Title: Discontinuous Finite Elements for coupled problems
Author: Maria Oliver Parera
Advisor: Sonia Fernández-Méndez and Marco Discacciati
Department: Matemàtica Aplicada III (MA3)
Academic year: 2012-2013
Discontinuous Finite Elements for coupled problems.

Maria Oliver i Parera

Advisor: Sònia Fernández-Méndez and Marco Discacciati

Matemàtica Aplicada III (MA3)
Contents

Chapter 1. Introduction 1
Chapter 2. Setting the problem 3
Chapter 3. Hybridizable Discontinuous Galerkin Method 5
  1. Description of the method 5
  2. Error analysis 9
Chapter 4. Darcy-Stokes problem 13
  1. Darcy HDG weak form 13
  2. Stokes CG weak form 14
  3. Coupled HDG-CG system of equations 14
  4. Error of the coupled problem 15
Chapter 5. The fish tank example 17
References 21
Chapter 6. Conclusions 23
Appendix A. Programa 25
Chapter 1
Introduction

The filtration of fluids through porous media is a challenging problem with many relevant applications in filtration problems, such as the filtration of blood through arterial vessel walls or the filtration of water through sand.

To model this problem we consider different systems of partial differential equations on each domain: in the free domain we will use Stokes equations, while the fluid in the porous domain is modelled using Darcy equations. In the free domain the fluid is discretized by the Continuous Galerkin method, but in the porous domain we will use de Hybridizable Discontinuous Galerkin method.

The project is focused in the Hybridizable Discontinuous Galerkin method, which is a new method that combines the advantages of the Discrete Galerkin methods with the computational efficiency of the Continuous Galerkin methods. The main advantages of this method are:

- Reduced number of degrees of freedom. With the hybridization process we can reduce the number of degrees of freedom at the boundaries of each element.
- Optimal convergence. It converges with order $k + 1$ in the $L^2$ norm, where $k$ is the degree of the polynomials used to approximate the solution. This convergence is for the solution and the derivative of the solution.
- Superconvergence and local postprocessing. The method allows us to use an element-by-element postprocessing to obtain a new and better approximation with order $k + 2$, for $k \geq 1$.

The structure of the project is as follows. At the first Chapter we set the problem: the equations that we will use and the boundary conditions. Next Chapter is dedicated to the Hybridizable Galerkin method and to the analysis of the error of this method, for this Chapter we reference to [1], [3], [4], [7], [5]. We continue with a Chapter dedicated to the coupling, where we remark the article [6]. And finally we apply our problem to the filtration of water through sand.
Chapter 2

Setting the problem

We consider a domain split into two parts: one occupied by the fluid, the other by the porous media. More precisely, let $\Omega \in \mathbb{R}^2$ be a bounded domain, partitioned into two non-intersecting subdomains $\Omega_f$ and $\Omega_p$ separated by an interface $\Gamma$. In Fig. 1, we represent a possible domain.

![Fig. 1. Representation of the domain $\Omega$.](image)

To describe the filtration of the fluid through $\Omega_p$ we use Darcy equations.

1. \[ k^{-1}\omega + \nabla \varphi = 0 \text{ in } \Omega_p \]
2. \[ \nabla \cdot \omega = g \text{ in } \Omega_p, \]
3. \[ \nabla \varphi \cdot \mathbf{n} = g_N \text{ on } \partial \Omega_p^N \]
4. \[ \varphi = g_D \text{ on } \partial \Omega_p^D \]

where,

- $k$ is the permeability of the fluid through the medium.
- $\varphi$ is the peziometric head, that represents the fluid pressure in $\Omega_p$.
- $\omega$ is the velocity.
- $g$ is the amount of liquid generated by unity of volume. From now on we will consider $g = 0$, except when we do the analysis of the error.
As the subdomain $\Omega_p$ is irregular (porous) the solution will have jumps in some regions, so if we want a more accurate solution we should avoid a continuous method, which will not detect the jumps, so we will use a Discrete Galerkin method in $\Omega_p$, in particular the Hybridized Discontinuous Galerkin Method, which is explained in Chapter 3.

The filtration of the fluid through $\Omega_f$ will be described by Stokes equations

\begin{align*}
-\nu \nabla^2 u + \nabla p &= f & \text{in } \Omega_f \\
-\nabla \cdot u &= 0 & \text{in } \Omega_f \\
u &= 0 & \text{on } \partial \Omega_f
\end{align*}

where,

- $\nu$ is the viscosity of the fluid in $\Omega_f$
- $p$ is the pressure of the fluid in $\Omega_f$
- $u$ is the velocity of the fluid in $\Omega_f$
- $f$ is a given volumetric force.

We will solve the Stokes equation using the Continuous Galerkin Finite Elements Method. Once we have each problem posed we have to couple suitably the Stokes and the Darcy problems, as we explain in Chapter 4.

In particular, we need to impose some conditions over the boundary that is shared by both subdomains, $\Omega_p$ and $\Omega_f$.

\begin{align*}
\omega \cdot n_p &= u \cdot n_p & \text{on } \Gamma \\
(-\nu \nabla u + pI) \cdot n_f &= \varphi n_f + (\alpha u \cdot t) t & \text{on } \Gamma
\end{align*}

where, $n_p$ is the normal vector pointing outwards to $\Omega_p$, $n_f = -n_p$ is the normal vector pointing outward to $\Omega_f$, and $t$ is the tangent vector on the interface $\Gamma = \bar{\Omega}_p \cap \bar{\Omega}_f$.

Equation (8) imposes continuity on the normal component of the velocity. Equation (9) can be decomposed into their normal and tangent components, by multiplying by the normal or the tangential vector, respectively:

\begin{align*}
n_f(-\nu \nabla u + pI) n_f &= \varphi \\
t(-\nu \nabla u + pI) n_f &= \alpha u
\end{align*}

Equation (10) relates the pressure of the fluid at $\Omega_p$ with the pressure of the fluid at $\Omega_f$, while condition (11) the tangential component of the velocity of the fluid with the tangential component of the velocity across the interface.
Chapter 3
Hybridizable Discontinuous Galerkin Method

1. Description of the method

In this chapter we introduce the Hybridizable Discontinuous Galerkin Method (HDG) for an elliptic problem. We will illustrate this method considering the Darcy equation.

\begin{align*}
  k^{-1} \omega + \nabla \varphi &= 0 \quad \text{in } \Omega_p, \\
  \nabla \cdot \omega &= 0 \quad \text{in } \Omega_p, \\
  \nabla \varphi \cdot n &= g_N \quad \text{on } \partial \Omega_p^N \cup \Gamma \\
  \varphi &= g_D \quad \text{on } \Omega_p^D
\end{align*}

1.1. Mesh and trace operators.

We consider a triangulation $\mathcal{T}_h$ of $\Omega_p$ made of shape-regular simplexes $K_i$, with boundaries $\partial K_i$. We define the boundary face of an element $K_i \in \mathcal{T}_h$ by

$$F^\partial = \partial \Omega_p \cap \partial K;$$

and, the interior face for two elements $K^+$ and $K^-$ in $\mathcal{T}_h$

$$F^0 = \partial K^- \cap \partial K^+.$$

We denote by $E^0_h$ and by $E^\partial_h$ the set of interior and boundary faces, respectively. We set $E_h = E^0_h \cup E^\partial_h$.

\begin{figure}[h]
  \centering
  \begin{tikzpicture}
    \node (K) at (0,0) {$K^+$};
    \node (K-1) at (1,0) {$K^-$};
    \node (K+1) at (1,1) {$K$};
    \node (K-) at (0,1) {$K$};
    \draw (K) -- (K-1) -- (K+1) -- (K-1) -- (K-1) -- (K-1);\end{tikzpicture}
  \caption{Two consecutive elements of a mesh of the HDG}
\end{figure}

Remark 3.1. We can observe in Fig.1. lines surrounding the elements, they represent a first approximation of the variable $\varphi$ on the faces which help to reduce
the number of degrees of freedom. We will represent the value of \( \varphi \) over the trace by \( \hat{\varphi} \) and, particularly, for each element we will denote by \( \hat{\varphi}^F_i \), \( i = 1, 2, 3 \), the values of \( \varphi \) at each face of the element.

As we are considering a discrete method we have discontinuities between elements, therefore we need to introduce the jump operator \([\cdot]\), which is defined along the interface \( F^0 \) using values from the elements of the left and right hand sides of the interface:
\[
[\omega \cdot n] = \omega^+ \cdot n^+ + \omega^- \cdot n^-
\]
where \( n^+ \) and \( n^- \) are the outward unit normal vectors on two neighbouring elements \( K^+ \) and \( K^- \), respectively.

1.2. Approximation spaces.
Let \( P^p(D) \) denote the space of polynomials of degree at most \( p \) on a domain \( D \) and let \( L^2(D) \) be the space of square integrable functions on \( D \). We set \( P^k(D) = \bigoplus P^p(K) \) and \( L^2(D) = \bigoplus L^2(F) \). The Hybridizable Discontinuous Galerkin (HDG) Method seeks an approximation to the exact solution \((\omega, \varphi, \hat{\varphi}), (\omega_h, \varphi_h, \hat{\varphi}_h)\) in the space \( V_h \times W_h \times M_h \) defined as
\[
V_h := \{ v \in L^2(\Omega_p) : v|_K \in P^p(K) \ \forall K \in T_h \}
\]
\[
W_h := \{ w \in L^2(\Omega_p) : w|_K \in P^p(K) \ \forall K \in T_h \}
\]
\[
M_h = \{ \mu \in L^2(\mathcal{E}) : \mu|_F \in P^p(F) \ \forall F \in T_h \}
\]

1.3. Definition of the HDG method.
The weak form of the Darcy problem in each element of the triangulation \( T_h \) is
\[
(\varsigma, \omega)_K + (\nabla \varsigma, \varphi)_K - (\varsigma \cdot n, \hat{\varphi})_{\partial K} = 0 \quad \forall \varsigma \in V_h \tag{19}
\]
\[
(\nabla \gamma, \omega)_K - (\gamma, \hat{\varphi} \cdot n)_{\partial K} = 0 \quad \forall \gamma \in W_h \tag{20}
\]
where
\[
\hat{\omega} = \omega + \tau (\varphi - \hat{\varphi}_h) \tag{21}
\]
and \( \tau > 0 \) is a stabilization parameter.

Remark 3.2. Notice that when \( \tau \simeq 0, \omega - \hat{\omega} \simeq 0; \) and when \( \tau \gg 0, \varphi - \hat{\varphi} \simeq 0. \) So, \( \tau \) is a parameter that links the jump of the flux with the solution.

Substituting (21) in (19)-(20) we obtain
\[
(\varsigma, \omega)_K - (\nabla \varsigma, \varphi)_K + (\varsigma \cdot n, \hat{\varphi})_{\partial K} = 0 \quad \forall \varsigma \in V_h \tag{22}
\]
\[
(\nabla \gamma, \omega)_K - (\gamma, \omega \cdot n)_{\partial K} + (\gamma \cdot \tau \hat{\varphi} n)_{\partial K} = 0 \quad \forall \gamma \in W_h \tag{23}
\]
Above and throughout, we use the notation
\[
(f, g)_\Omega = \int_\Omega f g d\Omega \quad \text{and} \quad (f, g)_{\partial \Omega} = \int_{\partial \Omega} f g d\Omega
\]
1.4. Discretization.
We consider in each element $K_i$ the values of $\omega, \varphi, \hat{\varphi}$ as linear combinations of the basis functions of the spaces $V_h, W_h$ and $M_h$. We denote by $\{N_i\}_{i=1}^{N_\omega}$ the basis functions of $W_h$, by $\{N_i\}_{i=1}^{N_\varphi}$ those of $V_h$, and by $\{M_i\}_{i=1}^{N_{\hat{\varphi}}}$ those of $M_h$.

\begin{align*}
\omega &= \sum_{j=1}^{N_\omega} \omega_j N_j \\
\varphi &= \sum_{j=1}^{N_\varphi} \varphi N_j \\
\hat{\varphi} &= \sum_{j=1}^{N_{\hat{\varphi}}} \hat{\varphi} M_j
\end{align*}

Using the finite expansions we can write (22) and (23) as

\begin{align*}
A^K_\omega \omega + A^K_\omega \varphi + A^K_\omega \hat{\varphi} = 0 \\
A^K_\varphi \omega + A^K_\varphi \varphi + A^K_\varphi \hat{\varphi} = 0
\end{align*}

where, $\Lambda^K = \begin{pmatrix} \frac{\hat{\varphi} F_1}{\varphi} \\ \frac{\hat{\varphi} F_2}{\varphi} \\ \frac{\hat{\varphi} F_3}{\varphi} \end{pmatrix}$ on each element $K$ and $(A^K_{\omega\omega})_{ij} = \int_K N_i N_j d\Omega$.

Rewriting (27) and (28) as

\begin{align*}
\begin{pmatrix} A^K_{\omega\omega} & A^K_{\omega\varphi} & A^K_{\omega\hat{\varphi}} \\ A^K_{\varphi\omega} & A^K_{\varphi\varphi} & A^K_{\varphi\hat{\varphi}} \\ A^K_{\hat{\varphi}\omega} & A^K_{\hat{\varphi}\varphi} & A^K_{\hat{\varphi}\hat{\varphi}} \end{pmatrix} \begin{pmatrix} \omega^K \\ \varphi^K \end{pmatrix} = - \begin{pmatrix} A^K_{\omega\omega} \\ A^K_{\varphi\varphi} \\ A^K_{\hat{\varphi}\hat{\varphi}} \end{pmatrix} \Lambda^K
\end{align*}

we obtain

\begin{align*}
\begin{pmatrix} \omega^K \\ \varphi^K \end{pmatrix} = - \begin{pmatrix} A^K_{\omega\omega} \\ A^K_{\varphi\varphi} \\ A^K_{\hat{\varphi}\hat{\varphi}} \end{pmatrix} \begin{pmatrix} A^K_{\omega\omega} & A^K_{\omega\varphi} \\ A^K_{\varphi\omega} & A^K_{\varphi\varphi} \\ A^K_{\hat{\varphi}\omega} & A^K_{\hat{\varphi}\varphi} \end{pmatrix}^{-1} \begin{pmatrix} A^K_{\omega\omega} \\ A^K_{\varphi\varphi} \\ A^K_{\hat{\varphi}\hat{\varphi}} \end{pmatrix} \Lambda^K =: \begin{pmatrix} Q^K \\ U^K \end{pmatrix} A^K
\end{align*}

Therefore, we have to solve

\begin{align*}
\omega^K &= Q^K \Lambda^K \\
\varphi^K &= U^K \Lambda^K
\end{align*}

But, it solves $\omega^K$ and $\varphi^K$ as functions of $\Lambda^K$. We need a way to find the value of the trace in each element.

1.5. Computation of $\hat{\varphi}$ in each element.
In the previous sections we have considered the problem only in the elements, but at the beginning we have introduced a new element around each $K_i$: the trace. So, the problem is not closed, we need to impose a conservativity condition over the traces:

\begin{align*}
\omega \cdot n = \nabla \varphi \cdot n
\end{align*}
together with the following boundary conditions

\begin{align}
\dot{\phi} &= g_D \quad \text{on } \partial \Omega_p^D \\
\nabla \omega \cdot n &= g_N \quad \text{on } \partial \Omega_p^N \\
\| \varphi \cdot n \| &= 0 \quad \text{on } \kappa_h
\end{align}

where (34) and (35) impose the same boundary conditions we were considering in the initial problem of this Chapter, while (36) imposes continuity of the velocity when it crosses from one element to another.

We need to introduce the weak form of (33):

\begin{equation}
\sum_K \langle \dot{\varphi} \cdot n, \mu \rangle_{\partial K} = \langle g_N, \mu \rangle_{\partial \Omega_p^N} \quad \forall \mu \in M_h
\end{equation}

Using (21) this weak form can be rewritten as

\begin{equation}
\sum_K \left( \langle \varphi \cdot n, \mu \rangle_{\partial K} + \langle \tau (\omega - \dot{\omega}), \mu \rangle_{\partial K} \right) = \langle g_N, \mu \rangle_{\partial \Omega_p^N} \quad \forall \mu \in M_h
\end{equation}

what is equivalent to

\begin{align}
\langle \varphi_L \cdot n_L, \mu \rangle_{\partial K} + \langle \varphi_R \cdot n_R, \mu \rangle_{\partial K} - \langle (\tau_L + \tau_R) \dot{\omega}, \mu \rangle_{\partial K} \\
+ \langle \tau_L \omega_L, \mu \rangle_{\partial K} + \langle \tau_R \omega_R, \mu \rangle_{\partial K} = 0 \quad \forall \mu \in M_h
\end{align}

and using the finite expansions introduced in (24) - (26), can be rewritten as

\begin{equation}
A_L^F \varphi^F + A_R^F \varphi^F + A_L^\varphi \varphi_L + A_R^\varphi \varphi_R = 0
\end{equation}

So, now we only have to substitute \( \varphi \) and \( \omega \) by (31) and (32), respectively to obtain an equation that depends only on \( \varphi^F \). Once we have the value of \( \varphi^F \) we only have to use (31) and (32) to recover \( \omega \) and \( \varphi \) on each element.

If we are in a boundary trace we have to procedure in the same way, but considering the right hand side of equation (41) equal to \( g_N \).

1.6. Postprocessing.

We show how to postprocess the approximate solution in every element to obtain a new approximation which converges with an additional order. We define on each element

\begin{align}
- \nabla \cdot (k \nabla \varphi^*_h) &= \nabla \cdot \omega_h, \quad \text{in } K \\
- k \nabla \varphi^*_h \cdot n &= \omega_h \cdot n \quad \text{on } \partial K,
\end{align}

but, as problem (43) has not unicity of solution we need an additional constrain:

\begin{equation}
\int_K \varphi^*_h d\Omega = \int_K \varphi_h d\Omega.
\end{equation}

that imposes equality between the mean value of the solution with the postprocessed solution. Equations (42) - (44) induce the following weak problem: find \( u^* \in V^*_h \) such that

\begin{align}
\langle \nabla \varphi^*_h, \nabla v \rangle_K &= \langle \omega_h, \nabla v \rangle_K \quad \forall v \in W^*_h \\
\langle u^*_h, 1 \rangle_K &= \langle u_h, 1 \rangle_K
\end{align}

with \( W^*_h := \{ w \in L^2(\Omega_h) : w|_K \in P^{p+1}(K) \ \forall K \in T_h \} \).
2. Error analysis

In this section we analyse the influence of parameter \( \tau \) and the behaviour of the error.

2.1. The influence of the parameter \( \tau \).

The coefficient \( \tau \) is a positive stabilization parameter that may be prescribed as a positive value in all faces (all faces approach) or a positive value in one face, arbitrary chosen, and 0 at the others (single face approach).

We are going to analyze the convergence of the exact \( L^2 \) error over the domain \( \Omega_f = [0,1] \times [0,1] \) as the element size decreases \((h = 1/2^n, n = 1, 2, 3, 4)\) for the solution \( \varphi \) and its postprocessed result \( \varphi^* \). We show four graphics for each case: for the all single approach (Fig. 2) and the single face approach Fig. 3) with values \( \tau = \{1, 10, 100, 1000\} \).

In both cases we can observe that when we increase the value of \( \tau \) we obtain a better convergence for the normal solution. But, the behaviour of the postprocessed solution varies: in the single face approach we can not observe a remarkable impact in the convergence of \( \varphi^* \) when we increase the value of \( \tau \); nevertheless, the all face approach gives us a worse result because we lose the convergence of the postprocessed solution when \( \tau \) is bigger.

![Graphs showing convergence of the HDG method using all faces approach.](image)

**Fig. 2.** Convergence of the HDG method using all faces approach.
2.2. Error estimate.

As the solution of $w$ can be derived from $\varphi$, in this section we will only analyse the behaviour of $\varphi$ and $\varphi^*$. We consider as analytical solution

\[ \varphi(x, y) = e^{\alpha (ax + by) + \beta \cos(cx + dy)} \]

with $\alpha = 0.1, \beta = 0.3, a = 5.1, b = -6.2, c = 4.3, d = 3.4$. The computational domain is $\Omega_p = [0, 1] \times [0, 1]$.

We are considering meshes that are obtained by splitting a regular $n \times n$ Cartesian grid into a total of $n^n + 2$ triangles, giving uniform element sizes of $h = 1/n$. On these meshes we consider polynomials of degree $k$ to do the approximation.

In table 1 we present the error and the order of convergence in $L^2$ norm using $\tau = 1$ in a single face approach.

We can observe as $\varphi$ converges with order $k + 1$, while the postprocessed solution, $\varphi^*$, converges with order $k + 2$, for $k > 1$, which is one order higher than the original solution.
2. ERROR ANALYSIS

<table>
<thead>
<tr>
<th>Degree</th>
<th>Mesh</th>
<th>( |\varphi - \varphi_h|_{\mathcal{T}_h} )</th>
<th>Error Order</th>
<th>( |\varphi^* - \varphi_h^*|_{\mathcal{T}_h} )</th>
<th>Error Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0.7153</td>
<td>0.0726</td>
<td>0.0726</td>
<td>0.0726</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.1710</td>
<td>2.1</td>
<td>0.0158</td>
<td>2.2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.0429</td>
<td>2</td>
<td>0.0041</td>
<td>1.9</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.0111</td>
<td>2</td>
<td>0.0010</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0.1876</td>
<td>–</td>
<td>0.0148</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.0284</td>
<td>2.7</td>
<td>0.0148</td>
<td>3.8</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.004</td>
<td>2.8</td>
<td>7.7113e^{-005}</td>
<td>3.8</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>5.0837e^{-004}</td>
<td>3</td>
<td>5.6406e^{-006}</td>
<td>3.8</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>0.0585</td>
<td>–</td>
<td>0.003</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.0054</td>
<td>3.4</td>
<td>1.4265e^{-004}</td>
<td>4.4</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3.5540e^{-006}</td>
<td>3.9</td>
<td>4.9109e^{-006}</td>
<td>4.9</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2.3592e^{-005}</td>
<td>3.9</td>
<td>1.69999e^{-007}</td>
<td>4.9</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>0.0162</td>
<td>–</td>
<td>7.4580e^{-004}</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>8.1766e^{-004}</td>
<td>4.3</td>
<td>1.6938e^{-005}</td>
<td>5.5</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3.2804e^{-005}</td>
<td>4.6</td>
<td>3.3959e^{-007}</td>
<td>5.6</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>1.0550e^{-006}</td>
<td>5</td>
<td>5.5743e^{-009}</td>
<td>5.9</td>
</tr>
</tbody>
</table>

Table 1. History of the convergence of \( \varphi \) using the HDG method for \( \tau = 1 \) using the analytical solution (47).
Chapter 4
Darcy-Stokes problem

In this chapter we are going to couple both Darcy and Stokes equations. The problem, with the condition in the interface, becomes:

\[ k^{-1} \omega + \nabla \varphi = 0 \quad \text{in } \Omega_p \]  
\[ \nabla \cdot \omega = 0 \quad \text{in } \Omega_p, \]  
\[ \omega \cdot n_p = u \cdot n_p \quad \text{on } \Gamma \]  
\[ -\nu \nabla^2 u + \nabla p = f \quad \text{in } \Omega_f \]  
\[ -\nabla \cdot u = 0 \quad \text{in } \Omega_f, \]  
\[ (-\nu \nabla u + p I) \cdot n_f = \varphi_n + (\alpha u \cdot t) t \quad \text{on } \Gamma \]  

1. Darcy HDG weak form

The local weak form for the Darcy problem (48)-(50) at each element \( K \) of the triangulation \( T_h \) is

\[ (\varsigma, \omega)_K - (\nabla \varsigma, \varphi)_K + (\varsigma \cdot n, \hat{\varphi})_{\partial K} = 0 \quad \forall \varsigma \in V_h \]  
\[ (\nabla \gamma, \omega)_K - (\gamma, \omega \cdot n)_{\partial K} + (\gamma, \tau \varphi n)_{\partial K} - (\gamma, \tau \varphi \cdot n)_{\partial K} = 0 \quad \forall \gamma \in W_h \]  
\[ \langle \mu, \omega \cdot n \rangle_{\partial T_h} - \langle \mu, \varphi \rangle_{\partial T_h} + \langle \varsigma \mu, \varphi \rangle_{\partial T_h} - \langle \mu, u \cdot n \rangle_{\Gamma} = 0 \quad \forall \mu \in M_h \]  

where, (56) comes from taking the weak form of (50):

\[ \langle \mu, \hat{\omega} \cdot n \rangle_{\partial T_h} = \langle \mu, u \cdot n \rangle_{\Gamma} \quad \forall \mu \in M_h \]  

and using

\[ \hat{\omega} = \omega + \tau (\varphi - \hat{\varphi}) \]

The discretization of this weak form leads to the following system of equations

\[ A^K_{\omega \omega} + A^K_{\varphi \varphi} + A^K_{\omega \varphi} + A^K_{\varphi \omega} = 0 \]  
\[ A^K_{\varphi \omega} + A^K_{\varphi \varphi} + A^K_{\varphi \varphi} + A^K_{\varphi \omega} = 0 \]  
\[ A^K_{\varphi \omega} + A^K_{\varphi \varphi} + A^K_{\varphi \varphi} + A^K_{\varphi \varphi} = M^T_n u \]
where,

\[ M^n_\Gamma = (M^n_{u_1}, M^n_{u_2}) , \quad \text{with} \quad (M^n_{u_k})_{ij} = \int_{\Gamma} n_k N_i N_j d\Gamma \]  

with \( n_p = (n_1, n_2)^T \). As equations (59) and (60) are local equations in each element \( K \), we can express \( \varphi^K \) and \( \omega^K \) in terms of the trace \( \Lambda^K \), leading to

\[ \omega^K = Q^K \Lambda^K \]  
\[ \varphi^K = U^K \Lambda^K \]  

with

\[ (U^K Q^K)_{ij} = \int_{\Gamma} N_i N_j d\Gamma \]  

Replacing (64) in (61) we obtain the following system of equations for the trace variable

\[ D \Lambda + M^n_\Gamma u = 0 \]  

with \( D = -(A_{\hat{\varphi}\varphi} U + A_{\hat{\varphi} \omega} Q + A_{\hat{\varphi} \hat{\varphi}}) \).

2. Stokes CG weak form

The Continuous Galerkin weak form for the Stokes problem (51) - (53) can be stated as

\[ (\nu \nabla u \nabla v)_{\Omega_f} - (p, \nabla \cdot v)_{\Omega_f} - \langle \hat{\varphi}, v \cdot n_p \rangle_{\Gamma} + \langle \alpha u \cdot t, v \cdot t \rangle_{\Gamma} = (f, v)_{\Omega_f} \quad \forall v \in [H^1_0(\Omega)]^d \]  
\[ - (\nabla \cdot u, q)_{\Omega_f} = 0 \quad \forall q \in L^2_0(\Omega_f) \]  

This weak form leads to the following system of equations:

\[ (K + \alpha M^n_\Gamma) u + G p - (M^n_n)^T \hat{\varphi} = f \]  
\[ G^T u = 0 \]  

with

\[ M^n_\Gamma = \begin{pmatrix} M^n_{\Gamma_{11}} & M^n_{\Gamma_{12}} \\ M^n_{\Gamma_{21}} & M^n_{\Gamma_{22}} \end{pmatrix} , \quad (M^n_{u_1})_{ij} = \int_{\Gamma} t_i t_j N_i N_j d\Gamma \]  

and \( t_1 = -n_2, t_1 = n_1 \).

3. Coupled HDG-CG system of equations

The coupled problem is obtained joining (66), (69) and (70)

\[ \begin{pmatrix} D & M^n_u & 0 \\ -(M^n_n)^T & S & G \\ 0 & G^T & 0 \end{pmatrix} \begin{pmatrix} \varphi \\ u \\ p \end{pmatrix} = \begin{pmatrix} 0 \\ f \\ 0 \end{pmatrix} \]  

with \( S = K + \alpha M^n_\Gamma \).
4. Error of the coupled problem

We consider as analytical solution

\[ \varphi = k^{-1} \left( x(1-x)(y-1) + \frac{(y-1)^3}{2} \right) \]

\[ \omega = k^{-1} \left( (2x-1)(y-1) - 2\nu, -x(1-x) - (y-1)^2 \right) \]

\[ u = ((y-1)^2 + (y-1) + 1, x(x-1)) \]

\[ p = 2\nu(x+y-1) \]

with \( \nu = 1 \) and \( k^{-1} = 1 \). The mesh that we consider is uniform with element size \( h = \frac{1}{2} \). And, for the hdg, we consider a single face approach with \( \tau = 1 \). On these meshes we consider polynomials of order \( k = 1 \) to compute the approximation.

In table 1 we show the error of the solution of the Darcy problem and the order of convergence in the norm \( L^2 \). In table 2 we show the same for the Stokes solution.

We can observe that the coupling does not effect the convergence in any method. As we are using linear polynomials for the approximation we obtain second order convergence for all the solutions, including the postprocessed.
Table 1. History of the convergence of the Darcy solution using the HDG method using the analytical solution (73)-(74).

| Mesh | $||\varphi - \varphi_h||_{T_h}$ | $||\omega - \omega_h||_{T_h}$ | $||\varphi^* - \varphi^*_h||_{T_h}$ |
|------|---------------------------------|---------------------------------|-----------------------------------|
| $n$  | Error                           | Order                           | Error                            | Order                            |
| 1    | 0.039798                        | –                               | 0.11942                          | –                                |
| 2    | 0.0099884                       | 2                               | 0.34487                          | 1.8                              |
| 3    | 0.0025501                       | 2                               | 0.0095844                        | 1.8                              |
| 4    | 0.00065083                      | 2                               | 0.0026083                        | 1.9                              |
| 5    | 0.00016483                      | 2                               | 0.00070037                       | 1.9                              |

Table 2. History of the convergence of the Stokes solution using the HDG method using the analytical solution (75)-(76).

| Mesh | $||u - u_h||_{T_h}$ | $||p - p_h||_{T_h}$ |
|------|--------------------|---------------------|
| $n$  | Error              | Order               | Error                            | Order                            |
| 1    | 0.003212           | –                   | 0.12927                          | –                                |
| 2    | 0.0083822          | 1.8                 | 0.030556                         | 2.1                              |
| 3    | 0.0020846          | 1.8                 | 0.008948                         | 1.8                              |
| 4    | 0.00051827         | 1.9                 | 0.0027191                        | 1.7                              |
| 5    | 0.00012913         | 1.9                 | 0.00084647                       | 1.7                              |
Chapter 5
The fish tank example

In this chapter we are considering a fish tank that contains sand at the bottom and water at the rest of the tank. The permeability of the sand will be considered as \( k = 10^{-3} \), while the viscosity of water is \( \nu = 10^{-6} \).

We are considering that the cover is moving to the right and provokes movement to the water that is at the top. At the the porous zone (sand) we are considering waterproof boundaries, which is represented by Neumann boundary conditions; and at the the liquid zone we are considering that the water has no velocity at the boundary, i.e we are using Dirichlet boundary conditions.
The equations that model the stated problem are:

\[
\begin{align*}
  k^{-1}\omega + \nabla \varphi &= 0 & \text{in } \Omega_p \\
  \nabla \cdot \omega &= 0 & \text{in } \Omega_p \\
  -\nu \nabla^2 \mathbf{u} + \nabla p &= f & \text{in } \Omega_f \\
  -\nabla \cdot \mathbf{u} &= 0 & \text{in } \Omega_f \\
  \omega \cdot \mathbf{n}_p &= \mathbf{u} \cdot \mathbf{n}_p & \text{on } 0 \leq x \leq 1 \text{ and } y = 1 \\
  (-\nu \nabla \mathbf{u} + p \mathbf{I}) \cdot \mathbf{n}_f &= \varphi \mathbf{n}_f + (\alpha \mathbf{u} \cdot \mathbf{t}) \mathbf{t} & \text{on } 0 \leq x \leq 1 \text{ and } y = 1 \\
  \mathbf{u} &= \mathbf{0} & \text{on } 0 \leq x \leq 1 \text{ and } 1 < y < 2 \\
  \mathbf{u} &= (1,0) & \text{on } 0 \leq x \leq 1 \text{ and } y = 2 \\
  \mathbf{w} \cdot \mathbf{n} &= 0 & \text{on } 0 \leq x \leq 1 \text{ and } 0 \leq y > 1.
\end{align*}
\]

(77)

where we are considering \( f = (0, 9.8)^t \) and \( \alpha = 1 \), because the only force that is affecting the water in the fluid medium is gravity.

We are considering polynomials of degree 2 for the pressure and 3 for the velocity.

Next we show the graphics obtained after applying the coupled problem to this particular case.

We can observe that the maximum movement of the x-component is at the top of the domain and it dissipates rapidly to become more homogeneous in the whole fluid domain, while the y-component has two important focus at the top: one at the left side and the other at the right. The blue color represents negative values, so the fluid is going down, while red color represents positive values, so in the left hand side the water is going up. In the porous domain we can’t appreciate any movement. So we have that the water is moving to the right hand side at the top and going down when it arrives at the right boundary to go up again at the left boundary. So, it has a circular movement at the fluid domain.
In Figure(2) we simulate the flux of the liquid in the domain. Here it is easier to appreciate the circular movement of the fluid. We can also see a little movement at the top of the porous domain due to the water that is moving horizontally very close to this part of the domain.
References


[6] Discacciati M., Coupling free and porous-media flows: models and numerical approximation. [To be published]


Chapter 6
Conclusions

We have solved the Darcy-Stokes a domain divided in a porous media and a free fluid media. In order to detect the discontinuities of the porous media we have implemented a discontinuous galerkin method at this part of the domain, in particular the hybridizable DG.

We have studied and implemented the HDG, doing a special emphasis to the behaviour of the method when varying the value of the parameter $\tau$ and its position on the faces. We have also studied the error of this method and have seen that, thanks to doing a postprocessing, the method converges with an extra order: when we approximate the solution using polynomials of order $k$, after doing the postprocessing, we obtain a convergence of order $k + 2$.

As we have used the continuous method at the free fluid media we needed to couple both methods. We has to impose a condition at the common interface to explain the behaviour of the fluid when it crosses from one domain to the other.

The example showed us the good efficiency of the method as it gave us the expected behaviour of the solution.
Appendix A
Programa

Once we have created the mesh we need to know which are the interior and the exterior faces of each element. For the interior faces we also need to know which face of the contiguous element is connected to it. From it we can compute the connectivity matrix.

Apart from it we need to compute the reference element because it simplifies the computations at each element. Remark that we need two reference elements: the HDG and the CG.

Finally we only have to impose the interface condition.

Once we have everything organized we only have to calculate the matrixes showed in Chapter 4, impose the boundary conditions and solve a linear system.

Next we show the program that have been used

```matlab
1 %—PREPROCESS
2 % HDG preprocess
3 infoFaces = GetFaces(TD(:,1:3));
4 nOfElements = size(TD,1); nOfElementNodes = size(TD,2);
5 nOfInteriorFaces = size(infoFaces.intFaces,1);
    nOfExteriorFaces = size(infoFaces.extFaces,1);
6 % HDG reference element
7 referenceElementHDG = createReferenceElement(elementTypeD,
        nOfElementNodesD);
8 nOfFaceNodes = size(referenceElementHDG.NodesCoord1d,1);
9 % Stokes reference element
10 referenceElementStokes = createReferenceElementStokes(
        elementTypeS,nOfElementNodesS);
11 nOfVelocityNodes = size(XS,1); if elementTypeS==2,
    nOfVelocityNodes = nOfVelocityNodes + size(TS,1); end
12 % INTERFACE definition in both domains
13 [infoFaces,nOfInterfaceFaces] = setBoundaryFacesDarcyStokes(
        infoFaces,XD_TD,referenceElementHDG);
```
A. PROGRAMA

F = getFaceConnectivity(infoFaces); nOfFaces = max(max(F));
interfaceFaces = infoFaces.extFaces(1:nOfInterfaceFaces,:);
Xinterface = computeNodalCoordinatesFaces(interfaceFaces,XD,
TD,referenceElementHDG);
hold on, plot(Xinterface(:,1),Xinterface(:,2),'k*');
Tinterface = CGconnectivity_HDGFaces(XS,interfaceFaces,XD,
TD,referenceElementHDG);
hold on, plotCG1DMesh(XS,Tinterface), hold off

%%—COMPUTATION—
% HDG computation
disp('HDG loop in elements...')
[KD fD QQ UU Qf Uf] = hdg_matrix_laplace(XD,TD,F,
referenceElementHDG,infoFaces,tau,coefk,@font);

% CG computation
disp('CG loop in elements...')
[KS,G,fS] = computeStokesMatrices(XS,TS,XP,TP,
referenceElementStokes,nu,@StokesSourceTerm);
disp('Interface computation...')
[Mn,Mt] = hdg_interfaceMatrices_DarcyStokes(infoFaces,
 nOfInterfaceFaces,...
XD,TD,Tinterface,nOfVelocityNodes,
referenceElementHDG);
disp('System assembly...')
nD = nOfFaceNodes*nOfFaces; nP = size(XP,1); aux = spalloc(nD
 ,nP,1); aux2 = spalloc(nP,nP,1);
totalnDOF = nD+nP+2*nOfVelocityNodes;
K = [-KD Mn aux; -Mn' (KS+alpha*Mt) G'; aux' G aux2];
f = [fD;fS;zeros(nP,1)];

% HDG—Darcy Dirichlet boundary conditions
% Assumed face ordering: interior, interface, other...
disp('Dirichlet BC...')
%DirichletFaces = infoFaces.extFaces(nOfInterfaceFaces+1:
end,:);
%XDDirichlet = computeNodalCoordinatesFaces(DirichletFaces,XD,
 TD,referenceElementHDG);
%valDBC_D = analyticalPotential(XDirichlet);
%indDBC_D = (nOfInteriorFaces+nOfInterfaceFaces)*
nOfFaceNodes+[1:size(valDBC_D,1)];
%CG—Stokes Dirichlet boundary conditions
[indDBC_S,valDBC_S]=setDirichletBCStokes(XS);
% System reduction (Dirichlet faces are set to prescribed
value)
indDBC = [nD+indDBC_S;1]; valDBC = [valDBC_S;0];
indDOF = setdiff([1:totalnDOF], indDBC);
f = f(indDOF) - K(indDOF, indDBC)*valDBC;
K = K(indDOF, indDOF);

%System solution
disp('System solution...')
systemSolution = K\f;

%Prescribed values
sol = zeros(totalnDOF, 1);
sol(indDOF) = systemSolution;
sol(indDBC) = valDBC;
disp('Done!')
disp('Postprocess...')

%—POSTPROCESS—
phiHat = sol(1:nD);
u = [sol(nD+[1:2:2*nOfVelocityNodes]) sol(nD+[2:2:2*nOfVelocityNodes])];
p = sol(nD+2*nOfVelocityNodes+1:end);

% HDG element-by-element postprocess
[phi,w]=computeElementsSolution(phiHat, UU, QQ, Uf, Qf, TD, F);
wx = w(1:2:end); wy= w(2:2:end);

% PLOTS
Xplot = XS; uplot=u;
if elementTypeS==2, [Xplot, uplot]=miniElementPlotVariables(Xplot, TS, uplot); end
tri = delaunay(Xplot(:,1), Xplot(:,2));
figure(3), clf
plotDiscontinuousSolution(XD, TD, wx, referenceElementHDG, 8);
hold on, trisurf(tri, Xplot(:,1), Xplot(:,2), uplot(1:size(Xplot,1),1)); shading interp, hold off
axis([0 1 0 2])
colorbar caxis(x)
title('Velocity x-component')

figure(4), clf
plotDiscontinuousSolution(XD, TD, wy, referenceElementHDG, 8);
hold on, trisurf(tri, Xplot(:,1), Xplot(:,2), uplot(1:size(Xplot,1),2)); shading interp, hold off
axis([0 1 0 2])
colorbar caxis(y)
title('Velocity y-component')

figure(5), clf
[Xplot1, plot1] = plotDiscontinuosSolution(XD, TD, wx, referenceElementHDG, 8);
[Xplot1, plot2] = plotDiscontinuosSolution(XD, TD, wy, referenceElementHDG, 8);
quiver(Xplot(:,1), Xplot(:,2), uplot(:,1), uplot(:,2));
hold on
quiver(Xplot1(:,1), Xplot1(:,2), plot1, plot2);
axis([0 1 0 2]), axis equal
title('Flux')

Finally, we do a brief description of the programs we are calling.

Here we introduce a summary of the routines we need for the main program. We get the input and the output.

**GetFaces**

- **input**: connectivity matrix.
- **output**: tells us how two elements are connected between themselves. Therefore, as output we get two matrices:
  1. Matrix for the interior faces. This is 5-columns matrix, and for each row we have that the first and third column corresponds to two elements in contact and the second and fourth rows tell us which face of each element is connected to which face of the other element. The last column is to know the orientation of the face.
  2. Matrix for the exterior faces. This matrix is smaller than the other one and only have two columns. The first column tells us which elements are exterior and the second column informs us, for each of these elements, which is the exterior face.

**createReferenceElement**

- **input**
  1. Type of element: can be a quadrilateral (number 0), a triangle (number 1) or the element mini (number 2)
  2. Number of Nodes of the reference element for the HDG method
- **output**: Information of the reference element for the HDG method.

**createReferenceElementStokes**

- **input**
  1. Type of element: can be a quadrilateral (number 0), a triangle (number 1) or the element mini (number 2)
  2. Number of Nodes of the reference element for the CG method
- **output**: Information of the reference element for the CG method.
setBoundaryFacesDarcyStokes

- input: the previous variable infoFaces (obtained with getFaces), the connectivity matrix and the “position” matrix, and the reference Element.
- output: by one hand the same information than the GetFaces, and by the other, the elements that are at the boundary.

getFaceConnectivity

- input: information about the Faces (the previous variable infoFaces)
- output: connectivity matrix for the faces: for every element gives us the number of its faces, taking into account that the row i-th corresponds to the element i-th.

computeNodalCoordinatesFaces

- input: Faces of the element (the 4 faces obtained previously, connectivity matrix, matrix of coordinates and the reference Element)
- output: the coordinates of the elements of the fluid domain that are in contact with the porous domain (only the y-coordinate)

CGconnectivityHDGFaces

- input: we introduce the position matrix of the CG, the connectivity and the position matrix of the HDG, the faces that are in the boundary of the HDG domain and finally the reference element of the HDG
- output: we obtain the elements that are in the boundary of the CG and the faces that are in contact with the other domain.

plotCG1DMesh
Plots the mesh of the CG method.

hdgmatrixlaplace

- input: the connectivity and position matrix of Darcy, the connectivity matrix for the Faces, the Reference element for the HDF, the information about the faces, things about our equations and we obtain the matrices explained at Chaper 3.
- output: we obtain the matrices of the HDG method.

computeStokesMatrice

- input: the connectivity matrices of the Stokes and the matrices of the coordinates; the reference element of the Stokes method, the viscosity of the fluid in $\Omega_f$ and the volumetric force
- output: the matrices for the CG

hdginterfaceMatricesDarcyStokes
• input: information about the faces, number of faces in contact between both domains, connectivity and position matrix, connectivity of the elements of the interfaces, velocity nodes and the reference element for the HDG
• output: Mn, Mt

**computeNodalCoordinatesFaces**

In this routine we impose the boundary conditions.

• input: we introduce the Faces, the connectivity and matrix and the matrix of coordinaes and the reference element for the HDG.
• output: coordinates of the nodes with boundary conditions.

**setDirichletBCStokes**

• input: The position matrix of the Stokes problem.
• output: elements and position of the dirichlet boundary conditions in the Stokes.

**computeElementsSolution**

• input: the matrices for computer the solution.
• output: the solution.

**plotDiscontiuosSolution**

• input: the connectivity matrix, the solution we want to plot and the reference element.
• output: the plot of the solution for the HDG