e \begin{Bmatrix} \mathbf{u}_e \\ \mathbf{q}_e \end{Bmatrix} = \begin{Bmatrix} \mathbf{f}_u \\ \mathbf{f}_q \end{Bmatrix}_e + \begin{bmatrix} \mathbf{A}_{u\hat{u}} \\ \mathbf{A}_{q\hat{u}} \end{bmatrix}_e \hat{\mathbf{u}}_e, \tag{12a}$$

Similarly, introducing the approximation of u, $\boldsymbol{q}$ and $\hat{u}$ given in Equations (10) and (11) in the weak form of the gloal problem given by Equation (9) produce the following system of equations

$$\sum_{i=1}^{\mathtt{n_{el}}} \Big\{ \begin{bmatrix} \mathbf{A}_{u\hat{u}}^T & \mathbf{A}_{q\hat{u}}^T \end{bmatrix}_e \begin{Bmatrix} \mathbf{u}_e \\ \mathbf{q}_e \end{Bmatrix} + [\mathbf{A}_{\hat{u}\hat{u}}]_e \, \hat{\mathbf{u}}_e \Big\} = \mathbf{0}. \tag{12b}$$

The elemental matrices appearing in (12) are detailed in Section 3.4.

After replacing the solution of the local problem (12a) in (12b), the global problem becomes

$$\widehat{\mathbf{K}}\hat{\mathbf{u}} = \hat{\mathbf{f}}, \tag{13}$$

with

$$\widehat{\mathbf{K}} = \mathop{\boldsymbol{\mathsf{A}}}_{i=1}^{\mathtt{n_{el}}} \begin{bmatrix} \mathbf{A}_{u\hat{u}}^T & \mathbf{A}_{q\hat{u}}^T \end{bmatrix}_e \begin{bmatrix} \mathbf{A}_{uu} & \mathbf{A}_{uq} \\ \mathbf{A}_{uq}^T & \mathbf{A}_{qq} \end{bmatrix}_e^{-1} \begin{bmatrix} \mathbf{A}_{u\hat{u}} \\ \mathbf{A}_{q\hat{u}} \end{bmatrix}_e + \begin{bmatrix} \mathbf{A}_{\hat{u}\hat{u}} \end{bmatrix}_e \tag{14a}$$

and

$$\hat{\mathbf{f}} = -\mathop{\boldsymbol{\mathsf{A}}}_{i=1}^{\mathtt{n_{el}}} \begin{bmatrix} \mathbf{A}_{u\hat{u}}^T & \mathbf{A}_{q\hat{u}}^T \end{bmatrix}_e \begin{bmatrix} \mathbf{A}_{uu} & \mathbf{A}_{uq} \\ \mathbf{A}_{uq}^T & \mathbf{A}_{qq} \end{bmatrix}_e^{-1} \begin{Bmatrix} \mathbf{f}_u \\ \mathbf{f}_q \end{Bmatrix}_e . \tag{14b}$$

3.3 Superconvergence by post-processing

The postprocessed solution is computed by performing a postprocessing similar to the projection traditionally employed in the mixed method by [9], see also [10]. More precisely, the superconvergent postprocessed solution is obtained by solving the following problem in each element

$$\left\{ \begin{aligned} -\boldsymbol{\nabla} \cdot \boldsymbol{\nabla} u_\star &= -\boldsymbol{\nabla} \cdot \mathbf{q}^h && \text{in } \Omega_e, \\ \boldsymbol{n} \cdot \boldsymbol{\nabla} u_\star &= \boldsymbol{n} \cdot \mathbf{q}^h && \text{on } \partial\Omega_e, \end{aligned} \right. \tag{15}$$

with the additional solvability constraint

$$(u_\star, 1)_{\Omega_e} = (u^h, 1)_{\Omega_e},$$

for $e = 1, \dots, \mathtt{n_{el}}$, see [3, 4, 11].

3.4 Computational aspects

3.4.1 Computation of elemental matrices

The interpolation functions u^h and $\boldsymbol{q}^h$ are defined in a reference element as detailed in Equation (10), with local coordinates $\boldsymbol{\xi}$. The isoparametric transformation is used to relate local and Cartesian coordinates, namely

$$\begin{aligned} \boldsymbol{\varphi}\ : \hat{\Omega} \subset \mathbb{R}^{\mathtt{n_{sd}}} &\longrightarrow \Omega_e \subset \mathbb{R}^{\mathtt{n_{sd}}} \\ \boldsymbol{\xi} &\longmapsto \boldsymbol{\varphi}(\boldsymbol{\xi}) := \sum_{j=1}^{\mathtt{n_{en}}} \boldsymbol{x}_j N_j(\boldsymbol{\xi}), \end{aligned} \tag{16}$$

where $\boldsymbol{x}_i$ denote the elemental nodal coordinates.

Similarly, the interpolation function $\hat{u}^h$ of Equation (11) is defined in a reference face, with local coordinates $\boldsymbol{\zeta}$. The isoparametric transformation used to relate local and Cartesian coordinates is

$$\begin{aligned} \boldsymbol{\psi} \; : \hat{\Gamma} \subset \mathbb{R}^{\mathtt{n_{sd}}-1} &\longrightarrow \Gamma_e^f \subset \mathbb{R}^{\mathtt{n_{sd}}} \\ \boldsymbol{\zeta} &\longmapsto \boldsymbol{\psi}(\boldsymbol{\zeta}) := \sum_{j=1}^{\mathtt{n_{fn}}} \boldsymbol{x}_j \hat{N}_j(\boldsymbol{\zeta}), \end{aligned} \tag{17}$$

where $\boldsymbol{x}_j$ denote the face nodal coordinates.

The following compact form of the interpolation functions is introduced

$$\mathbf{N} = \begin{bmatrix} N_1 & N_2 & \dots & N_{\mathtt{n_{en}}} \end{bmatrix}^T, \quad \widehat{\mathbf{N}} = \begin{bmatrix} \widehat{N}_1 & \widehat{N}_2 & \dots & \widehat{N}_{\mathtt{n_{fn}}} \end{bmatrix}^T,$$

$$\mathbf{N}_{\boldsymbol{n}} = \begin{bmatrix} N_1\boldsymbol{n} & N_2\boldsymbol{n} & \dots & N_{\mathtt{n_{en}}}\boldsymbol{n} \end{bmatrix}^T, \quad \widehat{\mathbf{N}}_{\boldsymbol{n}} = \begin{bmatrix} \widehat{N}_1\boldsymbol{n} & \widehat{N}_2\boldsymbol{n} & \dots & \widehat{N}_{\mathtt{n_{fn}}}\boldsymbol{n} \end{bmatrix}^T,$$

$$\boldsymbol{\nabla}\mathbf{N} = \begin{bmatrix} \left(\boldsymbol{J}^{-1}\boldsymbol{\nabla} N_1\right)^T & \left(\boldsymbol{J}^{-1}\boldsymbol{\nabla} N_2\right)^T & \dots & \left(\boldsymbol{J}^{-1}\boldsymbol{\nabla} N_{\mathtt{n_{en}}}\right)^T \end{bmatrix}^T,$$

$$\mathbf{N}_{\mathtt{n_{sd}}} = \begin{bmatrix} N_1\boldsymbol{I}_{\mathtt{n_{sd}}} & N_2\boldsymbol{I}_{\mathtt{n_{sd}}} & \dots & N_{\mathtt{n_{en}}}\boldsymbol{I}_{\mathtt{n_{sd}}} \end{bmatrix}^T,$$

where $\boldsymbol{n} = (n_1, \dots, n_{\mathtt{n_{sd}}})$ denotes the outward unit normal vector to an edge/face, $\boldsymbol{J}$ is the Jacobian of the isoparametric transformation and $\boldsymbol{I}_{\mathtt{n_{sd}}}$ is the identity matrix of dimension $\mathtt{n_{sd}}$.

The different matrices appearing in (12), computed for each element $e = 1, \dots, \mathtt{n_{el}}$, can be expressed as

$$[\mathbf{A}_{uu}]_e = \sum_{\partial\Omega_e} \tau_e \sum_{\mathrm{g}=1}^{\mathtt{m_{ip}}} \mathbf{N}(\boldsymbol{\zeta}_{\mathrm{g}})\, \mathbf{N}^T(\boldsymbol{\zeta}_{\mathrm{g}}) \|\boldsymbol{J}_{\boldsymbol{\psi}}\| \varpi_{\mathrm{g}},$$

$$[\mathbf{A}_{qq}]_e = -\sum_{\mathrm{g}=1}^{\mathtt{n_{ip}}} \mathbf{N}_{\mathtt{n_{sd}}}(\boldsymbol{\xi}_{\mathrm{g}})\, \mathbf{N}_{\mathtt{n_{sd}}}^T(\boldsymbol{\xi}_{\mathrm{g}}) |\boldsymbol{J}_{\boldsymbol{\varphi}}| \omega_{\mathrm{g}},$$

$$[\mathbf{A}_{uq}]_e = \sum_{\mathrm{g}=1}^{\mathtt{n_{ip}}} \mathbf{N}(\boldsymbol{\xi}_{\mathrm{g}})\, \boldsymbol{\nabla}\mathbf{N}^T(\boldsymbol{\xi}_{\mathrm{g}}) |\boldsymbol{J}_{\boldsymbol{\varphi}}| \omega_{\mathrm{g}},$$

$$[\mathbf{f}_u]_e = \sum_{\mathrm{g}=1}^{\mathtt{n_{ip}}} \mathbf{N}(\boldsymbol{\xi}_{\mathrm{g}})\, f\big(\boldsymbol{\varphi}(\boldsymbol{\xi}_{\mathrm{g}})\big) |\boldsymbol{J}_{\boldsymbol{\varphi}}| \omega_{\mathrm{g}} + \sum_{\partial\Omega_e \cap \partial\Omega} \tau_e \sum_{\mathrm{g}=1}^{\mathtt{m_{ip}}} \mathbf{N}(\boldsymbol{\zeta}_{\mathrm{g}})\, u_D\big(\boldsymbol{\psi}(\boldsymbol{\xi}_{\mathrm{g}})\big) \|\boldsymbol{J}_{\boldsymbol{\psi}}\| \varpi_{\mathrm{g}},$$

$$[\mathbf{f}_q]_e = \sum_{\partial\Omega_e \cap \partial\Omega} \sum_{\mathrm{g}=1}^{\mathtt{m_{ip}}} \mathbf{N}_{\boldsymbol{n}}(\boldsymbol{\zeta}_{\mathrm{g}})\, u_D\big(\boldsymbol{\psi}(\boldsymbol{\zeta}_{\mathrm{g}})\big) \|\boldsymbol{J}_{\boldsymbol{\psi}}\| \varpi_{\mathrm{g}},$$

$$[\mathbf{A}_{u\hat{u}}]_e = \sum_{\partial\Omega_e \setminus \partial\Omega} \tau_e \sum_{\mathrm{g}=1}^{\mathtt{m_{ip}}} \mathbf{N}(\boldsymbol{\zeta}_{\mathrm{g}})\, \widehat{\mathbf{N}}^T(\boldsymbol{\zeta}_{\mathrm{g}}) \|\boldsymbol{J}_{\boldsymbol{\psi}}\| \varpi_{\mathrm{g}},$$

$$[\mathbf{A}_{q\hat{u}}]_e = \sum_{\partial\Omega_e\setminus\partial\Omega} \sum_{\mathrm{g}=1}^{\mathtt{m_{ip}}} \mathbf{N}_{\boldsymbol{n}}(\boldsymbol{\zeta}_{\mathrm{g}})\,\widehat{\mathbf{N}}^T(\boldsymbol{\zeta}_{\mathrm{g}})\|\boldsymbol{J}_\psi\|\varpi_{\mathrm{g}},$$

and

$$[\mathbf{A}_{\hat{u}\hat{u}}]_e = -\sum_{\partial\Omega_e\setminus\partial\Omega} \tau_e \sum_{\mathrm{g}=1}^{\mathtt{m_{ip}}} \widehat{\mathbf{N}}_{\boldsymbol{n}}(\boldsymbol{\zeta}_{\mathrm{g}})\,\widehat{\mathbf{N}}^T(\boldsymbol{\zeta}_{\mathrm{g}})\|\boldsymbol{J}_\psi\|\varpi_{\mathrm{g}}$$

In the above expressions, $\{\boldsymbol{\xi}_g\}_{g=1}^{\mathtt{n_{ip}}}$ and $\{\omega\}_{g=1}^{\mathtt{n_{ip}}}$ are integration points and weights defined on the reference element and $\{\boldsymbol{\zeta}_g\}_{g=1}^{\mathtt{m_{ip}}}$ and $\{\varpi\}_{g=1}^{\mathtt{m_{ip}}}$ are the integration points and weights defined on the reference edge/face.

3.4.2 Storage of relevant information to solve the local problems

From a computational point of view, some auxiliary vectors are defined. Equations (12a) can be written as

$$\begin{Bmatrix} \mathbf{u}_e \\ \mathbf{q}_e \end{Bmatrix} = \begin{Bmatrix} \mathbf{z}_u^f \\ \mathbf{z}_q^f \end{Bmatrix}_e + \begin{bmatrix} \mathbf{Z}_u^{\hat{u}} \\ \mathbf{Z}_q^{\hat{u}} \end{bmatrix}_e \hat{\mathbf{u}}_e \tag{18}$$

where

$$\begin{Bmatrix} \mathbf{z}_u^f \\ \mathbf{z}_q^f \end{Bmatrix}_e = \begin{bmatrix} \mathbf{A}_{uu} & \mathbf{A}_{uq} \\ \mathbf{A}_{uq}^T & \mathbf{A}_{qq} \end{bmatrix}_e^{-1} \begin{Bmatrix} \mathbf{f}_u \\ \mathbf{f}_q \end{Bmatrix}_e \quad \text{and} \quad \begin{bmatrix} \mathbf{Z}_u^{\hat{u}} \\ \mathbf{Z}_q^{\hat{u}} \end{bmatrix}_e = \begin{bmatrix} \mathbf{A}_{uu} & \mathbf{A}_{uq} \\ \mathbf{A}_{uq}^T & \mathbf{A}_{qq} \end{bmatrix}_e^{-1} \begin{bmatrix} \mathbf{A}_{u\hat{u}} \\ \mathbf{A}_{q\hat{u}} \end{bmatrix}_e$$

Then, (18) is replaced in (12b), which induces the same system of equations (13) but the matrix and vector defined by (14) are computed as follows:

$$\widehat{\mathbf{K}} = \mathop{\mathbf{\Large A}}_{e=1}^{\mathtt{n_{el}}} \begin{bmatrix} \mathbf{A}_{u\hat{u}}^T & \mathbf{A}_{q\hat{u}}^T \end{bmatrix}_e \begin{bmatrix} \mathbf{Z}_u^{\hat{u}} \\ \mathbf{Z}_q^{\hat{u}} \end{bmatrix}_e + [\mathbf{A}_{\hat{u}\hat{u}}]_e \quad \text{and} \quad \hat{\mathbf{f}} = -\mathop{\mathbf{\Large A}}_{e=1}^{\mathtt{n_{el}}} \begin{bmatrix} \mathbf{A}_{u\hat{u}}^T & \mathbf{A}_{q\hat{u}}^T \end{bmatrix}_e \begin{Bmatrix} \mathbf{z}_u^f \\ \mathbf{z}_q^f \end{Bmatrix}_e .$$

3.4.3 Accounting for isoparametric curved elements

It is worth noting that the code provided do not exploit the fact that the Jacobian for triangular elements with planar edges is constant as done in the code provided for solving hyperbolic problems. Being a stationary problem the computational cost is less critical than in the transient problems solved in the previous lab but still this could be introduced as a way to accelerate the code. More important, this means that the code provided for solving the Poisson equation also works, with no modification, if the mesh contains curved elements. In fact, this can also be used to extend the code provided for hyperbolic problems to handle curved elements, see [12] for more details.

3.4.4 Implementation details

The code provided use long names for many of the key variables in order to facilitate its understanding. This section provides a brief description of the most important variables.

The structure `referenceElement` stores all the information related to the reference element. It contains:

- `IPcoordinates`: Coordinates of the element integration points. Dimension $\mathtt{n_{ip}} \times \mathtt{n_{sd}}$.
- `IPweights`: Weights of the element integration points. Dimension $\mathtt{n_{ip}} \times 1$.
- `N`: Element shape functions evaluated at the integration points. Dimension $\mathtt{n_{ip}} \times \mathtt{n_{en}}$.
- `Nxi,Neta`: Derivatives of the element shape functions evaluated at the integration points. Dimension $\mathtt{n_{ip}} \times \mathtt{n_{en}}$.
- `IPcoordinates1d`: Coordinates of the face integration points. Dimension $\mathtt{m_{ip}} \times (\mathtt{n_{sd}} - 1)$.
- `IPweights1d`: Weights of the face integration points. Dimension $1 \times \mathtt{m_{ip}}$.
- `N1d`: Face shape functions evaluated at the integration points. Dimension $\mathtt{m_{ip}} \times \mathtt{n_{fn}}$.
- `faceNodes`: Array of dimension $\mathtt{n_{fa}} \times \mathtt{n_{fn}}$. Row f contains the nodes of the element in the f-th face.
- `innerNodes`: List of interior nodes.
- `faceNodes1d`: Local node numbering for the face. Dimension $1 \times \mathtt{n_{fn}}$.
- `NodesCoord`: Coordinates of the element nodes used for interpolation. Dimension $\mathtt{n_{en}} \times \mathtt{n_{sd}}$.
- `NodesCoord1d`: Coordinates of the face nodes used for interpolation. Dimension $\mathtt{n_{fn}} \times (\mathtt{n_{sd}} - 1)$.
- `degree`: Degree of approximation (k).

The structure `infoFaces` contains `intFaces` and `extFaces`, which were also used in the DG solver for hyperbolic problems. `intFaces` is an array of dimension $\mathtt{n^i_{fa}} \times 5$, where $\mathtt{n^i_{fa}}$ is the total number of interior faces. In two dimensions only the first four columns are of interest and they contain, for a given internal face (i.e. for a given row of `intFaces`)

- The first element sharing this face.
- The local face number for the first element.
- The second element sharing this face.
- The local face number for the second element.

Similarly, `extFaces` is an array of dimension $\mathtt{n^e_{fa}} \times 2$, where $\mathtt{n^e_{fa}}$ is the total number of exterior faces. For a given external face (i.e. for a given row of `extFaces`)

- The element sharing this face.

- The local face number .

The array `F` contains the reciprocal information of `infoFaces`. It is an array of size $\mathtt{n_{el}} \times \mathtt{n_{fa}}$. The terms $\mathtt{F}_{e,f}$ contains an integer number r that

- if less or equal than $\mathtt{n^i_{fa}}$, indicates that the face f of elemetn e is internal and its information can be found in row r of `intFaces`.
- otherwise, indicates that the face f of elemetn e is external and its information can be found in row $r - \mathtt{n^i_{fa}}$ of `extFaces`.

Figure 1 shows the flowchart of the DG code for solving hyperbolic problems.

4 Numerical Example

To illustrate the concepts presented in these notes, the model problem (1) is solved in $\Omega := [0,1]^2$ where the source and boundary conditions are taken such that the analytical solution is given by

$$u(x,y) = 4y^2 - 4\lambda^2 y \exp(-\lambda y)\cos(6\pi x) + \lambda \exp(-2\lambda y),$$

where λ is a parameter.

The first example involves the solution of the model problem with a value of λ=4. An extremely coarse mesh, with only eight elements, is considered, as shown in the left plot of Figure 2. The centre plot of Figure 2 depicts the degrees of freedom used in an HDG computation with approximation order k=6. The black dots on the triangles denote the nodes used to build the polynomial approximation of the primal and dual solutions, u^h and $\boldsymbol{q}^h$ respectively. The red lines are the set of edges Γ where the trace of the solution is approximated and the dots over these lines are the nodes used to build the polynomial approximation of $\hat{u}$. Note that there are no $\hat{u}^h$ unknowns along the boundary. The numerical solution computed with a polynomial approximation of degree k=6 is depicted in the right plot of Figure 2, showing both the approximation of the solution in the element interiors and the approximation of the trace of the solution on Γ.

Next, the model problem is considered with a value of λ=10. Figure 3 shows the numerical solution computed on a finer mesh, with 32 elements, and with a degree of approximation k=4 and k=5. It is worth noting how the jump of the solution on the element interfaces decreases as the degree of the approximation increases, suggesting the higher accuracy of the solution computed with k=5. Finally, the postprocessed solution is also shown, illustrated the gain in accuracy due to the element by element postprocess described in Section 3.3

5 Exercise

Modify the computer program given to include Neumman boundary conditions. Check [1] for the details about the HDG formulation with Neumann boundary conditions.

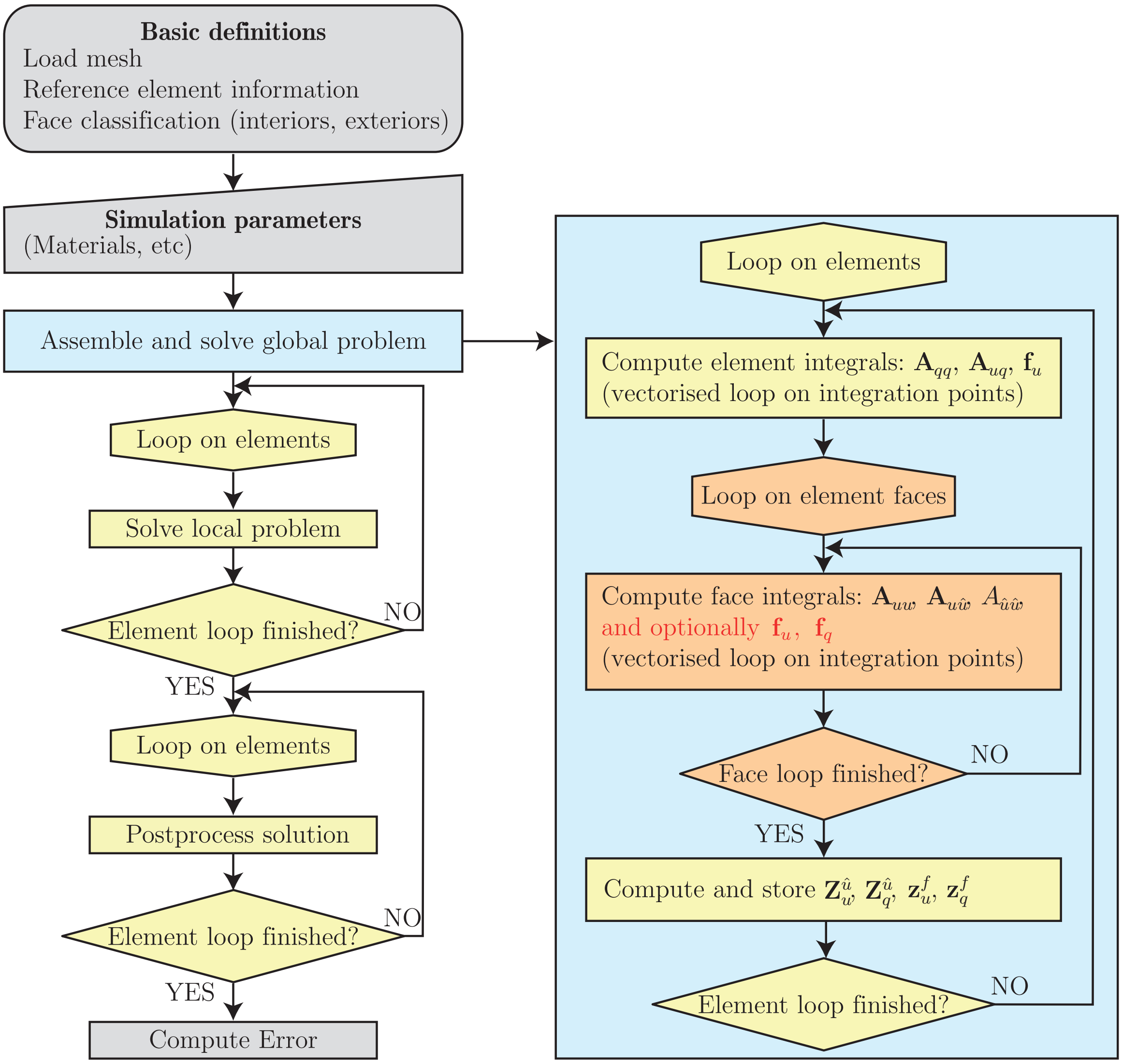
Basic definitions
Load mesh
Reference element information
Face classification (interiors, exteriors)
Simulation parameters
(Materials, etc)
Assemble and solve global problem
Loop on elements
Solve local problem
Element loop finished?
NO
YES
Loop on elements
Postprocess solution
Element loop finished?
NO
YES
Compute Error
Loop on elements
Compute element integrals: $\mathbf{A}_{qq}$, $\mathbf{A}_{uq}$, $\mathbf{f}_u$
(vectorised loop on integration points)
Loop on element faces
Compute face integrals: $\mathbf{A}_{uw}$, $\mathbf{A}_{u\hat{w}}$, $A_{\hat{u}\hat{w}}$
and optionally $\mathbf{f}_u$, $\mathbf{f}_q$
(vectorised loop on integration points)
Face loop finished?
NO
YES
Compute and store $\mathbf{Z}_w^{\hat{u}}$, $\mathbf{Z}_q^{\hat{u}}$, $\mathbf{z}_u^f$, $\mathbf{z}_q^f$
Element loop finished?
NO

Figure 1: Flowchart of the HDG code for solving second-order elliptic problems.

References

[1] R. Sevilla and A. Huerta, "Tutorial on hybridizable discontinuous Galerkin (HDG) for second-order elliptic problems," in *Advanced Finite Element Technologies*, pp. 105–129, Springer, 2016.

[2] A. Montlaur, S. Fernández-Méndez, and A. Huerta, "Discontinuous Galerkin methods for the Stokes equations using divergence-free approximations," *Int. J. Numer. Methods Fluids*, vol. 57, no. 9, pp. 1071–1092, 2008.

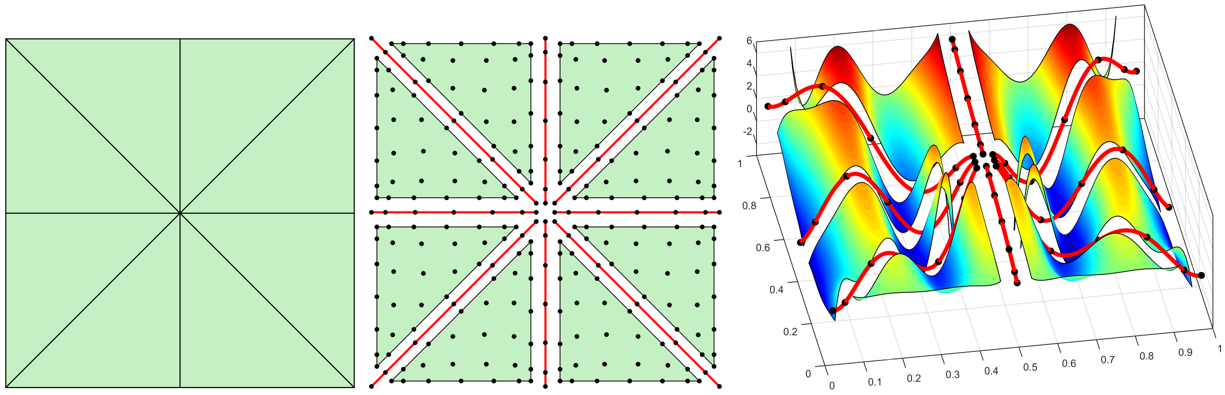

Figure 2: Coarse mesh of the domain Ω (left), illustration of the degrees of freedom employed in an HDG computation with k=6 (centre) and solution or k=6 on the mesh of Figure 2 showing both u^h in the element interiors and $\hat{u}^h$ on the edges Γ.

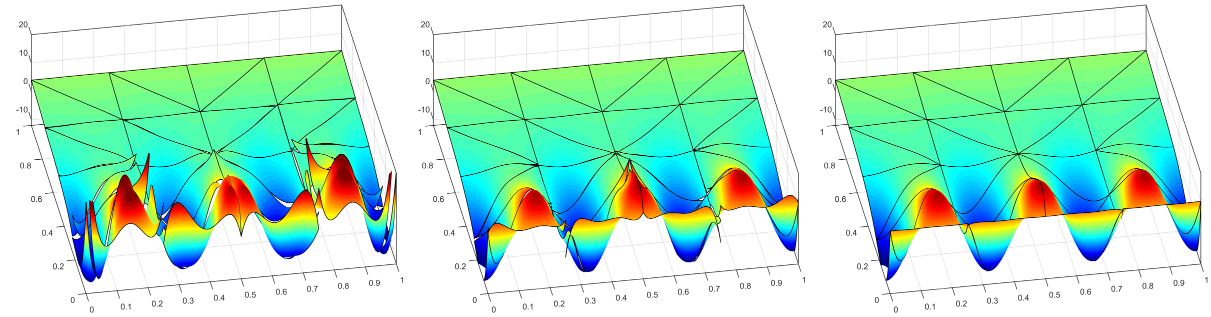

Figure 3: Model problem solution for k=4 (left) and k=5 (centre) showing the improvement induced by an increase on the degree of approximation. The postprocessed solution with k=5 is shown on the right plot.

[3] B. Cockburn, J. Gopalakrishnan, and R. Lazarov, "Unified hybridization of discontinuous Galerkin, mixed, and continuous Galerkin methods for second order elliptic problems," *SIAM J. Numer. Anal.*, vol. 47, no. 2, pp. 1319–1365, 2009.

[4] B. Cockburn, B. Dong, and J. Guzmán, "A superconvergent LDG-hybridizable Galerkin method for second-order elliptic problems," *Math. Comp.*, vol. 77, no. 264, pp. 1887–1916, 2008.

[5] N. C. Nguyen, J. Peraire, and B. Cockburn, "An implicit high-order hybridizable discontinuous Galerkin method for linear convection-diffusion equations," *J. Comput. Phys.*, vol. 228, no. 9, pp. 3232–3254, 2009.

[6] N. C. Nguyen, J. Peraire, and B. Cockburn, "An implicit high-order hybridizable discontinuous Galerkin method for nonlinear convection-diffusion equations," *J. Comput. Phys.*, vol. 228, no. 23, pp. 8841–8855, 2009.

[7] N. Nguyen, J. Peraire, and B. Cockburn, “A hybridizable discontinuous Galerkin method for Stokes flow,” *Comput. Methods Appl. Mech. Eng.*, vol. 199, no. 9-12, pp. 582–597, 2010.

[8] N. C. Nguyen, J. Peraire, and B. Cockburn, “An implicit high-order hybridizable discontinuous Galerkin method for the incompressible Navier-Stokes equations,” *J. Comput. Phys.*, vol. 230, no. 4, pp. 1147–1170, 2011.

[9] P.-A. Raviart and J. M. Thomas, “A mixed finite element method for 2nd order elliptic problems,” in *Mathematical aspects of finite element methods (Proc. Conf., Consiglio Naz. delle Ricerche (C.N.R.), Rome, 1975)*, pp. 292–315. Lecture Notes in Math., Vol. 606, Springer, Berlin, 1977.

[10] D. N. Arnold and F. Brezzi, “Mixed and nonconforming finite element methods: implementation, postprocessing and error estimates,” *RAIRO Modél. Math. Anal. Numér.*, vol. 19, no. 1, pp. 7–32, 1985.

[11] B. Cockburn, J. Guzmán, and H. Wang, “Superconvergent discontinuous Galerkin methods for second-order elliptic problems,” *Math. Comp.*, vol. 78, no. 265, pp. 1–24, 2009.

[12] R. Sevilla, S. Fernández−Méndez, and A. Huerta, “Comparison of high–order curved finite elements,” *International Journal for Numerical Methods in Engineering*, vol. 87, no. 8, pp. 719–734, 2011.