A short and efficient way to solve PDE's with P1 finite elements in vector languages

François Cuvelier and Gilles Scarella

Laboratoire d'Analyse Géométrie et Applications
Institut Galilée
Université Paris XIII.

Tuesday 24th March, 2015

Outline

Outline

Scalar BVP - definition

Scalar BVP

Find $u \in \mathrm{H}^2(\Omega)$ such that

$$\begin{aligned} \mathcal{L}(u) &= f && \text{in } \Omega \subset \mathbb{R}^d,\ d \geqslant 1 \\ u &= g^D && \text{on } \Gamma^D \subset \partial\Omega, \\ \frac{\partial u}{\partial n_{\mathcal{L}}} + a^R u &= g^R && \text{on } \Gamma^R \subset \partial\Omega. \end{aligned}$$

$$\mathcal{L} = \mathcal{L}_{\mathbb{A},\boldsymbol{b},\boldsymbol{c},a_0}(u) \overset{\text{def}}{=} -\operatorname{div}(\mathbb{A}\nabla u) + \operatorname{div}(\boldsymbol{b}u) + \langle \nabla u, \boldsymbol{c}\rangle + a_0 u$$

where $\mathbb{A} \in (L^\infty(\Omega))^{d\times d}$, $\boldsymbol{b} \in (L^\infty(\Omega))^d$, $\boldsymbol{c} \in (L^\infty(\Omega))^d$ and $a_0 \in L^\infty(\Omega)$.

$$\frac{\partial u}{\partial n_{\mathcal{L}}} = \langle \mathbb{A}\nabla u, \boldsymbol{n}\rangle - \langle \boldsymbol{b}u, \boldsymbol{n}\rangle$$

We set

$$\begin{aligned}
\mathcal{D}_{\mathcal{L}}(u,v) &= \langle \mathbb{A}\,\nabla u, \nabla v\rangle - (u\langle \boldsymbol{b}, \nabla v\rangle - v\langle \nabla u, \boldsymbol{c}\rangle) + a_0 uv.\\
\mathcal{A}_{\mathcal{L}}(u,v) &= \int_\Omega \mathcal{D}_{\mathcal{L}}(u,v)d\mathrm{q} + \int_{\Gamma^{\boldsymbol{R}}} a^R uv d\sigma\\
\mathcal{F}(v) &= \int_\Omega fv d\mathrm{q} + \int_{\Gamma^{\boldsymbol{R}}} g^R v d\sigma
\end{aligned}$$

Variational formulation

Find $u \in \mathrm{H}^1_{g^{\boldsymbol{D}},\Gamma^{\boldsymbol{D}}}(\Omega)$ such that

$$\mathcal{A}_{\mathcal{L}}(u,v) = \mathcal{F}(v),\ \forall v \in \mathrm{H}^1_{0,\Gamma^{\boldsymbol{D}}}(\Omega)$$

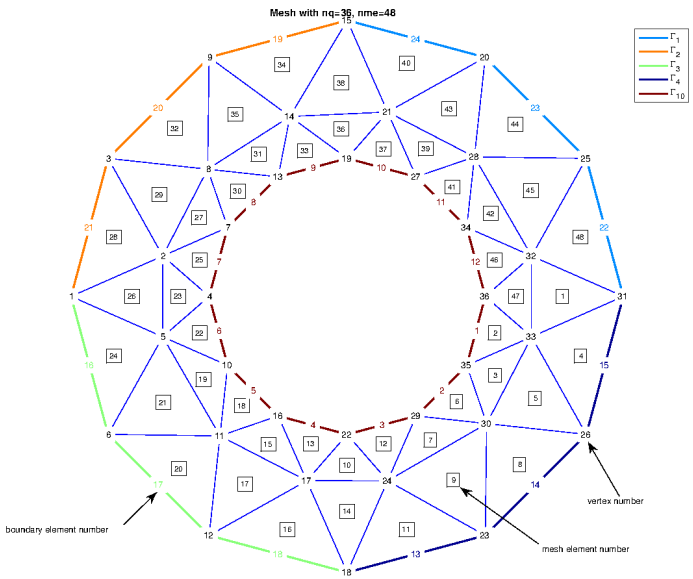
Variational formulation with an extension function R^D

Find $w \in \mathrm{H}^1_{0,\Gamma^{\boldsymbol{D}}}(\Omega)$ such that

$$\mathcal{A}_{\mathcal{L}}(w,v) = \mathcal{F}(v) - \mathcal{A}_{\mathcal{L}}(R^D,v),\ \forall v \in \mathrm{H}^1_{0,\Gamma^{\boldsymbol{D}}}(\Omega)$$

d-simplicial mesh in $\mathbb{R}^n$ $(d \leqslant n)$

d	:	dimension of the simplices
$\mathrm{n_q}$	:	total number of vertices
$\mathrm{n_{me}}$	:	number of mesh elements
q	:	n-by-$\mathrm{n_q}$ array of vertices coordinates
me	:	$(d+1)$-by-$\mathrm{n_{me}}$ connectivity array for **mesh elements**
vols	:	1-by-$\mathrm{n_{me}}$ array of volumes of the mesh elements
$\vdots$		



$\Omega \subset \mathbb{R}^n$, $n = 2$

- $\mathcal{T}_h$ is the 2-simplicial mesh

$$\Omega_h = \bigcup_{K \in \mathcal{T}_h} K = \bigcup_{k=1}^{\mathcal{T}_h.\mathrm{n_{me}}} T_k.$$

- $\mathcal{B}_h$ is the array of 1-simplicial meshes

$$\Gamma_h = \bigcup_{l=1}^{\mathrm{n_{lab}}} \bigcup_{K \in \mathcal{B}_h(l)} K$$

Matrix representation

Find $\boldsymbol{w}_j$, $\forall j \in \mathcal{I}_D^c$ such that

$$\sum_{j \in \mathcal{I}_D^c} \mathcal{A}_{\mathcal{L}}^h(\varphi_j, \varphi_i)\boldsymbol{w}_j = \mathcal{F}^h(\varphi_i) - \sum_{j \in \mathcal{I}_D} \mathcal{A}_{\mathcal{L}}^h(\varphi_j, \varphi_i)\boldsymbol{R}_j, \quad \forall i \in \mathcal{I}_D^c$$

$\forall (i,j) \in [\![1, \mathrm{n_q}]\!]^2$, we set

$$\mathbb{A}_{i,j}^{\mathcal{L}} = \mathcal{A}_{\mathcal{L}}^h(\varphi_j, \varphi_i) = \int_{\Omega_h} \mathcal{D}_{\mathcal{L}}(\varphi_j, \varphi_i) d\mathrm{q} + \int_{\Gamma_h^R} a^R \varphi_j \varphi_i d\mathrm{q} = \mathbb{D}_{i,j}^{\mathcal{L}} + \mathbb{A}_{i,j}^R,$$

$$\boldsymbol{b}_i^{\mathcal{L}} = \mathcal{F}^h(\varphi_i) = \int_{\Omega_h} f \varphi_i d\mathrm{q} + \int_{\Gamma_h^R} g^R \varphi_i d\sigma = \boldsymbol{b}_i^f + \boldsymbol{b}_i^R.$$

Matrix representation

$\boldsymbol{w}(\mathcal{I}_D^c)$ is the solution of

$$\mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D^c)\boldsymbol{x} = \boldsymbol{b}^{\mathcal{L}}(\mathcal{I}_D^c) - \mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D)\boldsymbol{R}(\mathcal{I}_D)$$

Algorithm 1 Steps for solving the *Scalar* Boundary Value Problem

1: Assembly of the matrix $\mathbb{D}^{\mathcal{L}}$
2: Computation of $\boldsymbol{b}^f$
3: Computation of *Robin* contributions $\mathbb{A}^R$ and $\boldsymbol{b}^R$
4: Computation of *Dirichlet* contributions $\mathcal{I}_D$, $\mathcal{I}_D^c$ and $\boldsymbol{R}$.
5: $\boldsymbol{b}^{\mathcal{L}} \leftarrow \boldsymbol{b}^f + \boldsymbol{b}^R$
6: $\mathbb{A}^{\mathcal{L}} \leftarrow \mathbb{D}^{\mathcal{L}} + \mathbb{A}^R$
7: $\boldsymbol{u}(\mathcal{I}_D^c) \leftarrow \text{Solve}(\mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D^c), \boldsymbol{b}(\mathcal{I}_D^c) - \mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D)\boldsymbol{R}(\mathcal{I}_D))$
8: $\boldsymbol{u}(\mathcal{I}_D) \leftarrow \boldsymbol{R}(\mathcal{I}_D)$

? But ...

How to assemble the matrices $\mathbb{D}^{\mathcal{L}}$, $\mathbb{A}^R$ and the vectors $\boldsymbol{b}^f$, $\boldsymbol{b}^R$?

Assembly of the matrix $\mathbb{D}^{\mathcal{L}}$

$\mathbb{D}^{\mathcal{L}}$ is the $\mathrm{n_q}$-by-$\mathrm{n_q}$ matrix defined $\forall (i,j) \in [\![1, \mathrm{n_q}]\!]^2$ by

$$\mathbb{D}^{\mathcal{L}}_{i,j} = \int_{\Omega_h} \mathcal{D}_{\mathcal{L}}(\varphi_j, \varphi_i) d\mathrm{q} = \sum_{k=1}^{\mathrm{n_{me}}} \int_{T_k} \mathcal{D}_{\mathcal{L}}(\varphi_j, \varphi_i) d\mathrm{q}$$

Remark

The $(d+1)$ P_1-Lagrange basis functions on a d-simplex K are the barycentric coordinates $(\lambda_\alpha)_{\alpha \in [\![1, d+1]\!]}$.

We denote by $\mathbb{D}^{e,\mathcal{L}}(K)$ the $(d+1) \times (d+1)$ **element matrix** defined by

$$\mathbb{D}^{e,\mathcal{L}}_{\alpha,\beta}(K) = \int_K \mathcal{D}_{\mathcal{L}}(\lambda_\beta, \lambda_\alpha)(\mathrm{q}) d\mathrm{q}, \quad \forall (\alpha, \beta) \in [\![1, d+1]\!]^2. \tag{1.1}$$

On $K = T_k$ we have $\lambda_\alpha = \varphi_{\mathrm{me}(\alpha,k)}$, $\forall \alpha \in [\![1, d+1]\!]$.

Algorithm 2 Classical assembly with element matrices

1: $\mathbb{D}^{\mathcal{L}} \leftarrow 0$ ▷ Sparse matrix $\mathrm{n_q}$-by-$\mathrm{n_q}$
2: **for** $k \leftarrow 1$ **to** $\mathrm{n_{me}}$ **do**
3: $\quad \mathbb{E} \leftarrow \mathbb{D}^{e,\mathcal{L}}(T^k)$
4: $\quad$ **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
5: $\quad\quad i \leftarrow \mathrm{me}(\alpha, k)$
6: $\quad\quad$ **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
7: $\quad\quad\quad j \leftarrow \mathrm{me}(\beta, k)$
8: $\quad\quad\quad \mathbb{D}^{\mathcal{L}}(i,j) \leftarrow \mathbb{D}^{\mathcal{L}}(i,j) + \mathbb{E}(\alpha, \beta)$
9: $\quad\quad$ **end for**
10: $\quad$ **end for**
11: **end for**

? How to compute the element matrix $\mathbb{D}^{e,\mathcal{L}}(K)$?

$\forall (\alpha, \beta) \in [\![1, d+1]\!]^2$,

$$\mathbb{D}^{e,\mathcal{L}}_{\alpha,\beta}(K) = \int_K \mathcal{D}_{\mathcal{L}}(\lambda_\beta, \lambda_\alpha)(\mathrm{q}) d\mathrm{q},$$

$$= \sum_{i=1}^{d} \sum_{j=1}^{d} \int_K \mathbb{A}_{i,j} \frac{\partial \lambda_\beta}{\partial x_j} \frac{\partial \lambda_\alpha}{\partial x_i} d\mathrm{q} - \sum_{i=1}^{d} \int_K \boldsymbol{b}_i \frac{\partial \lambda_\alpha}{\partial x_i} \lambda_\beta d\mathrm{q}$$

$$+ \sum_{i=1}^{d} \int_K \boldsymbol{c}_i \frac{\partial \lambda_\beta}{\partial x_i} \lambda_\alpha d\mathrm{q} + \int_K a_0 \lambda_\alpha \lambda_\beta d\mathrm{q}.$$

$$\mathbb{D}^{e,\mathcal{L}}(K) = \sum_{i=1}^{d} \sum_{j=1}^{d} \mathbb{D}^{e}_{dudv}(K, \mathbb{A}_{i,j}, i, j) - \sum_{i=1}^{d} \mathbb{D}^{e}_{udv}(K, \boldsymbol{b}_i, i)$$

$$+ \sum_{i=1}^{d} \mathbb{D}^{e}_{duv}(K, \boldsymbol{c}_i, i) + \mathbb{D}^{e}_{uv}(K, a_0)$$

Theorem (Magic formula)

Let K be a d-simplex and $(\lambda_i(\mathrm{q}))_{i\in[\![1,d+1]\!]}$ be the barycentric coordinates of a point $\mathrm{q}\in K$, then for every $n_i\in\mathbb{N}$, $i\in[\![1,d+1]\!]$, we have

$$\int_K \prod_{i=1}^{d+1}\lambda_i^{n_i}(\mathrm{q})d\mathrm{q} = |K|\frac{d!\prod_{i=1}^{d+1} n_i!}{(d+\sum_{i=1}^{d+1} n_i)!}. \tag{1.2}$$

where $|K|$ is the volume of K.

Remark

As λ_α are polynomials of degree 1 on K, their derivatives are constant on K and we set $\lambda_{\alpha,i}^K = \frac{\partial\lambda_\alpha}{\partial x_i}_{|K}, \alpha\in[\![1,d+1]\!]$.

Computation of $\mathbb{D}^e_{udv}(K,g,i)$

$$(\mathbb{D}^e_{udv}(K,g,i))_{\alpha,\beta} \stackrel{\text{def}}{=} \int_K g(\mathrm{q})\frac{\partial \lambda_\alpha}{\partial x_i}(\mathrm{q})\lambda_\beta(\mathrm{q})d\mathrm{q} \approx \int_K g_h(\mathrm{q})\frac{\partial \lambda_\alpha}{\partial x_i}(\mathrm{q})\lambda_\beta(\mathrm{q})$$

$$\approx \lambda^K_{\alpha,i}\sum_{l=1}^{d+1}\tilde{g}_l\int_K \lambda_l(\mathrm{q})\lambda_\beta(\mathrm{q})d\mathrm{q}.$$

The magic formula gives

Lemma

$$(\mathbb{D}^e_{udv}(K,g,i))_{\alpha,\beta} \approx \frac{d!|K|}{(d+2)!}\lambda^K_{\alpha,i}(\tilde{g}_\beta + \sum_{l=1}^{d+1}\tilde{g}_l). \tag{1.3}$$

This approximation is exact if g is a polynomial of degree 1 or 0 on K. Furthermore, if $g = g_K$ is constant on K we obtain

$$(\mathbb{D}^e_{udv}(K,g,i))_{\alpha,\beta} = \frac{d!|K|}{(d+1)!}\lambda^K_{\alpha,i}g_K.$$

Computation of $\mathbb{D}^e_{udv}(K, g, i)$

$$(\mathbb{D}^e_{udv}(K, g, i))_{\alpha,\beta} \approx \frac{d!|K|}{(d+2)!} \lambda^K_{\alpha,i} (\tilde{g}_\beta + \sum_{l=1}^{d+1} \tilde{g}_l).$$

Algorithm 3 Function DElemP1_gudv. Computation of the element matrix $\mathbb{D}^e_{udv}(K, g, i)$ on a d-simplex K.

Input :

d	:	dimension of the simplex K,
vol	:	volume of the mesh element K,
$\tilde{\boldsymbol{g}}$	:	$(d+1)$-by-1 array of g values on the vertices of K,
i	:	index in $[\![1, d]\!]$,
$\boldsymbol{G}$	:	$(d$-by-$(d+1))$ local gradient array $\quad \boldsymbol{G}(i, \alpha) = \lambda^K_{\alpha,i}, \forall \alpha \in [\![1, d+1]\!]$

Output :

$\mathbb{D}^e$	:	$(d+1)$-by-$(d+1)$ matrix $\mathbb{D}^e_{duv}(K, g, i)$.

1: **Function** $\mathbb{D}^e \leftarrow$ DElemP1_gudv($d, \text{vol}, \tilde{\boldsymbol{g}}, i, \boldsymbol{G}$)
2: $\quad \tilde{g}_s \leftarrow \text{sum}(\tilde{\boldsymbol{g}})$
3: $\quad$ **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
4: $\quad\quad$ **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
5: $\quad\quad\quad \mathbb{D}^e(\alpha, \beta) \leftarrow \frac{d!}{(d+2)!} * \text{vol} * (\tilde{g}_s + \tilde{\boldsymbol{g}}(\beta)) * \boldsymbol{G}(i, \alpha)$
6: $\quad\quad$ **end for**
7: $\quad$ **end for**
8: **end Function**

$$\mathbb{D}^{e,\mathcal{L}}(K) = \sum_{i=1}^{d}\sum_{j=1}^{d} \mathbb{D}^{e}_{dudv}(K, \mathbb{A}_{i,j}, i, j) - \sum_{i=1}^{d} \mathbb{D}^{e}_{udv}(K, \boldsymbol{b}_i, i) + \sum_{i=1}^{d} \mathbb{D}^{e}_{duv}(K, \boldsymbol{c}_i, i) + \mathbb{D}^{e}_{uv}(K, a_0)$$

Algorithm 4 function DElemP1. Computation of the element matrix $\mathbb{D}^{e,\mathcal{L}}(K)$

Input :

$\mathcal{T}_h$: d-simplicial mesh structure,
$\mathcal{L}_h$: discretized $\mathcal{L}$ operator,
k : index of the k-th mesh element $K = T_k$,

Output :

$\mathbb{D}^e$: $(d+1)$-by-$(d+1)$ matrix,

```
1:  Function 𝔻^e ← DElemP1( 𝒯_h, ℒ_h, k)
2:     𝔻^e ← DElemP1_guv(d, vol, getLocFdata(ℒ_h.a0, 𝒯_h, k))
3:     if ℒ_h.order > 0 then
4:        G ← Gradients(d, 𝒯_h.q(:, 𝒯_h.me(:, k)), 𝒯_h.vols(k))
5:        for i ← 1 to d do
6:           for j ← 1 to d do
7:              𝔻^e ← 𝔻^e + DElemP1_gdudv(d, vol, getLocFdata(ℒ_h.A(i,j), 𝒯_h, k), i, j, G)
8:           end for
9:           𝔻^e ← 𝔻^e − DElemP1_gudv(d, vol, getLocFdata(ℒ_h.b(i), 𝒯_h, k), i, G)
10:          𝔻^e ← 𝔻^e + DElemP1_gduv(d, vol, getLocFdata(ℒ_h.c(i), 𝒯_h, k), i, G)
11:       end for
12:    end if
13: end Function
```

Assembly of the matrix $\mathbb{D}^{\mathcal{L}}$

1: **Function** $\mathbb{D}^{\mathcal{L}} \leftarrow$ DAssemblyP1_base$(\mathcal{T}_h, \mathcal{L})$
2: $\quad \mathbb{D}^{\mathcal{L}} \leftarrow$ Sparse$(\mathcal{T}_h.\mathrm{n_q}, \mathcal{T}_h.\mathrm{n_q})$
3: $\quad \mathcal{L}_h \leftarrow$ setDdata$(\mathcal{L}, \mathcal{T}_h)$
4: $\quad$ **for** $k \leftarrow 1$ **to** $\mathcal{T}_h.\mathrm{n_{me}}$ **do**
5: $\quad\quad \mathbb{D}^e \leftarrow$ DElemP1$(\mathcal{T}_h, \mathcal{L}_h, k)$
6: $\quad\quad$ **for** $\alpha \leftarrow 1$ **to** $\mathcal{T}_h.d + 1$ **do**
7: $\quad\quad\quad i \leftarrow \mathcal{T}_h.\mathrm{me}(\alpha, k)$
8: $\quad\quad\quad$ **for** $\beta \leftarrow 1$ **to** $\mathcal{T}_h.d + 1$ **do**
9: $\quad\quad\quad\quad j \leftarrow \mathcal{T}_h.\mathrm{me}(\beta, k)$
10: $\quad\quad\quad\quad \mathbb{D}^{\mathcal{L}}(i,j) \leftarrow \mathbb{D}^{\mathcal{L}}(i,j) + \mathbb{D}^e(\alpha,\beta)$
11: $\quad\quad\quad$ **end for**
12: $\quad\quad$ **end for**
13: $\quad$ **end for**
14: **end Function**

- Computation of the *Mass* matrix $\mathbb{M}$

$$\mathbb{M}_{i,j} = \int_{\Omega_h} \varphi_j \varphi_i d\mathrm{q} = \mathbb{D}^{\mathcal{L}}_{i,j}, \text{ with } \mathcal{L} = \mathcal{L}_{\mathbb{O},\mathbf{0},\mathbf{0},1}$$

1: LMass $\leftarrow$ Loperator$(d, \mathbb{O}_{d\times d}, \mathbf{0}_d, \mathbf{0}_d, 1)$
2: $\mathbb{M} \leftarrow$ DAssemblyP1_base$(\mathcal{T}_h, \mathtt{LMass})$

Assembly of the matrix $\mathbb{D}^{\mathcal{L}}$

1: **Function** $\mathbb{D}^{\mathcal{L}} \leftarrow$ DAssemblyP1_base$(\mathcal{T}_h, \mathcal{L})$
2: $\quad \mathbb{D}^{\mathcal{L}} \leftarrow$ Sparse$(\mathcal{T}_h.\mathrm{n_q}, \mathcal{T}_h.\mathrm{n_q})$
3: $\quad \mathcal{L}_h \leftarrow$ setDdata$(\mathcal{L}, \mathcal{T}_h)$
4: $\quad$ **for** $k \leftarrow 1$ **to** $\mathcal{T}_h.\mathrm{n_{me}}$ **do**
5: $\quad\quad \mathbb{D}^{e} \leftarrow$ DElemP1$(\mathcal{T}_h, \mathcal{L}_h, k)$
6: $\quad\quad$ **for** $\alpha \leftarrow 1$ **to** $\mathcal{T}_h.d + 1$ **do**
7: $\quad\quad\quad i \leftarrow \mathcal{T}_h.\mathrm{me}(\alpha, k)$
8: $\quad\quad\quad$ **for** $\beta \leftarrow 1$ **to** $\mathcal{T}_h.d + 1$ **do**
9: $\quad\quad\quad\quad j \leftarrow \mathcal{T}_h.\mathrm{me}(\beta, k)$
10: $\quad\quad\quad\quad \mathbb{D}^{\mathcal{L}}(i,j) \leftarrow \mathbb{D}^{\mathcal{L}}(i,j) + \mathbb{D}^{e}(\alpha,\beta)$
11: $\quad\quad\quad$ **end for**
12: $\quad\quad$ **end for**
13: $\quad$ **end for**
14: **end Function**

- Computation of the *Stiffness* matrix $\mathbb{S}$

$$\mathbb{S}_{i,j} = \int_{\Omega_h} \langle \nabla \varphi_j, \nabla \varphi_i \rangle \, d\mathrm{q} = \mathbb{D}^{\mathcal{L}}_{i,j}, \text{ with } \mathcal{L} = \mathcal{L}_{\mathbb{I},\mathbf{0},\mathbf{0},0}$$

1: LStiff $\leftarrow$ Loperator$(d, \mathbb{I}_{d\times d}, \mathbf{0}_d, \mathbf{0}_d, 0)$
2: $\mathbb{S} \leftarrow$ DAssemblyP1_base$(\mathcal{T}_h,$ LStiff$)$

Algorithm 2 Steps for solving the *Scalar* Boundary Value Problem

1: Assembly of the matrix $\mathbb{D}^{\mathcal{L}}$
2: Computation of $\boldsymbol{b}^f$
3: Computation of *Robin* contributions $\mathbb{A}^R$ and $\boldsymbol{b}^R$
4: Computation of *Dirichlet* contributions $\mathcal{I}_D$, $\mathcal{I}_D^c$ and $\boldsymbol{R}$.
5: $\boldsymbol{b}^{\mathcal{L}} \leftarrow \boldsymbol{b}^f + \boldsymbol{b}^R$
6: $\mathbb{A}^{\mathcal{L}} \leftarrow \mathbb{D}^{\mathcal{L}} + \mathbb{A}^R$
7: $\boldsymbol{u}(\mathcal{I}_D^c) \leftarrow \text{Solve}(\mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D^c), \boldsymbol{b}(\mathcal{I}_D^c) - \mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D)\boldsymbol{R}(\mathcal{I}_D))$
8: $\boldsymbol{u}(\mathcal{I}_D) \leftarrow \boldsymbol{R}(\mathcal{I}_D)$

$$\boldsymbol{b}_i^f \overset{\text{def}}{=} \int_{\Omega_h} f\varphi_i d\mathrm{q} \approx \int_{\Omega_h} f_h\varphi_i d\mathrm{q} = \sum_{j=1}^{\mathrm{n_q}} \boldsymbol{f}_j \int_{\Omega_h} \varphi_j\varphi_i d\mathrm{q}$$

1: **Function** $\boldsymbol{b}^f \leftarrow \text{RHS}(\mathcal{T}_h, f)$
2: $\quad \mathrm{DMass} \leftarrow \text{Loperator}(\mathcal{T}_h.d, \mathbb{O}, \mathbf{0}, \mathbf{0}, 1)$
3: $\quad \mathbb{M} \leftarrow \text{DAssemblyP1\_base}(\mathcal{T}_h, \mathrm{DMass})$
4: $\quad \boldsymbol{b}^f \leftarrow \mathbb{M} * \text{setFdata}(f, \mathcal{T}_h)$
5: **end Function**

Algorithm 2 Steps for solving the *Scalar* Boundary Value Problem

1: Assembly of the matrix $\mathbb{D}^{\mathcal{L}}$
2: Computation of $\boldsymbol{b}^f$
3: Computation of *Robin* contributions $\mathbb{A}^R$ and $\boldsymbol{b}^R$
4: Computation of *Dirichlet* contributions $\mathcal{I}_D$, $\mathcal{I}_D^c$ and $\boldsymbol{R}$.
5: $\boldsymbol{b}^{\mathcal{L}} \leftarrow \boldsymbol{b}^f + \boldsymbol{b}^R$
6: $\mathbb{A}^{\mathcal{L}} \leftarrow \mathbb{D}^{\mathcal{L}} + \mathbb{A}^R$
7: $\boldsymbol{u}(\mathcal{I}_D^c) \leftarrow \text{Solve}(\mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D^c), \boldsymbol{b}(\mathcal{I}_D^c) - \mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D)\boldsymbol{R}(\mathcal{I}_D))$
8: $\boldsymbol{u}(\mathcal{I}_D) \leftarrow \boldsymbol{R}(\mathcal{I}_D)$

$$\boldsymbol{b}_i^R \overset{\text{def}}{=} \int_{\Gamma_h^R} g^R \times \mathcal{T}_h.\varphi_i d\sigma = \sum_{l \in \mathcal{I}_{\text{labels}}^R} \int_{\mathcal{B}_h(l)} g^R \times \mathcal{T}_h.\varphi_i d\sigma = \sum_{l \in \mathcal{I}_{\text{labels}}^R} \boldsymbol{b}_i^l$$

$$\boldsymbol{b}_{\mathcal{I}_l^c}^l = 0 \quad \text{and} \quad \boldsymbol{b}_{\mathcal{I}_l(s)}^l \approx \sum_{r=1}^{\mathcal{B}_h(l).\mathrm{n_q}} g_r^l \int_{\mathcal{B}_h(l)} \mathcal{B}_h(l).\varphi_r \times \mathcal{B}_h(l).\varphi_s d\sigma.$$

1: $\mathrm{LMass} \leftarrow \text{Loperator}(\mathrm{d}, \mathbb{O}, \mathbf{0}, \mathbf{0}, 1)$
2: $\mathbb{M}^l \leftarrow \text{DAssemblyP1\_base}(\mathcal{B}_h(l), \mathrm{LMass})$
3: $\boldsymbol{b}^l(\mathcal{I}_l) \leftarrow \mathbb{M}^l \boldsymbol{g}^l, \quad \boldsymbol{b}^l(\mathcal{I}_l^c) \leftarrow 0$

Algorithm 2 Steps for solving the *Scalar* Boundary Value Problem

1: Assembly of the matrix $\mathbb{D}^{\mathcal{L}}$
2: Computation of $\boldsymbol{b}^f$
3: Computation of *Robin* contributions $\mathbb{A}^R$ and $\boldsymbol{b}^R$
4: Computation of *Dirichlet* contributions $\mathcal{I}_D$, $\mathcal{I}_D^c$ and $\boldsymbol{R}$.
5: $\boldsymbol{b}^{\mathcal{L}} \leftarrow \boldsymbol{b}^f + \boldsymbol{b}^R$
6: $\mathbb{A}^{\mathcal{L}} \leftarrow \mathbb{D}^{\mathcal{L}} + \mathbb{A}^R$
7: $\boldsymbol{u}(\mathcal{I}_D^c) \leftarrow \text{Solve}(\mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D^c), \boldsymbol{b}(\mathcal{I}_D^c) - \mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D)\boldsymbol{R}(\mathcal{I}_D))$
8: $\boldsymbol{u}(\mathcal{I}_D) \leftarrow \boldsymbol{R}(\mathcal{I}_D)$

$$\mathbb{A}^R_{i,j} \overset{\text{def}}{=} \int_{\Gamma_h^R} a^R \varphi_j \varphi_i d\sigma = \sum_{l \in \mathcal{I}^R_{\text{labels}}} \int_{\mathcal{B}_h(l)} a^R \times \mathcal{T}_h.\varphi_j \times \mathcal{T}_h.\varphi_i d\sigma = \sum_{l \in \mathcal{I}^R_{\text{labels}}} \mathbb{A}^l_{i,j}$$

$$\mathbb{A}^l_{\mathcal{I}_l(r),\mathcal{I}_l(s)} = \int_{\mathcal{B}_h(l)} a^R \times \mathcal{B}_h(l).\varphi_s \times \mathcal{B}_h(l).\varphi_r d\sigma$$

1: $\text{LWMass} \leftarrow \text{Loperator}(\text{d}, \mathbb{O}, \boldsymbol{0}, \boldsymbol{0}, a^R)$
2: $\mathbb{A}^l \leftarrow 0,$
3: $\mathbb{A}^l(\mathcal{I}_l, \mathcal{I}_l) \leftarrow \text{DAssemblyP1\_base}(\mathcal{B}_h(l), \text{LWMass})$

Algorithm 5 function SolvePDE : building and solving a scalar BVP

Input :

pde : a PDE structure

Output :

$\boldsymbol{u}$: vector of dimension $\mathrm{n_{dof}} = \mathrm{pde}.\mathcal{T}_h.\mathrm{n_q}$.

Function $\boldsymbol{u} \leftarrow \mathrm{SolvePDE}(\mathrm{pde})$
  $\mathbb{D}^{\mathcal{L}} \leftarrow \mathrm{DAssemblyP1\_base}(\mathrm{pde}.\mathcal{T}_h, \mathrm{pde.op})$
  $\boldsymbol{b}^f \leftarrow \mathrm{RHS}(\mathrm{pde}.\mathcal{T}_h, \mathrm{pde.f})$
  $[\boldsymbol{b}^R, \mathbb{M}^R] \leftarrow \mathrm{RobinBC}(\mathrm{pde})$
  $[\mathcal{I}_D, \mathcal{I}_D^c, \boldsymbol{R}] \leftarrow \mathrm{DirichletBC}(\mathrm{pde})$
  $\boldsymbol{b}^{\mathcal{L}} \leftarrow \boldsymbol{b}^f + \boldsymbol{b}^R$
  $\mathbb{A}^{\mathcal{L}} \leftarrow \mathbb{D}^{\mathcal{L}} + \mathbb{A}^R$
  $\boldsymbol{u}(\mathcal{I}_D^c) \leftarrow \mathrm{Solve}(\mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D^c), \boldsymbol{b}(\mathcal{I}_D^c) - \mathbb{A}^{\mathcal{L}}(\mathcal{I}_D^c, \mathcal{I}_D)\boldsymbol{R}(\mathcal{I}_D))$
  $\boldsymbol{u}(\mathcal{I}_D) \leftarrow \boldsymbol{R}(\mathcal{I}_D)$
end Function

Algorithms efficiency

The algorithms contain few lines

	algorithm	Matlab	Python	Scilab
RHS	5	9	4	5
RobinBC	19	21	21	21
DirichletBC	13	21	11	21
SolvePDE	10	11	10	11
DAssemblyP1_base	14	15	17	15
DElemP1	13	24	25	25
DElemP1_guv	8	8	7	8
DElemP1_gdudv	8	8	6	8
DElemP1_gudv	8	8	6	8
DElemP1_gduv	8	8	6	8
Gradients	5	6	11	6
setFdata	5	18	25	9
BuildBoundaryMeshes	22	18	30	18
ComputeVolVec	15	30	13	30
Loperator	16	12	30	11
setFdata	5	18	25	9
getLocFdata	3	3	2	3
initPDE	24	8	19	26
setBC_PDE	12	28	30	24
Total	208	256	273	282

Input/output functions were not counted.

Benchmark of stationary convection-diffusion problem

n_{dof}	Matlab		Octave		Python		FreeFEM++	
Classic	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)
2448	23.132	0.022	44.350	0.010	7.619	0.012	0.082	0.002
6456	62.699	0.080	119.240	0.027	20.384	0.029	0.085	0.003
12205	121.701	0.079	229.142	0.056	38.651	0.061	0.162	0.004
25253	267.526	0.256	494.841	0.135	80.334	0.136	0.339	0.009
55996	642.744	0.576	1197.113	0.348	178.354	0.358	0.754	0.023
98712	1434.475	0.868	2555.707	0.725	315.874	0.741	1.351	0.064
118216	1602.288	1.424	3397.917	0.918	379.210	0.941	1.621	0.079

- Matlab R2014b
- Octave 3.8.1
- Python 3.4.0 (NumPy 1.8.2, SciPy 0.13.3)
- FreeFEM++ 3.31-3

Computer characteristics: 2 x Intel Xeon E5-2630v2 (6 cores) at 2.60Ghz, 64Go RAM under Ubuntu 14.04 LTS (64-bit)

Classical algorithm : conclusion

- **Pros**
  - easy to write
  - very short codes
  - can (try to) solve a wide variety of BVP's
- **Cons**
  - So loooong computation to assemble the linear system!!!

What can we do?
Find another way to assembly the linear system :

`OptV3` **vectorized algorithms!**

Outline

Why classical algorithm is so long?

Algorithm 6 Classical assembly with element matrices

```
1:  D^L ← 0                                              ▷ Sparse matrix n_q-by-n_q
2:  for k ← 1 to n_me do
3:      E ← D^{e,L}(T^k)
4:      for α ← 1 to d + 1 do
5:          i ← me(α, k)
6:          for β ← 1 to d + 1 do
7:              j ← me(β, k)
8:              D^L(i, j) ← D^L(i, j) + E(α, β)          ▷ insert/add
9:          end for
10:     end for
11: end for
```

- Due to the sparse matrix storage format (CSR,CSC,HYB,...), insertions of elements are very expensive.
- Some dynamic reallocation problems may also occur.

See report in collaboration with Japhet C. and Scarella G.

Definition (Usage of the sparse function)

$$\texttt{M} \leftarrow \text{sparse}(\texttt{Ig}, \texttt{Jg}, \texttt{Kg}, \texttt{m}, \texttt{n})$$

- M is a m $\times$n sparse matrix such that
$$\texttt{M}(\texttt{Ig}(k), \texttt{Jg}(k)) \leftarrow \texttt{M}(\texttt{Ig}(k), \texttt{Jg}(k)) + \texttt{Kg}(k).$$
- The 1d-arrays Ig, Jg and Kg have the same length.
- The elements of Kg having the same indices in Ig and Jg are summed.

Examples in some languages :

- Python : M=sparse.<format>_matrix((Kg,(Ig,Jg)),shape=(m,n)) (csc, csr, lil, ...),
- Matlab : M=sparse(Ig,Jg,Kg,m,n), only csc format,
- Octave : M=sparse(Ig,Jg,Kg,m,n), only csc format,
- Scilab : M=sparse([Ig,Jg],Kg,[m,n]), only row-by-row format,
- Julia : M=sparse(Ig, Jg, Kg, m, n) , csc format,
- R : M <- sparseMatrix(Ig, Jg, x = Kg, dims = c(m,n)), csc format,
- C : see cs_compress of the *SuiteSparse* developed by T. Davis,
- CUDA : see the *Thrust* and *Cusp* libraries.

Definition (Usage of the sparse function)

$$\texttt{M} \leftarrow \text{sparse}(\texttt{Ig}, \texttt{Jg}, \texttt{Kg}, \texttt{m}, \texttt{n})$$

- M is a m $\times$n sparse matrix such that
$$\texttt{M}(\texttt{Ig}(k), \texttt{Jg}(k)) \leftarrow \texttt{M}(\texttt{Ig}(k), \texttt{Jg}(k)) + \texttt{Kg}(k).$$
- The 1d-arrays Ig, Jg and Kg have the same length.
- The elements of Kg having the same indices in Ig and Jg are summed.

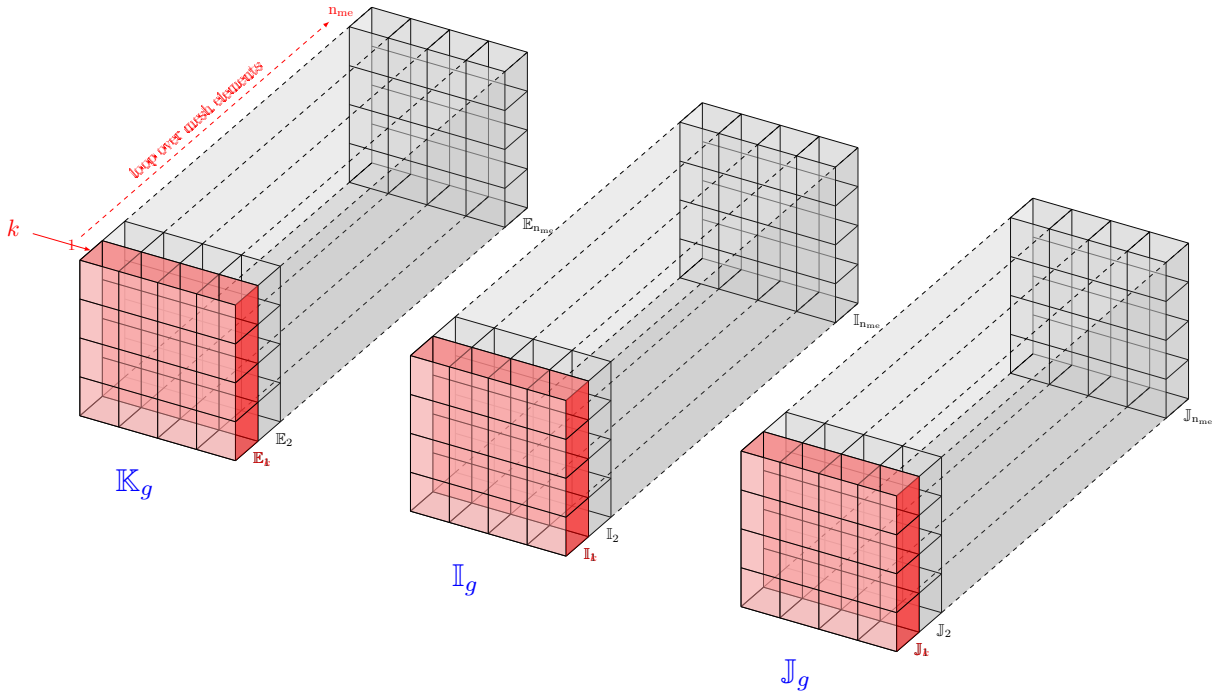
Examples in some languages :

- Python : `M=sparse.<format>_matrix((Kg,(Ig,Jg)),shape=(m,n))` (csc, csr, lil, ...),
- Matlab : `M=sparse(Ig,Jg,Kg,m,n)`, only csc format,
- Octave : `M=sparse(Ig,Jg,Kg,m,n)`, only csc format,
- Scilab : `M=sparse([Ig,Jg],Kg,[m,n])`, only row-by-row format,
- Julia : `M=sparse(Ig, Jg, Kg, m, n)` , csc format,
- R : `M <- sparseMatrix(Ig, Jg, x = Kg, dims = c(m,n) )`, csc format,
- C : see cs_compress of the *SuiteSparse* developed by T. Davis,
- CUDA : see the *Thrust* and *Cusp* libraries.

3d-OptV1 algorithm

better usage of the sparse function :

1: Creation of the 3d-arrays $\mathbb{I}_g$, $\mathbb{J}_g$ and $\mathbb{K}_g$
2: $\mathbb{M} \leftarrow \text{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$



3*d*-OptV1 algorithm

Algorithm 7 *3d*-OptV1 algorithm

1: $\mathbb{K}_g \leftarrow \mathbb{I}_g \leftarrow \mathbb{J}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}} \times (d+1) \times (d+1)}$ ▷ 3d arrays contiguous in memory
2: **for** $k \leftarrow 1$ **to** $\mathrm{n_{me}}$ **do** ▷ Loop over mesh elements
3: $\quad \mathbb{E} \leftarrow \mathbb{D}^{e,\mathcal{L}}(T^k)$
4: $\quad$ **for** $\alpha \leftarrow 1$ **to** $d+1$ **do** ▷ Loop over local nodes
5: $\quad\quad$ **for** $\beta \leftarrow 1$ **to** $d+1$ **do** ▷ Loop over local nodes
6: $\quad\quad\quad \mathbb{K}_g(k,\alpha,\beta) \leftarrow \mathbb{E}(\alpha,\beta)$
7: $\quad\quad\quad \mathbb{I}_g(k,\alpha,\beta) \leftarrow \mathrm{me}(\alpha,k)$
8: $\quad\quad\quad \mathbb{J}_g(k,\alpha,\beta) \leftarrow \mathrm{me}(\beta,k)$
9: $\quad\quad$ **end for**
10: $\quad$ **end for**
11: **end for**
12: $\mathbb{M} \leftarrow \mathrm{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$

Performance : similar to the classical version!

Transformation of the 3d-OptV1 algorithm

1: $\mathbb{I}_g, \mathbb{J}_g, \mathbb{K}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}} \times (d+1) \times (d+1)}$
2: **for** $k \leftarrow 1$ **to** $\mathrm{n_{me}}$ **do**
3: $\quad \mathbb{E} \leftarrow \mathbb{D}^{e,\mathcal{L}}(T^k)$
4: $\quad$ **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
5: $\quad\quad$ **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
6: $\quad\quad\quad \mathbb{K}_g(k,\alpha,\beta) \leftarrow \mathbb{E}(\alpha,\beta)$
7: $\quad\quad\quad \mathbb{I}_g(k,\alpha,\beta) \leftarrow \mathrm{me}(\alpha,k)$
8: $\quad\quad\quad \mathbb{J}_g(k,\alpha,\beta) \leftarrow \mathrm{me}(\beta,k)$
9: $\quad\quad$ **end for**
10: $\quad$ **end for**
11: **end for**
12: $\mathbb{M} \leftarrow \mathsf{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$

- There exists a function EMent which returns the (α,β) entry of the matrix $\mathbb{D}^{e,\mathcal{L}}(T^k)$.
- Loops permutation

Transformation of the 3d-OptV1 algorithm

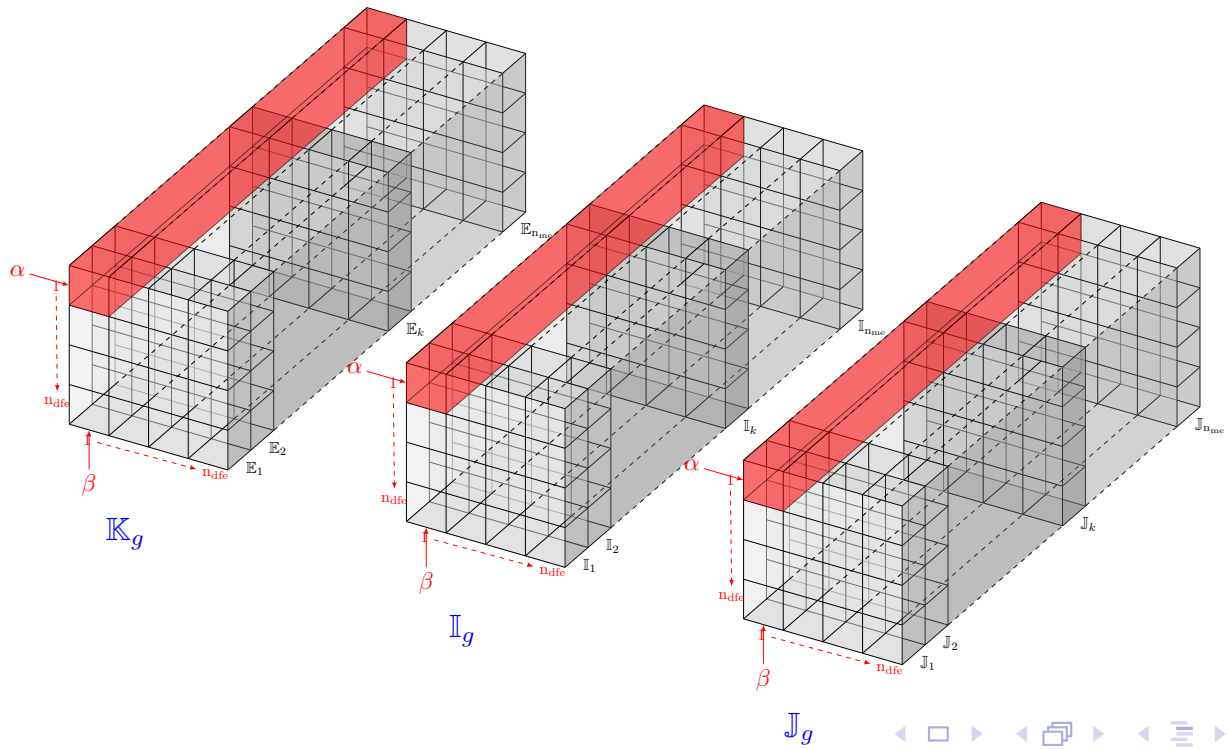
1: $\mathbb{I}_g, \mathbb{J}_g, \mathbb{K}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}} \times (d+1) \times (d+1)}$
2: **for** $k \leftarrow 1$ **to** $\mathrm{n_{me}}$ **do**
3: **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
4: **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
5: $\mathbb{K}_g(k, \alpha, \beta) \leftarrow \mathrm{EMent}(\alpha, \beta, \mathrm{vol}(k), \ldots)$
6: $\mathbb{I}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\alpha, k)$
7: $\mathbb{J}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\beta, k)$
8: **end for**
9: **end for**
10: **end for**
11: $\mathbb{M} \leftarrow \mathrm{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$

- There exists a function EMent which returns the (α, β) entry of the matrix $\mathbb{D}^{e,\mathcal{L}}(T^k)$.
- Loops permutation

Transformation of the 3d-OptV1 algorithm

1: $\mathbb{I}_g, \mathbb{J}_g, \mathbb{K}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}} \times (d+1) \times (d+1)}$
2: **for** $k \leftarrow 1$ **to** $\mathrm{n_{me}}$ **do**
3: **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
4: **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
5: $\mathbb{K}_g(k, \alpha, \beta) \leftarrow \mathsf{EMent}(\alpha, \beta, \mathrm{vol}(k), \ldots)$
6: $\mathbb{I}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\alpha, k)$
7: $\mathbb{J}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\beta, k)$
8: **end for**
9: **end for**
10: **end for**
11: $\mathbb{M} \leftarrow \mathsf{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$

1: $\mathbb{I}_g, \mathbb{J}_g, \mathbb{K}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}} \times (d+1) \times (d+1)}$
2: **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
3: **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
4: **for** $k \leftarrow 1$ **to** $\mathrm{n_{me}}$ **do**
5: $\mathbb{K}_g(k, \alpha, \beta) \leftarrow \mathsf{EMent}(\alpha, \beta, \mathrm{vol}(k), \ldots)$
6: $\mathbb{I}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\alpha, k)$
7: $\mathbb{J}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\beta, k)$
8: **end for**
9: **end for**
10: **end for**
11: $\mathbb{M} \leftarrow \mathsf{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$



Vectorization of the 3d-OptV1 algorithm

1: $\mathbb{I}_g, \mathbb{J}_g, \mathbb{K}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}} \times (d+1) \times (d+1)}$
2: **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
3: **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
4: **for** $k \leftarrow 1$ **to** $\mathrm{n_{me}}$ **do**
5: $\mathbb{K}_g(k, \alpha, \beta) \leftarrow \mathsf{EMent}(\alpha, \beta, \mathrm{vol}(k), \ldots)$
6: $\mathbb{I}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\alpha, k)$
7: $\mathbb{J}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\beta, k)$
8: **end for**
9: **end for**
10: **end for**
11: $\mathbb{M} \leftarrow \mathsf{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$

Vectorization of the 3d-OptV1 algorithm

1: $\mathbb{I}_g, \mathbb{J}_g, \mathbb{K}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}} \times (d+1) \times (d+1)}$
2: **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
3: **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
4: **for** $k \leftarrow 1$ **to** $\mathrm{n_{me}}$ **do**
5: $\mathbb{K}_g(k, \alpha, \beta) \leftarrow \mathsf{EMent}(\alpha, \beta, \mathrm{vol}(k), \ldots)$
6: $\mathbb{I}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\alpha, k)$
7: $\mathbb{J}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\beta, k)$
8: **end for**
9: **end for**
10: **end for**
11: $\mathbb{M} \leftarrow \mathsf{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$

1: $\mathbb{I}_g, \mathbb{J}_g, \mathbb{K}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}} \times (d+1) \times (d+1)}$
2: **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
3: **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
4: **for** $k \leftarrow 1$ **to** $\mathrm{n_{me}}$ **do**
5: $\mathbb{K}_g(k, \alpha, \beta) \leftarrow \mathsf{EMent}(\alpha, \beta, \mathrm{vol}(k), \ldots)$
6: **end for**
7: $\mathbb{I}_g(:, \alpha, \beta) \leftarrow \mathrm{me}(\alpha, :)$
8: $\mathbb{J}_g(:, \alpha, \beta) \leftarrow \mathrm{me}(\beta, :)$
9: **end for**
10: **end for**
11: $\mathbb{M} \leftarrow \mathsf{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$

Vectorization of the 3d-OptV1 algorithm

1: $\mathbb{I}_g, \mathbb{J}_g, \mathbb{K}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}} \times (d+1) \times (d+1)}$
2: **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
3: **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
4: **for** $k \leftarrow 1$ **to** $\mathrm{n_{me}}$ **do**
5: $\mathbb{K}_g(k, \alpha, \beta) \leftarrow \mathrm{EMent}(\alpha, \beta, \mathrm{vol}(k), \ldots)$
6: $\mathbb{I}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\alpha, k)$
7: $\mathbb{J}_g(k, \alpha, \beta) \leftarrow \mathrm{me}(\beta, k)$
8: **end for**
9: **end for**
10: **end for**
11: $\mathbb{M} \leftarrow \mathrm{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$

Algorithm 10 Generic OptV3 algorithm

1: $\mathbb{I}_g, \mathbb{J}_g, \mathbb{K}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}} \times (d+1) \times (d+1)}$
2: **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
3: **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
4: $\mathbb{K}_g(:, \alpha, \beta) \leftarrow \mathrm{ElemMatVec}(\alpha, \beta, \mathrm{vol}, \ldots)$
5: $\mathbb{I}_g(:, \alpha, \beta) \leftarrow \mathrm{me}(\alpha, :)$
6: $\mathbb{J}_g(:, \alpha, \beta) \leftarrow \mathrm{me}(\beta, :)$
7: **end for**
8: **end for**
9: $\mathbb{M} \leftarrow \mathrm{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$

1: $\mathbb{I}_g, \mathbb{J}_g, \mathbb{K}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}} \times (d+1) \times (d+1)}$
2: **for** $\alpha \leftarrow 1$ **to** $d+1$ **do**
3: **for** $\beta \leftarrow 1$ **to** $d+1$ **do**
4: **for** $k \leftarrow 1$ **to** $\mathrm{n_{me}}$ **do**
5: $\mathbb{K}_g(k, \alpha, \beta) \leftarrow \mathrm{EMent}(\alpha, \beta, \mathrm{vol}(k), \ldots)$
6: **end for**
7: $\mathbb{I}_g(:, \alpha, \beta) \leftarrow \mathrm{me}(\alpha, :)$
8: $\mathbb{J}_g(:, \alpha, \beta) \leftarrow \mathrm{me}(\beta, :)$
9: **end for**
10: **end for**
11: $\mathbb{M} \leftarrow \mathrm{sparse}(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathrm{n_q}, \mathrm{n_q})$

Vectorized computation of $\mathbb{K}_g$

$$\begin{aligned}\mathbb{K}_g(k,:,:) = (\mathbb{K}_g)_k =& \mathbb{D}^{e,\mathcal{L}}(T_k) \\ &= \sum_{i=1}^{d}\sum_{j=1}^{d} \mathbb{D}^{e}_{dudv}(T_k, \mathbb{A}_{i,j}, i, j) - \sum_{i=1}^{d} \mathbb{D}^{e}_{udv}(T_k, \boldsymbol{b}_i, i) \\ &\quad + \sum_{i=1}^{d} \mathbb{D}^{e}_{duv}(T_k, \boldsymbol{c}_i, i) + \mathbb{D}^{e}_{uv}(T_k, a_0)\end{aligned}$$

$$\mathbb{K}_g = \sum_{i=1}^{d}\sum_{j=1}^{d} \mathbb{K}_{dudv}(\mathbb{A}_{i,j}, i, j) - \sum_{i=1}^{d} \mathbb{K}_{udv}(\boldsymbol{b}_i, i) + \sum_{i=1}^{d} \mathbb{K}_{duv}(\boldsymbol{c}_i, i) + \mathbb{K}_{uv}(a_0)$$

$$\mathbb{K}_g = \sum_{i=1}^{d}\sum_{j=1}^{d} \mathbb{K}_{dudv}(\mathbb{A}_{i,j}, i, j) - \sum_{i=1}^{d} \mathbb{K}_{udv}(\boldsymbol{b}_i, i) + \sum_{i=1}^{d} \mathbb{K}_{duv}(\boldsymbol{c}_i, i) + \mathbb{K}_{uv}(a_0)$$

Algorithm 11 KgP1_OptV3 function : computation of $\mathbb{K}_g$

Input :

$\mathcal{T}_h$: d-simplicial mesh structure associated to Ω_h

$\mathcal{L}$: Loperator structure

$\mathbb{G}$: gradient array ($\mathrm{n_{me}}$-by-$(d+1)$-by-d), $\mathbb{G}(k, \alpha, :) = \nabla \lambda_\alpha^k(\mathrm{q})$, $\forall \alpha \in [\![1, d+1]\!]$

Output :

$\mathbb{K}_g$: $\mathrm{n_{me}}$-by-$(d+1)$-by-$(d+1)$ array

Function $\mathbb{K}_g \leftarrow$ KgP1_OptV3$(\mathcal{T}_h, \mathcal{L}, \mathbb{G})$
  $\mathbb{K}_g \leftarrow$ KgP1_OptV3_guv$(\mathcal{T}_h, \mathcal{L}.a0)$
  for $i \leftarrow 1$ **to** $\mathcal{T}_h.d$ **do**
    for $j \leftarrow 1$ **to** $\mathcal{T}_h.d$ **do**
      $\mathbb{K}_g \leftarrow \mathbb{K}_g +$ KgP1_OptV3_gdudv$(\mathcal{T}_h, \mathcal{L}.A(i,j), \mathbb{G}, i, j)$
    end for
    $\mathbb{K}_g \leftarrow \mathbb{K}_g -$ KgP1_OptV3_gudv$(\mathcal{T}_h, \mathcal{L}.b(i), \mathbb{G}, i)$
    $\mathbb{K}_g \leftarrow \mathbb{K}_g +$ KgP1_OptV3_gduv$(\mathcal{T}_h, \mathcal{L}.c(i), \mathbb{G}, i)$
  end for
end Function

Computation of the 3d-array $\mathbb{K}_{udv}(g,i)$

By definition

$$(\mathbb{K}_{udv}(g,i))_k = \mathbb{D}^e_{udv}(T_k,g,i) = \left(\int_{T_k} g\frac{\partial\lambda^k_\alpha}{\partial x_i}\lambda^k_\beta d\mathrm{q}\right)_{\alpha,\beta=1}^{d+1}$$

We recall equation (1.3)

$$(\mathbb{D}^e_{udv}(K,g,i))_{\alpha,\beta} \approx \frac{d!|K|}{(d+2)!}(\tilde{g}_\beta + \sum_{l=1}^{d+1}\tilde{g}_l)\frac{\partial\lambda_\alpha}{\partial x_i}$$

We set $\boldsymbol{g}$ (1-by-$\mathrm{n_q}$), $\mathrm{g_{me}}$ ($(d+1)$-by-$\mathrm{n_{me}}$) and $\boldsymbol{g}^s$ (1-by-$\mathrm{n_{me}}$)

$$\boldsymbol{g}(i) = g(\mathrm{q}^i), \qquad \mathrm{g_{me}}(\alpha,k) = \boldsymbol{g}(\mathrm{me}(\alpha,k)), \qquad \boldsymbol{g}^s(k) = \sum_{\alpha=1}^{d+1}\mathrm{g_{me}}(\alpha,k)$$

So

$$(\mathbb{K}_{udv}(g,i))_{k,\alpha,\beta} \approx \frac{d!|T_k|}{(d+2)!}(\mathrm{g}_{me}(\beta,k)+\boldsymbol{g}^s(k))\times\mathbb{G}(k,\alpha,i)$$

We obtain the vectorized computation of $(\mathbb{K}_{udv}(g))_{:,\alpha,\beta}$:

$$(\mathbb{K}_{udv}(g,i))_{:,\alpha,\beta} \approx \frac{d!}{(d+2)!}\times\mathcal{T}_h.\mathrm{vols}\times(\mathrm{g}_{me}(\beta,:)+\boldsymbol{g}^s)\times\mathbb{G}(:,\alpha,i) \qquad (2.1)$$

Computation of the 3d-array $\mathbb{K}_{udv}(g,i)$

By definition

$$(\mathbb{K}_{udv}(g,i))_k = \mathbb{D}^e_{udv}(T_k,g,i) = \left(\int_{T_k} g\frac{\partial\lambda^k_\alpha}{\partial x_i}\lambda^k_\beta d\mathrm{q}\right)_{\alpha,\beta=1}^{d+1}$$

We recall equation (1.3)

$$(\mathbb{D}^e_{udv}(K,g,i))_{\alpha,\beta} \approx \frac{d!|K|}{(d+2)!}(\tilde{g}_\beta + \sum_{l=1}^{d+1}\tilde{g}_l)\frac{\partial\lambda_\alpha}{\partial x_i}$$

We set $\boldsymbol{g}$ (1-by-$\mathrm{n_q}$), $\mathrm{g_{me}}$ ($(d+1)$-by-$\mathrm{n_{me}}$) and $\boldsymbol{g}^s$ (1-by-$\mathrm{n_{me}}$)

$$\boldsymbol{g}(i) = g(\mathrm{q}^i), \qquad \mathrm{g_{me}}(\alpha,k) = \boldsymbol{g}(\mathrm{me}(\alpha,k)), \qquad \boldsymbol{g}^s(k) = \sum_{\alpha=1}^{d+1}\mathrm{g_{me}}(\alpha,k)$$

So

$$(\mathbb{K}_{udv}(g,i))_{k,\alpha,\beta} \approx \frac{d!|T_k|}{(d+2)!}(\mathrm{g}_{me}(\beta,k) + \boldsymbol{g}^s(k)) \times \mathbb{G}(k,\alpha,i)$$

We obtain the vectorized computation of $(\mathbb{K}_{udv}(g))_{:,\alpha,\beta}$:

$$(\mathbb{K}_{udv}(g,i))_{:,\alpha,\beta} \approx \frac{d!}{(d+2)!} \times \mathcal{T}_h.\mathrm{vols} \times (\mathrm{g}_{me}(\beta,:) + \boldsymbol{g}^s) \times \mathbb{G}(:,\alpha,i) \qquad (2.1)$$

Computation of $\mathbb{K}_{udv}(g,i)$

$$(\mathbb{K}_{udv}(g,i))_{:,\alpha,\beta} \approx \frac{d!}{(d+2)!} \times \mathcal{T}_h.\text{vols} \times (\mathrm{g}_{me}(\beta,:) + \boldsymbol{g}^s) \times \mathbb{G}(:,\alpha,i)$$

Algorithm 12 Function KgP1_OptV3_gudv : computation of $\mathbb{K}_{udv}(g,i)$

Function $\mathbb{K}_g \leftarrow$ KgP1_OptV3_gudv$(\mathcal{T}_h, g, \mathbb{G}, i)$
  $\mathbb{K}_g \leftarrow \mathbb{O}_{\mathrm{n_{me}},\mathrm{n_{dfe}},\mathrm{n_{dfe}}}$ ▷ $\mathbb{K}$: zeros $\mathrm{n_{me}}$-by-$(d+1)$-by-$(d+1)$ array
  $\boldsymbol{g}_h \leftarrow g(\mathcal{T}_h.q)$, $\mathrm{g_{me}} \leftarrow \boldsymbol{g}_h(\mathcal{T}_h.\mathrm{me})$
  $\boldsymbol{g}^s \leftarrow$ sum$(\mathrm{g_{me}}, 1)$
  for $\alpha \leftarrow 1$ **to** $d+1$ **do**
    for $\beta \leftarrow 1$ **to** $d+1$ **do**
      $\mathbb{K}_g(:,\alpha,\beta) \leftarrow \frac{d!}{(d+2)!} \times \mathcal{T}_h.\text{vols} \times (\mathrm{g_{me}}(\beta,:) + \boldsymbol{g}^s) \times \mathbb{G}(:,\alpha,i)$
    end for
  end for
end Function

Algorithm 13 DAssemblyP1_OptV3

Input :

$\mathcal{T}_h$: mesh structure associated to Ω_h
$\mathcal{L}$: Doperator structure
$\mathbb{G}$: 3d-array given by function GradientsVec or empty (useful for $\mathcal{H}$ operator)

Output :

$\mathbb{D}^{\mathcal{L}}$: $\mathcal{T}_h.\mathrm{n_q}$-by-$\mathcal{T}_h.\mathrm{n_q}$ sparse matrix.

1: **Function** $\mathbb{D}^{\mathcal{L}} \leftarrow$ DAssemblyP1_OptV3($\mathcal{T}_h, \mathcal{L}, \mathbb{G}$)
2: $[\mathbb{I}_g, \mathbb{J}_g] \leftarrow$ IgJgP1_OptV3$(d, \mathcal{T}_h.\mathrm{n_{me}}, \mathcal{T}_h.\mathrm{me})$
3: **if** $\mathbb{G} = \varnothing$ **then**
4: $\mathbb{G} \leftarrow$ GradientsVec$(\mathcal{T}_h.\mathrm{q}, \mathcal{T}_h.\mathrm{me}, \mathcal{T}_h.\mathrm{vols})$
5: **end if**
6: $\mathbb{K}_g \leftarrow$ KgP1_OptV3$(\mathcal{T}_h, \mathcal{L}, \mathbb{G})$
7: $\mathbb{D}^{\mathcal{L}} \leftarrow$ Sparse$(\mathbb{I}_g(:), \mathbb{J}_g(:), \mathbb{K}_g(:), \mathcal{T}_h.\mathrm{n_q}, \mathcal{T}_h.\mathrm{n_q})$
8: **end Function**

Remark

RobinBC, RHS *and* SolvePDE *functions used the function* DAssemblyP1_base.
We just have to replace it by the new vectorized DAssemblyP1_OptV3 *function.*

Benchmarks : 2d stationary convection-diffusion problem

n_{dof} Classic	Matlab		Octave		Python		FreeFEM++	
	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)
2448	23.132	0.022	44.350	0.010	7.619	0.012	0.082	0.002
6456	62.699	0.080	119.240	0.027	20.384	0.029	0.085	0.003
12205	121.701	0.079	229.142	0.056	38.651	0.061	0.162	0.004
25253	267.526	0.256	494.841	0.135	80.334	0.136	0.339	0.009
55996	642.744	0.576	1197.113	0.348	178.354	0.358	0.754	0.023
98712	1434.475	0.868	2555.707	0.725	315.874	0.741	1.351	0.064
118216	1602.288	1.424	3397.917	0.918	379.210	0.941	1.621	0.079

n_{dof} OptV3	Matlab		Octave		Python		FreeFEM++	
	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)
25253	0.375	0.376	0.157	0.161	0.183	0.133	0.375	0.009
98712	1.135	1.086	0.469	0.714	0.820	0.720	1.346	0.063
226226	2.254	2.251	1.343	2.085	1.954	2.085	3.145	0.161
393202	3.277	3.981	2.547	4.494	3.521	4.556	5.463	0.292
605547	5.363	6.267	4.096	8.652	5.522	8.899	8.452	0.463
878642	7.952	9.738	5.970	15.367	8.019	15.852	12.201	0.691
1190916	10.583	15.057	8.211	26.755	11.000	27.621	16.809	0.961

Outline

Vector BVP

Find $\boldsymbol{u} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m) \in (\mathrm{H}^2(\Omega))^m$ such that

$$
\begin{aligned}
\mathcal{H}(\boldsymbol{u}) &= \boldsymbol{f} && \text{in } \Omega, \\
\boldsymbol{u}_\alpha &= g_\alpha^D && \text{on } \Gamma_\alpha^D, \ \forall \alpha \in [\![1, m]\!], \\
\frac{\partial \boldsymbol{u}}{\partial n_{\mathcal{H}_\alpha}} + a_\alpha^R \boldsymbol{u}_\alpha &= g_\alpha^R && \text{on } \Gamma_\alpha^R, \ \forall \alpha \in [\![1, m]\!].
\end{aligned}
$$

$$
\sum_{\beta=1}^{m} \mathcal{H}_{\alpha,\beta}(\boldsymbol{u}_\beta) = \boldsymbol{f}_\alpha, \quad \forall \alpha \in [\![1, m]\!],
$$

$$
\mathcal{H}_{\alpha,\beta} \overset{\text{def}}{=} \mathcal{L}_{\mathbb{A}^{\alpha,\beta}, \boldsymbol{b}^{\alpha,\beta}, \boldsymbol{c}^{\alpha,\beta}, a_0^{\alpha,\beta}}
$$

$$
\frac{\partial \boldsymbol{u}}{\partial n_{\mathcal{H}_\alpha}} \overset{\text{def}}{=} \sum_{\beta=1}^{m} \frac{\partial \boldsymbol{u}_\beta}{\partial n_{\mathcal{H}_{\alpha,\beta}}} = \sum_{\beta=1}^{m} \left(\left\langle \mathbb{A}^{\alpha,\beta} \nabla \boldsymbol{u}_\beta, \boldsymbol{n} \right\rangle - \left\langle \boldsymbol{b}^{\alpha,\beta} \boldsymbol{u}_\beta, \boldsymbol{n} \right\rangle \right).
$$

2D elasticity problem

Find $\boldsymbol{u} = (\mathrm{H}^2(\Omega))^2$ such that

$$
\begin{aligned}
-\operatorname{div}(\boldsymbol{\sigma}(\boldsymbol{u})) &= \boldsymbol{f} \ \text{ in } \Omega \subset \mathbb{R}^2, && (3.1)\\
\boldsymbol{\sigma}(\boldsymbol{u}).\boldsymbol{n} &= \boldsymbol{0} \ \text{ on } \Gamma^N, && (3.2)\\
\boldsymbol{u} &= \boldsymbol{0} \ \text{ on } \Gamma^D. && (3.3)
\end{aligned}
$$

2D elasticity problem

Find $\boldsymbol{u} = (\mathrm{H}^2(\Omega))^2$ such that

$$
\begin{aligned}
-\operatorname{div}(\boldsymbol{\sigma}(\boldsymbol{u})) &= \boldsymbol{f} \ \text{ in } \Omega \subset \mathbb{R}^2, && (3.1)\\
\boldsymbol{\sigma}(\boldsymbol{u}).\boldsymbol{n} &= \boldsymbol{0} \ \text{ on } \Gamma^N, && (3.2)\\
\boldsymbol{u} &= \boldsymbol{0} \ \text{ on } \Gamma^D. && (3.3)
\end{aligned}
$$

Lemma

Let $\mathcal{H}$ be the 2-by-2 matrix of the second order linear differential operators where $\mathcal{H}_{\alpha,\beta} = \mathcal{L}_{\mathbb{A}^{\alpha,\beta},\boldsymbol{0},\boldsymbol{0},0}$, $\forall (\alpha,\beta) \in [\![1,2]\!]^2$, with

$$
\mathbb{A}^{1,1} = \begin{pmatrix} \gamma & 0 \\ 0 & \mu \end{pmatrix}, \ \mathbb{A}^{1,2} = \begin{pmatrix} 0 & \lambda \\ \mu & 0 \end{pmatrix}, \ \mathbb{A}^{2,1} = \begin{pmatrix} 0 & \mu \\ \lambda & 0 \end{pmatrix}, \ \mathbb{A}^{2,2} = \begin{pmatrix} \mu & 0 \\ 0 & \gamma \end{pmatrix}
$$

where $\gamma = \lambda + 2\mu$. Then

$$
\mathcal{H}(\boldsymbol{u}) = -\operatorname{div} \boldsymbol{\sigma}(\boldsymbol{u}) \qquad (3.4)
$$

and, $\forall \alpha \in [\![1,2]\!]$,

$$
\frac{\partial \boldsymbol{u}}{\partial n_{\mathcal{H}_\alpha}} = (\boldsymbol{\sigma}(\boldsymbol{u}).\boldsymbol{n})_\alpha. \qquad (3.5)
$$

2D elasticity problem

Find $\boldsymbol{u} = (\mathrm{H}^2(\Omega))^2$ such that

$$
\begin{aligned}
-\operatorname{div}(\boldsymbol{\sigma}(\boldsymbol{u})) &= \boldsymbol{f} \text{ in } \Omega \subset \mathbb{R}^2, && (3.1)\\
\boldsymbol{\sigma}(\boldsymbol{u}).\boldsymbol{n} &= \boldsymbol{0} \text{ on } \Gamma^N, && (3.2)\\
\boldsymbol{u} &= \boldsymbol{0} \text{ on } \Gamma^D. && (3.3)
\end{aligned}
$$

2D elasticity problem with 2-by-2 $\mathcal{H}$-operator

Find $\boldsymbol{u} = (\boldsymbol{u}_1, \boldsymbol{u}_2) \in (\mathrm{H}^2(\Omega))^2$ such that

$$
\begin{aligned}
\mathcal{H}(\boldsymbol{u}) &= \boldsymbol{f}, && \text{in } \Omega,\\
\frac{\partial \boldsymbol{u}}{\partial n_{\mathcal{H}_\alpha}} &= 0, && \text{on } \Gamma^N_\alpha = \Gamma^N,\ \forall \alpha \in [\![1,2]\!]\\
\boldsymbol{u}_\alpha &= 0, && \text{on } \Gamma^D_\alpha = \Gamma^D,\ \forall \alpha \in [\![1,2]\!].
\end{aligned}
$$

Variational formulation for the *vector* BVP

$$\sum_{\beta=1}^{m} \mathcal{H}_{\alpha,\beta}(\boldsymbol{u}_\beta) = \boldsymbol{f}_\alpha, \quad \forall \alpha \in [\![1, m]\!].$$

Let $\boldsymbol{v} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) \in \left(\mathrm{H}^1(\Omega)\right)^m$. So we have, $\forall \alpha \in [\![1, m]\!]$

$$\sum_{\beta=1}^{m} \int_\Omega \mathcal{H}_{\alpha,\beta}(\boldsymbol{u}_\beta)\boldsymbol{v}_\alpha d\mathrm{q} = \int_\Omega \boldsymbol{f}_\alpha \boldsymbol{v}_\alpha d\mathrm{q}$$

As $\mathcal{H}_{\alpha,\beta}$ is a $\mathcal{L}$-operator, we have

$$\int_\Omega \mathcal{H}_{\alpha,\beta}(\boldsymbol{u}_\beta)\boldsymbol{v}_\alpha d\mathrm{q} = \int_\Omega \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_\beta, \boldsymbol{v}_\alpha) d\mathrm{q} - \int_\Gamma \frac{\partial \boldsymbol{u}_\beta}{\partial n_{\mathcal{H}_{\alpha,\beta}}} \boldsymbol{v}_\alpha d\sigma.$$

and then $\forall \alpha \in [\![1, m]\!]$

$$\sum_{\beta=1}^{m} \int_\Omega \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_\beta, \boldsymbol{v}_\alpha) d\mathrm{q} - \sum_{\beta=1}^{m} \int_\Gamma \frac{\partial \boldsymbol{u}_\beta}{\partial n_{\mathcal{H}_{\alpha,\beta}}} \boldsymbol{v}_\alpha d\sigma = \int_\Omega \boldsymbol{f}_\alpha \boldsymbol{v}_\alpha d\mathrm{q}.$$

Variational formulation for the *vector* BVP

$$\sum_{\beta=1}^{m} \mathcal{H}_{\alpha,\beta}(\boldsymbol{u}_\beta) = \boldsymbol{f}_\alpha, \quad \forall \alpha \in [\![1, m]\!].$$

Let $\boldsymbol{v} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m) \in \left(\mathrm{H}^1(\Omega)\right)^m$. So we have, $\forall \alpha \in [\![1, m]\!]$

$$\sum_{\beta=1}^{m} \int_\Omega \mathcal{H}_{\alpha,\beta}(\boldsymbol{u}_\beta)\boldsymbol{v}_\alpha d\mathrm{q} = \int_\Omega \boldsymbol{f}_\alpha \boldsymbol{v}_\alpha d\mathrm{q}$$

As $\mathcal{H}_{\alpha,\beta}$ is a $\mathcal{L}$-operator, we have

$$\int_\Omega \mathcal{H}_{\alpha,\beta}(\boldsymbol{u}_\beta)\boldsymbol{v}_\alpha d\mathrm{q} = \int_\Omega \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_\beta, \boldsymbol{v}_\alpha) d\mathrm{q} - \int_\Gamma \frac{\partial \boldsymbol{u}_\beta}{\partial n_{\mathcal{H}_{\alpha,\beta}}} \boldsymbol{v}_\alpha d\sigma.$$

and then $\forall \alpha \in [\![1, m]\!]$

$$\sum_{\beta=1}^{m} \int_\Omega \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_\beta, \boldsymbol{v}_\alpha) d\mathrm{q} - \int_\Gamma \frac{\partial \boldsymbol{u}}{\partial n_{\mathcal{H}_\alpha}} \boldsymbol{v}_\alpha d\sigma = \int_\Omega \boldsymbol{f}_\alpha \boldsymbol{v}_\alpha d\mathrm{q}.$$

Variational formulation for the *vector* BVP

$\forall \alpha \in [\![1, m]\!]$

$$\sum_{\beta=1}^{m} \int_{\Omega} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_\beta, \boldsymbol{v}_\alpha) d\mathrm{q} - \int_{\Gamma} \frac{\partial \boldsymbol{u}}{\partial n_{\mathcal{H}_\alpha}} \boldsymbol{v}_\alpha d\sigma = \int_{\Omega} \boldsymbol{f}_\alpha \boldsymbol{v}_\alpha d\mathrm{q}.$$

As $\boldsymbol{u}_\alpha \in H^1_{\boldsymbol{g}_\alpha^{\boldsymbol{D}}, \Gamma_\alpha^{\boldsymbol{D}}}$, we now take $\boldsymbol{v}_\alpha \in H^1_{0, \Gamma_\alpha^{\boldsymbol{D}}}$ and so we obtain

$$\sum_{\beta=1}^{m} \int_{\Omega} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_\beta, \boldsymbol{v}_\alpha) d\mathrm{q} - \int_{\Gamma_\alpha^{\boldsymbol{R}}} \frac{\partial \boldsymbol{u}}{\partial n_{\mathcal{H}_\alpha}} \boldsymbol{v}_\alpha d\sigma = \int_{\Omega} \boldsymbol{f}_\alpha \boldsymbol{v}_\alpha d\mathrm{q}.$$

From *Robin* boundary conditions $\frac{\partial \boldsymbol{u}}{\partial n_{\mathcal{H}_\alpha}} + a_\alpha^R \boldsymbol{u}_\alpha = g_\alpha^R$ on Γ_α^R, we have $\forall \alpha \in [\![1, m]\!]$

$$\sum_{\beta=1}^{m} \int_{\Omega} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_\beta, \boldsymbol{v}_\alpha) d\mathrm{q} + \int_{\Gamma_\alpha^{\boldsymbol{R}}} a_\alpha^R \boldsymbol{u}_\alpha \boldsymbol{v}_\alpha d\sigma = \int_{\Gamma_\alpha^{\boldsymbol{R}}} g_\alpha^R \boldsymbol{v}_\alpha d\sigma + \int_{\Omega} \boldsymbol{f}_\alpha \boldsymbol{v}_\alpha d\mathrm{q}.$$

Summing these equations on α gives ...

Variational formulation for the *vector* BVP

We set

$$\begin{aligned} \boldsymbol{A}_{\mathcal{H}}(\boldsymbol{u},\boldsymbol{v}) &= \int_{\Omega} \boldsymbol{\mathcal{D}}_{\mathcal{H}}(\boldsymbol{u},\boldsymbol{v}) d\mathrm{q} + \sum_{\alpha=1}^{m} \int_{\Gamma_{\alpha}^{\boldsymbol{R}}} a_{\alpha}^{R} \boldsymbol{u}_{\alpha} \boldsymbol{v}_{\alpha} d\sigma \\ \boldsymbol{\mathcal{F}}(\boldsymbol{v}) &= \int_{\Omega} \langle \boldsymbol{f}, \boldsymbol{v} \rangle d\mathrm{q} + \sum_{\alpha=1}^{m} \int_{\Gamma_{\alpha}^{\boldsymbol{R}}} g_{\alpha}^{R} \boldsymbol{v}_{\alpha} d\sigma \end{aligned}$$

where

$$\boldsymbol{\mathcal{D}}_{\mathcal{H}}(\boldsymbol{u},\boldsymbol{v}) = \sum_{\alpha=1}^{m} \sum_{\beta=1}^{m} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_{\beta}, \boldsymbol{v}_{\alpha})$$

Variational formulation for the *vector* BVP

Find $\boldsymbol{u} \in H^1_{g_1^{\boldsymbol{D}},\Gamma_1^{\boldsymbol{D}}} \times \ldots \times H^1_{g_m^{\boldsymbol{D}},\Gamma_m^{\boldsymbol{D}}}$ such that

$$\boldsymbol{A}_{\mathcal{H}}(\boldsymbol{u},\boldsymbol{v}) = \boldsymbol{\mathcal{F}}(\boldsymbol{v}), \ \forall \boldsymbol{v} \in H^1_{0,\Gamma_1^{\boldsymbol{D}}} \times \ldots \times H^1_{0,\Gamma_m^{\boldsymbol{D}}}$$

Or ...

Variational formulation for the *vector* BVP

We set

$$\begin{aligned} \boldsymbol{A}_{\mathcal{H}}(\boldsymbol{u},\boldsymbol{v}) &= \int_{\Omega} \boldsymbol{\mathcal{D}}_{\mathcal{H}}(\boldsymbol{u},\boldsymbol{v}) d\mathrm{q} + \sum_{\alpha=1}^{m} \int_{\Gamma_{\alpha}^{\boldsymbol{R}}} a_{\alpha}^{R} \boldsymbol{u}_{\alpha} \boldsymbol{v}_{\alpha} d\sigma \\ \boldsymbol{\mathcal{F}}(\boldsymbol{v}) &= \int_{\Omega} \langle \boldsymbol{f}, \boldsymbol{v} \rangle \, d\mathrm{q} + \sum_{\alpha=1}^{m} \int_{\Gamma_{\alpha}^{\boldsymbol{R}}} g_{\alpha}^{R} \boldsymbol{v}_{\alpha} d\sigma \end{aligned}$$

where

$$\boldsymbol{\mathcal{D}}_{\mathcal{H}}(\boldsymbol{u},\boldsymbol{v}) = \sum_{\alpha=1}^{m} \sum_{\beta=1}^{m} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_{\beta}, \boldsymbol{v}_{\alpha})$$

Variational formulation for the *vector* BVP with extension functions

Find $\boldsymbol{w} \in H^1_{0,\Gamma_{\boldsymbol{1}}^{\boldsymbol{D}}} \times \ldots \times H^1_{0,\Gamma_{\boldsymbol{m}}^{\boldsymbol{D}}}$ such that

$$\boldsymbol{A}_{\mathcal{H}}(\boldsymbol{w},\boldsymbol{v}) = \boldsymbol{\mathcal{F}}(\boldsymbol{v}) - \boldsymbol{A}_{\mathcal{H}}(\boldsymbol{R}^{D},\boldsymbol{v}), \ \forall \boldsymbol{v} \in H^1_{0,\Gamma_{\boldsymbol{1}}^{\boldsymbol{D}}} \times \ldots \times H^1_{0,\Gamma_{\boldsymbol{m}}^{\boldsymbol{D}}}$$

where $\boldsymbol{R}^{D} = (R_1^D, \ldots, R_m^D) \in \left(\mathrm{H}^1(\Omega)\right)^m$ with $\gamma_{\Gamma_{\alpha}^{\boldsymbol{D}}}(R_{\alpha}^{D}) = g_{\alpha}^{D}$ and $\boldsymbol{w} = \boldsymbol{u} - \boldsymbol{R}^{D}$.

Discrete variational formulation for the *vector* BVP

We set

$$\begin{aligned} \boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{u}_h, \boldsymbol{v}_h) &= \int_{\Omega_{\boldsymbol{h}}} \mathcal{D}_{\mathcal{H}}(\boldsymbol{u}_h, \boldsymbol{v}_h) d\mathrm{q} + \sum_{\alpha=1}^{m} \int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} a^R_\alpha \boldsymbol{u}_{h,\alpha} \boldsymbol{v}_{h,\alpha} d\sigma \\ \boldsymbol{\mathcal{F}}^h(\boldsymbol{v}_h) &= \int_{\Omega_{\boldsymbol{h}}} \langle \boldsymbol{f}, \boldsymbol{v}_h \rangle \, d\mathrm{q} + \sum_{\alpha=1}^{m} \int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} g^R_\alpha \boldsymbol{v}_{h,\alpha} d\sigma \end{aligned}$$

Discrete variational formulation with extension functions

Find $\boldsymbol{w}_h \in \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},1}} \times \ldots \times \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},m}}$ such that

$$\boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{w}_h, \boldsymbol{v}_h) = \boldsymbol{\mathcal{F}}^h(\boldsymbol{v}_h) - \boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{R}^D_h, \boldsymbol{v}_h), \ \forall \boldsymbol{v}_h \in \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},1}} \times \ldots \times \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},m}}$$

Obviously, we have $\boldsymbol{u}_h = \boldsymbol{w}_h + \boldsymbol{R}^D_h$.

Matrix representation : some notations

Let $\mathrm{n_{dof}} = m \times \mathrm{n_q}$ and, $\forall \alpha \in [\![1, m]\!]$,

$$\mathcal{I}_{D,\alpha} = \left\{ i \in [\![1, \mathrm{n_q}]\!] \mid \mathrm{q}^i \in \overline{\Gamma^D_{h,\alpha}} \right\} \qquad \mathcal{I}^c_{D,\alpha} = [\![1, \mathrm{n_q}]\!] \backslash \mathcal{I}_{D,\alpha}$$

Let

$$\boldsymbol{u_h} = (\boldsymbol{u}_{h,1}, \ldots, \boldsymbol{u}_{h,m}) \in \mathrm{H}^{1,h}_{g^D_{h,1},\Gamma^D_{h,1}} \times \ldots \times \mathrm{H}^{1,h}_{g^D_{h,m},\Gamma^D_{h,m}}$$

and

$$\boldsymbol{U} = (\boldsymbol{u}^1, \ldots, \boldsymbol{u}^m)^t \in \mathbb{R}^{\mathrm{n_{dof}}} \quad \text{with } \boldsymbol{u}^\mu \in \mathbb{R}^{\mathrm{n_q}},\ \boldsymbol{u}^\mu_i = \boldsymbol{u}_{h,\mu}(\mathrm{q}^i).$$

Then

$$\boldsymbol{u_h} = \sum_{\mu=1}^{m} \boldsymbol{u}_{h,\mu} \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \left(\sum_{i \in \mathcal{I}^c_{D,\mu}} \boldsymbol{u}^\mu_i \varphi_i + \sum_{i \in \mathcal{I}_{D,\mu}} g^D_{h,\mu}(\mathrm{q}^i) \varphi_i \right) \boldsymbol{e}_\mu$$

$$= \boldsymbol{w_h} + \boldsymbol{R^D_h}$$

Matrix representation

Discrete variational formulation with extension functions

Find $\boldsymbol{w}_h \in \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\mathbf{1}}} \times \ldots \times \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\boldsymbol{m}}}$ such that $\forall \boldsymbol{v}_h \in \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\mathbf{1}}} \times \ldots \times \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\boldsymbol{m}}}$

$$\boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{w}_h, \boldsymbol{v}_h) = \boldsymbol{\mathcal{F}}^h(\boldsymbol{v}_h) - \boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{R}^D_h, \boldsymbol{v}_h)$$

$$\mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\mathbf{1}}} \times \ldots \times \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\boldsymbol{m}}} = \mathrm{Span}\left\{\varphi_i \boldsymbol{e}_\gamma \mid \gamma \in [\![1, m]\!],\ i \in \mathcal{I}^c_{D,\gamma}\right\}$$

$$\boldsymbol{w}_h = \sum_{\mu=1}^{m} \boldsymbol{w}_{h,\mu} \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \left(\sum_{j \in \mathcal{I}^c_{\boldsymbol{D},\mu}} \boldsymbol{w}^\mu_j \varphi_j \right) \boldsymbol{e}_\mu.$$

$$\boldsymbol{R}^D_h = \sum_{\mu=1}^{m} \boldsymbol{R}^D_{h,\mu} \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{\boldsymbol{D},\mu}} \boldsymbol{R}^\mu_j \varphi_j \boldsymbol{e}_\mu, \quad \boldsymbol{R}^\mu_j = g^D_{h,\mu}(\mathrm{q}^j)$$

Matrix representation

Discrete variational formulation with extension functions

Find $\boldsymbol{w}_h \in \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\mathbf{1}}} \times \ldots \times \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\boldsymbol{m}}}$ such that $\forall \gamma \in [\![1, m]\!],\ \forall i \in \mathcal{I}^c_{D,\gamma}$

$$\boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{w}_h, \varphi_i \boldsymbol{e}_\gamma) = \boldsymbol{\mathcal{F}}^h(\varphi_i \boldsymbol{e}_\gamma) - \boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{R}^D_h, \varphi_i \boldsymbol{e}_\gamma)$$

$$\mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\mathbf{1}}} \times \ldots \times \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\boldsymbol{m}}} = \mathrm{Span}\left\{\varphi_i \boldsymbol{e}_\gamma \mid \gamma \in [\![1, m]\!],\ i \in \mathcal{I}^c_{D,\gamma}\right\}$$

$$\boldsymbol{w}_h = \sum_{\mu=1}^{m} \boldsymbol{w}_{h,\mu} \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \left(\sum_{j \in \mathcal{I}^c_{\boldsymbol{D},\mu}} \boldsymbol{w}^\mu_j \varphi_j \right) \boldsymbol{e}_\mu.$$

$$\boldsymbol{R}^D_h = \sum_{\mu=1}^{m} \boldsymbol{R}^D_{h,\mu} \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{\boldsymbol{D},\mu}} \boldsymbol{R}^\mu_j \varphi_j \boldsymbol{e}_\mu, \quad \boldsymbol{R}^\mu_j = g^D_{h,\mu}(\mathrm{q}^j)$$

Matrix representation

Discrete variational formulation with extension functions

Find $\boldsymbol{w}_h \in \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\mathbf{1}}} \times \ldots \times \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\boldsymbol{m}}}$ such that $\forall \gamma \in [\![1,m]\!],\ \forall i \in \mathcal{I}^c_{D,\gamma}$

$$\boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{w}_h, \varphi_i \boldsymbol{e}_\gamma) = \boldsymbol{\mathcal{F}}^h(\varphi_i \boldsymbol{e}_\gamma) - \boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{R}^D_h, \varphi_i \boldsymbol{e}_\gamma)$$

$$\mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\mathbf{1}}} \times \ldots \times \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\boldsymbol{m}}} = \mathrm{Span}\left\{\varphi_i \boldsymbol{e}_\gamma \mid \gamma \in [\![1,m]\!],\ i \in \mathcal{I}^c_{D,\gamma}\right\}$$

$$\boldsymbol{w}_h = \sum_{\mu=1}^{m} \boldsymbol{w}_{h,\mu} \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \left(\sum_{j \in \mathcal{I}^c_{\boldsymbol{D},\mu}} \boldsymbol{w}^\mu_j \varphi_j \right) \boldsymbol{e}_\mu.$$

$$\boldsymbol{R}^D_h = \sum_{\mu=1}^{m} \boldsymbol{R}^D_{h,\mu} \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{\boldsymbol{D},\mu}} \boldsymbol{R}^\mu_j \varphi_j \boldsymbol{e}_\mu, \quad \boldsymbol{R}^\mu_j = g^D_{h,\mu}(\mathrm{q}^j)$$

Matrix representation

Discrete variational formulation with extension functions

Find $\boldsymbol{w}_j^\mu,\ \forall \mu \in [\![1,m]\!],\ \forall j \in \mathcal{I}_{D,\gamma}^c$ such that $\forall \gamma \in [\![1,m]\!],\ \forall i \in \mathcal{I}_{D,\gamma}^c$

$$\sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{D,\mu}^c} \boldsymbol{w}_j^\mu \boldsymbol{A}_{\mathcal{H}}^h(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) = \boldsymbol{\mathcal{F}}^h(\varphi_i \boldsymbol{e}_\gamma) - \boldsymbol{A}_{\mathcal{H}}^h(\boldsymbol{R}_h^D, \varphi_i \boldsymbol{e}_\gamma)$$

$$\mathrm{H}_{0,\Gamma_{h,1}^D}^{1,h} \times \ldots \times \mathrm{H}_{0,\Gamma_{h,m}^D}^{1,h} = \mathrm{Span}\left\{\varphi_i \boldsymbol{e}_\gamma \mid \gamma \in [\![1,m]\!],\ i \in \mathcal{I}_{D,\gamma}^c\right\}$$

$$\boldsymbol{w}_h = \sum_{\mu=1}^{m} \boldsymbol{w}_{h,\mu} \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \left(\sum_{j \in \mathcal{I}_{D,\mu}^c} \boldsymbol{w}_j^\mu \varphi_j \right) \boldsymbol{e}_\mu.$$

$$\boldsymbol{R}_h^D = \sum_{\mu=1}^{m} \boldsymbol{R}_{h,\mu}^D \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{D,\mu}} \boldsymbol{R}_j^\mu \varphi_j \boldsymbol{e}_\mu, \quad \boldsymbol{R}_j^\mu = g_{h,\mu}^D(\mathrm{q}^j)$$

Matrix representation

Discrete variational formulation with extension functions

Find $\boldsymbol{w}_j^\mu,\ \forall \mu \in [\![1,m]\!],\ \forall j \in \mathcal{I}_{D,\gamma}^c$ such that $\forall \gamma \in [\![1,m]\!],\ \forall i \in \mathcal{I}_{D,\gamma}^c$

$$\sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{D,\mu}^c} \boldsymbol{w}_j^\mu \boldsymbol{A}_{\mathcal{H}}^h(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) = \boldsymbol{\mathcal{F}}^h(\varphi_i \boldsymbol{e}_\gamma) - \boldsymbol{A}_{\mathcal{H}}^h(\boldsymbol{R}_h^D, \varphi_i \boldsymbol{e}_\gamma)$$

$$\mathrm{H}_{0,\Gamma_{h,1}^D}^{1,h} \times \ldots \times \mathrm{H}_{0,\Gamma_{h,m}^D}^{1,h} = \mathrm{Span}\left\{\varphi_i \boldsymbol{e}_\gamma \mid \gamma \in [\![1,m]\!],\ i \in \mathcal{I}_{D,\gamma}^c\right\}$$

$$\boldsymbol{w}_h = \sum_{\mu=1}^{m} \boldsymbol{w}_{h,\mu} \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \left(\sum_{j \in \mathcal{I}_{D,\mu}^c} \boldsymbol{w}_j^\mu \varphi_j \right) \boldsymbol{e}_\mu.$$

$$\boldsymbol{R}_h^D = \sum_{\mu=1}^{m} \boldsymbol{R}_{h,\mu}^D \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{D,\mu}} \boldsymbol{R}_j^\mu \varphi_j \boldsymbol{e}_\mu, \quad \boldsymbol{R}_j^\mu = g_{h,\mu}^D(\mathrm{q}^j)$$

Matrix representation

Discrete variational formulation with extension functions

Find $\boldsymbol{w}_j^\mu,\ \forall \mu \in [\![1,m]\!],\ \forall j \in \mathcal{I}_{D,\gamma}^c$ such that $\forall \gamma \in [\![1,m]\!],\ \forall i \in \mathcal{I}_{D,\gamma}^c$

$$\sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{\boldsymbol{D},\mu}^c} \boldsymbol{w}_j^\mu \boldsymbol{A}_{\mathcal{H}}^h(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) = \boldsymbol{\mathcal{F}}^h(\varphi_i \boldsymbol{e}_\gamma) - \sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{\boldsymbol{D},\mu}} \boldsymbol{R}_j^\mu \boldsymbol{A}_{\mathcal{H}}^h(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma)$$

$$\mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\mathbf{1}}} \times \ldots \times \mathrm{H}^{1,h}_{0,\Gamma^{\boldsymbol{D}}_{\boldsymbol{h},\boldsymbol{m}}} = \mathrm{Span}\left\{ \varphi_i \boldsymbol{e}_\gamma \mid \gamma \in [\![1,m]\!],\ i \in \mathcal{I}_{D,\gamma}^c \right\}$$

$$\boldsymbol{w}_h = \sum_{\mu=1}^{m} \boldsymbol{w}_{h,\mu} \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \left(\sum_{j \in \mathcal{I}_{\boldsymbol{D},\mu}^c} \boldsymbol{w}_j^\mu \varphi_j \right) \boldsymbol{e}_\mu.$$

$$\boldsymbol{R}_h^D = \sum_{\mu=1}^{m} \boldsymbol{R}_{h,\mu}^D \boldsymbol{e}_\mu = \sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{\boldsymbol{D},\mu}} \boldsymbol{R}_j^\mu \varphi_j \boldsymbol{e}_\mu, \quad \boldsymbol{R}_j^\mu = g_{h,\mu}^D(\mathrm{q}^j)$$

Matrix representation : matrix construction

Let $\mathbb{A}$ be the $\mathrm{n_{dof}}$-by-$\mathrm{n_{dof}}$ block matrix defined by

$$\mathbb{A} := \left[\begin{array}{c|c|c} \mathbb{A}^{1,1} & \dots & \mathbb{A}^{1,m} \\ \hline \vdots & \ddots & \vdots \\ \hline \mathbb{A}^{m,1} & \dots & \mathbb{A}^{m,m} \end{array}\right] \tag{3.4}$$

where each block $\mathbb{A}^{\gamma,\mu}$ is the $\mathrm{n_q}$-by-$\mathrm{n_q}$ matrix given $\forall (i,j) \in [\![1, \mathrm{n_q}]\!]^2$ by

$$\mathbb{A}^{\gamma,\mu}_{i,j} = \boldsymbol{A}^h_{\mathcal{H}}(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma). \tag{3.5}$$

We set $\boldsymbol{b} = (\boldsymbol{b}^1, \dots, \boldsymbol{b}^m)^t$ as the block vector in $\mathbb{R}^{m \times \mathrm{n_q}}$ defined by

$$\boldsymbol{b}^\gamma_i = \boldsymbol{\mathcal{F}}^h(\varphi_i \boldsymbol{e}_\gamma) \tag{3.6}$$

Discrete variational formulation with extension functions

Find $\boldsymbol{w}_j^\mu,\ \forall \mu \in [\![1,m]\!],\ \forall j \in \mathcal{I}^c_{D,\gamma}\ \forall \gamma \in [\![1,m]\!],\ \forall i \in \mathcal{I}^c_{D,\gamma}$

$$\sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}^c_{D,\mu}} \boldsymbol{A}^h_{\mathcal{H}}(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) \boldsymbol{w}_j^\mu = \boldsymbol{\mathcal{F}}^h(\varphi_i \boldsymbol{e}_\gamma) - \sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{D,\mu}} \boldsymbol{A}^h_{\mathcal{H}}(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) \boldsymbol{R}_j^\mu$$

$$\mathbb{A}_{i,j}^{\gamma,\mu} = \boldsymbol{A}^h_{\mathcal{H}}(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) \qquad \boldsymbol{b}_i^\gamma = \boldsymbol{\mathcal{F}}^h(\varphi_i \boldsymbol{e}_\gamma)$$

$$\mathcal{I}_D = \bigcup_{\alpha=1}^{m} \{i + (\alpha-1)\mathrm{n_q},\ \forall i \in \mathcal{I}_{D,\alpha}\} \subset [\![1, \mathrm{n_{dof}}]\!]$$

Matrix formulation with extension functions

Find $W \in \mathbb{R}^{\mathrm{n_{dof}}}$ such that

$$\begin{aligned} \mathbb{A}(\mathcal{I}^c_D, \mathcal{I}^c_D) W(\mathcal{I}^c_D) &= \boldsymbol{b}(\mathcal{I}^c_D) - \mathbb{A}(\mathcal{I}^c_D, \mathcal{I}_D) R(\mathcal{I}_D) \\ W(\mathcal{I}_D) &= 0. \end{aligned}$$

Discrete variational formulation with extension functions

Find $\boldsymbol{w}_j^\mu,\ \forall \mu \in [\![1,m]\!],\ \forall j \in \mathcal{I}_{D,\gamma}^c\ \forall \gamma \in [\![1,m]\!],\ \forall i \in \mathcal{I}_{D,\gamma}^c$

$$\sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{\boldsymbol{D},\mu}^c} \mathbb{A}_{i,j}^{\gamma,\mu} \boldsymbol{w}_j^\mu = \boldsymbol{b}_i^\gamma - \sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{\boldsymbol{D},\mu}} \mathbb{A}_{i,j}^{\gamma,\mu} \boldsymbol{R}_j^\mu$$

$$\mathbb{A}_{i,j}^{\gamma,\mu} = \boldsymbol{A}_{\mathcal{H}}^h(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) \qquad \boldsymbol{b}_i^\gamma = \boldsymbol{\mathcal{F}}^h(\varphi_i \boldsymbol{e}_\gamma)$$

$$\mathcal{I}_D = \bigcup_{\alpha=1}^{m} \{i + (\alpha-1)\mathrm{n_q},\ \forall i \in \mathcal{I}_{\boldsymbol{D},\alpha}\} \subset [\![1, \mathrm{n_{dof}}]\!]$$

Matrix formulation with extension functions

Find $W \in \mathbb{R}^{\mathrm{n_{dof}}}$ such that

$$\begin{aligned} \mathbb{A}(\mathcal{I}_D^c, \mathcal{I}_D^c) W(\mathcal{I}_D^c) &= \boldsymbol{b}(\mathcal{I}_D^c) - \mathbb{A}(\mathcal{I}_D^c, \mathcal{I}_D) R(\mathcal{I}_D) \\ W(\mathcal{I}_D) &= 0. \end{aligned}$$

Discrete variational formulation with extension functions

Find $\boldsymbol{w}_j^\mu,\ \forall \mu \in [\![1,m]\!],\ \forall j \in \mathcal{I}_{D,\gamma}^c\ \forall \gamma \in [\![1,m]\!],\ \forall i \in \mathcal{I}_{D,\gamma}^c$

$$\sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{\boldsymbol{D},\mu}^c} \mathbb{A}_{i,j}^{\gamma,\mu} \boldsymbol{w}_j^\mu = \boldsymbol{b}_i^\gamma - \sum_{\mu=1}^{m} \sum_{j \in \mathcal{I}_{\boldsymbol{D},\mu}} \mathbb{A}_{i,j}^{\gamma,\mu} \boldsymbol{R}_j^\mu$$

$$\mathbb{A}_{i,j}^{\gamma,\mu} = \boldsymbol{A}_{\mathcal{H}}^h(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) \qquad \boldsymbol{b}_i^\gamma = \boldsymbol{\mathcal{F}}^h(\varphi_i \boldsymbol{e}_\gamma)$$

$$\mathcal{I}_D = \bigcup_{\alpha=1}^{m} \{i + (\alpha-1)\mathrm{n_q},\ \forall i \in \mathcal{I}_{\boldsymbol{D},\alpha}\} \subset [\![1, \mathrm{n_{dof}}]\!]$$

Matrix formulation with extension functions

Find $\boldsymbol{W} \in \mathbb{R}^{\mathrm{n_{dof}}}$ such that

$$\begin{aligned} \mathbb{A}(\mathcal{I}_D^c, \mathcal{I}_D^c) \boldsymbol{W}(\mathcal{I}_D^c) &= \boldsymbol{b}(\mathcal{I}_D^c) - \mathbb{A}(\mathcal{I}_D^c, \mathcal{I}_D) \boldsymbol{R}(\mathcal{I}_D) \\ \boldsymbol{W}(\mathcal{I}_D) &= 0. \end{aligned}$$

Vectorized assembly of $\mathcal{H}$-operator

we recall that :

- $\boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{u}_h, \boldsymbol{v}_h) = \int_{\Omega_{\boldsymbol{h}}} \boldsymbol{\mathcal{D}}_{\mathcal{H}}(\boldsymbol{u}_h, \boldsymbol{v}_h) d\mathrm{q} + \sum_{\alpha=1}^{m} \int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} a^R_\alpha \boldsymbol{u}_{h,\alpha} \boldsymbol{v}_{h,\alpha} d\sigma$

Vectorized assembly of $\mathcal{H}$-operator

we recall that :

- $\boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{u}_h, \boldsymbol{v}_h) = \sum_{\alpha=1}^{m} \sum_{\beta=1}^{m} \int_{\Omega_{\boldsymbol{h}}} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_{h,\beta}, \boldsymbol{v}_{h,\alpha}) d\mathrm{q} + \sum_{\alpha=1}^{m} \int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} a^R_\alpha \boldsymbol{u}_{h,\alpha} \boldsymbol{v}_{h,\alpha} d\sigma$

Vectorized assembly of $\mathcal{H}$-operator

we recall that :

- $\boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{u}_h, \boldsymbol{v}_h) = \sum_{\alpha=1}^{m}\sum_{\beta=1}^{m}\int_{\Omega_{\boldsymbol{h}}} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_{h,\beta}, \boldsymbol{v}_{h,\alpha})d\mathrm{q} + \sum_{\alpha=1}^{m}\int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} a^R_\alpha \boldsymbol{u}_{h,\alpha}\boldsymbol{v}_{h,\alpha}d\sigma$

  Then $\forall(\mu,\gamma) \in [\![1, m]\!]^2$, $\forall(i,j) \in [\![1, \mathrm{n_q}]\!]^2$,

$$\boldsymbol{A}^h_{\mathcal{H}}(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) = \sum_{\alpha=1}^{m}\sum_{\beta=1}^{m}\int_{\Omega_{\boldsymbol{h}}} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\varphi_j\delta_{\mu,\beta}, \varphi_i\delta_{\gamma,\alpha})d\mathrm{q} + \sum_{\alpha=1}^{m}\int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} a^R_\alpha \varphi_j\delta_{\mu,\alpha}\varphi_i\delta_{\gamma,\alpha}d\sigma$$

Vectorized assembly of $\mathcal{H}$-operator

we recall that :

- $\boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{u}_h, \boldsymbol{v}_h) = \sum_{\alpha=1}^{m}\sum_{\beta=1}^{m}\int_{\Omega_{\boldsymbol{h}}} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_{h,\beta}, \boldsymbol{v}_{h,\alpha})d\mathrm{q} + \sum_{\alpha=1}^{m}\int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} a^R_\alpha \boldsymbol{u}_{h,\alpha}\boldsymbol{v}_{h,\alpha}d\sigma$

  Then $\forall(\mu,\gamma) \in [\![1,m]\!]^2$, $\forall(i,j) \in [\![1,\mathrm{n_q}]\!]^2$,

$$\mathbb{A}^{\gamma,\mu}_{i,j} \overset{\text{def}}{=} \boldsymbol{A}^h_{\mathcal{H}}(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) = \int_{\Omega_{\boldsymbol{h}}} \mathcal{D}_{\mathcal{H}_{\gamma,\mu}}(\varphi_j, \varphi_i)d\mathrm{q} + \delta_{\mu,\gamma}\int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\gamma}} a^R_\gamma \varphi_j\varphi_i d\sigma$$
$$= \mathbb{H}^{\gamma,\mu}_{i,j} + \mathbb{B}^{\gamma,\mu}_{i,j}$$

Vectorized assembly of $\mathcal{H}$-operator

we recall that :

- $\boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{u}_h, \boldsymbol{v}_h) = \sum_{\alpha=1}^{m}\sum_{\beta=1}^{m}\int_{\Omega_{\boldsymbol{h}}} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_{h,\beta}, \boldsymbol{v}_{h,\alpha})d\mathrm{q} + \sum_{\alpha=1}^{m}\int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} a^R_\alpha \boldsymbol{u}_{h,\alpha}\boldsymbol{v}_{h,\alpha}d\sigma$

  Then $\forall(\mu,\gamma) \in [\![1, m]\!]^2$, $\forall(i,j) \in [\![1, \mathrm{n_q}]\!]^2$,

$$\mathbb{A}^{\gamma,\mu}_{i,j} \overset{\text{def}}{=} \boldsymbol{A}^h_{\mathcal{H}}(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) = \int_{\Omega_{\boldsymbol{h}}} \mathcal{D}_{\mathcal{H}_{\gamma,\mu}}(\varphi_j, \varphi_i)d\mathrm{q} + \delta_{\mu,\gamma}\int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\gamma}} a^R_\gamma \varphi_j\varphi_i d\sigma$$
$$= \mathbb{H}^{\gamma,\mu}_{i,j} + \mathbb{B}^{\gamma,\mu}_{i,j}$$

- $\boldsymbol{\mathcal{F}}^h(\boldsymbol{v}_h) = \int_{\Omega_{\boldsymbol{h}}} \langle \boldsymbol{f}, \boldsymbol{v}_h\rangle d\mathrm{q} + \sum_{\alpha=1}^{m}\int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} g^R_\alpha \boldsymbol{v}_{h,\alpha}d\sigma$

  Then $\forall\gamma \in [\![1, m]\!]$, $\forall i \in [\![1, \mathrm{n_q}]\!]$

$$\boldsymbol{b}^\gamma_i \overset{\text{def}}{=} \boldsymbol{\mathcal{F}}^h(\varphi_i \boldsymbol{e}_\gamma) = \sum_{\alpha=1}^{m}\int_{\Omega_{\boldsymbol{h}}} \boldsymbol{f}_\alpha \varphi_i \delta_{\gamma,\alpha} d\mathrm{q} + \sum_{\alpha=1}^{m}\int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} g^R_\alpha \varphi_i \delta_{\gamma,\alpha} d\sigma$$

Vectorized assembly of $\mathcal{H}$-operator

we recall that :

- $\boldsymbol{A}^h_{\mathcal{H}}(\boldsymbol{u}_h, \boldsymbol{v}_h) = \sum_{\alpha=1}^{m}\sum_{\beta=1}^{m}\int_{\Omega_{\boldsymbol{h}}} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\boldsymbol{u}_{h,\beta}, \boldsymbol{v}_{h,\alpha})d\mathrm{q} + \sum_{\alpha=1}^{m}\int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} a^R_\alpha \boldsymbol{u}_{h,\alpha}\boldsymbol{v}_{h,\alpha}d\sigma$

  Then $\forall(\mu,\gamma) \in [\![1, m]\!]^2$, $\forall(i,j) \in [\![1, \mathrm{n_q}]\!]^2$,

$$\mathbb{A}^{\gamma,\mu}_{i,j} \overset{\text{def}}{=} \boldsymbol{A}^h_{\mathcal{H}}(\varphi_j \boldsymbol{e}_\mu, \varphi_i \boldsymbol{e}_\gamma) = \int_{\Omega_{\boldsymbol{h}}} \mathcal{D}_{\mathcal{H}_{\gamma,\mu}}(\varphi_j, \varphi_i)d\mathrm{q} + \delta_{\mu,\gamma}\int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\gamma}} a^R_\gamma \varphi_j\varphi_i d\sigma$$
$$= \mathbb{H}^{\gamma,\mu}_{i,j} + \mathbb{B}^{\gamma,\mu}_{i,j}$$

- $\boldsymbol{\mathcal{F}}^h(\boldsymbol{v}_h) = \int_{\Omega_{\boldsymbol{h}}} \langle \boldsymbol{f}, \boldsymbol{v}_h\rangle d\mathrm{q} + \sum_{\alpha=1}^{m}\int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\alpha}} g^R_\alpha \boldsymbol{v}_{h,\alpha}d\sigma$

  Then $\forall\gamma \in [\![1, m]\!]$, $\forall i \in [\![1, \mathrm{n_q}]\!]$

$$\boldsymbol{b}^\gamma_i = \int_{\Omega_{\boldsymbol{h}}} \boldsymbol{f}_\gamma \varphi_i d\mathrm{q} + \int_{\Gamma^{\boldsymbol{R}}_{\boldsymbol{h},\gamma}} g^R_\gamma \varphi_i d\sigma$$
$$= \boldsymbol{b}^{f,\gamma}_i + \boldsymbol{b}^{R,\gamma}_i$$

Solving *vector* B.V.P.

Algorithm 14 function SolvePDE : solving a *scalar* or *vector* BVP.

Function $\boldsymbol{U} \leftarrow \text{SolvePDE}(\text{pde})$
  $\text{n}_{\text{dof}} \leftarrow \text{pde.m} \times \text{pde.}\mathcal{T}_h\text{.n}_\text{q}$
  $\mathbb{H} \leftarrow \text{HAssemblyP1\_OptV3}(\text{pde.}\mathcal{T}_h, \text{pde.Op})$
  $\boldsymbol{b}^{\boldsymbol{f}} \leftarrow \text{RHS}(\text{pde.}\mathcal{T}_h, \text{pde.}\boldsymbol{f})$
  $[\boldsymbol{b}^R, \mathbb{B}^R] \leftarrow \text{RobinBC}(\text{pde})$
  $[\mathcal{I}_D, \mathcal{I}_D^c, \boldsymbol{R}] \leftarrow \text{DirichletBC}(\text{pde})$
  $\boldsymbol{b} \leftarrow \boldsymbol{b}^{\boldsymbol{f}} + \boldsymbol{b}^R$
  $\mathbb{A} \leftarrow \mathbb{H} + \mathbb{B}^R$
  $\boldsymbol{U}(\mathcal{I}_D^c) \leftarrow \text{Solve}(\mathbb{A}(\mathcal{I}_D^c, \mathcal{I}_D^c), \boldsymbol{b}(\mathcal{I}_D^c) - \mathbb{A}(\mathcal{I}_D^c, \mathcal{I}_D)\boldsymbol{R}(\mathcal{I}_D))$
  $\boldsymbol{U}(\mathcal{I}_D) \leftarrow \boldsymbol{R}(\mathcal{I}_D)$
end Function

Functions RHS, RobinBC and DirichletBC are easy to write
Let us focus on HAssemblyP1_base or HAssemblyP1_OptV3 functions!

Vectorized assembly of $\mathcal{H}$-operator

$\mathbb{H}$ m-by-m block matrix, $\mathbb{H}_{i,j}^{\alpha,\beta} = \int_{\Omega_h} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\varphi_j, \varphi_i) d\mathrm{q}$

Algorithm 15 Function HAssemblyP1_OptV3 - Matrix $\mathbb{H}$ assembly

Function $\mathbb{H} \leftarrow$ HAssemblyP1_OptV3$(\mathcal{T}_h, \mathrm{Hop})$
  $\mathrm{n_{dof}} \leftarrow \mathrm{Hop.m} * \mathcal{T}_h.\mathrm{n_q}$
  $\mathbb{H} \leftarrow$ Sparse$(\mathrm{n_{dof}}, \mathrm{n_{dof}})$
  $\mathbb{G} \leftarrow$ GradientsVec$(\mathcal{T}_h.\mathrm{q}, \mathcal{T}_h.\mathrm{me}, \mathcal{T}_h.\mathrm{vols})$
  for $\alpha \leftarrow 1$ **to** $\mathrm{Hop.m}$ **do**
    $\mathcal{I} \leftarrow [\![1, \mathcal{T}_h.\mathrm{n_q}]\!] + (\alpha - 1)\mathcal{T}_h.\mathrm{n_q}$
    for $\beta \leftarrow 1$ **to** $\mathrm{Hop.m}$ **do** ▷ computation of block matrix (α, β)
      $\mathcal{J} \leftarrow [\![1, \mathcal{T}_h.\mathrm{n_q}]\!] + (\beta - 1)\mathcal{T}_h.\mathrm{n_q}$
      $\mathbb{H}(\mathcal{I}, \mathcal{J}) \leftarrow$ DAssemblyP1_OptV3$(\mathcal{T}_h, \mathrm{Hop.H}(\alpha, \beta), \mathbb{G})$
    end for
  end for
end Function

One can also use DAssemblyP1_base instead of DAssemblyP1_OptV3!

Vectorized assembly of $\mathcal{H}$-operator

$\mathbb{H}$ m-by-m block matrix, $\mathbb{H}_{i,j}^{\alpha,\beta} = \int_{\Omega_h} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\varphi_j, \varphi_i) d\mathrm{q}$

Algorithm 16 Function HAssemblyP1_OptV3 - Matrix $\mathbb{H}$ assembly

Function $\mathbb{H} \leftarrow$ HAssemblyP1_OptV3$(\mathcal{T}_h, \mathrm{Hop})$
  $\mathrm{n_{dof}} \leftarrow \mathrm{Hop.m} * \mathcal{T}_h.\mathrm{n_q}$
  $\mathbb{H} \leftarrow$ Sparse$(\mathrm{n_{dof}}, \mathrm{n_{dof}})$
  $\mathbb{G} \leftarrow$ GradientsVec$(\mathcal{T}_h.\mathrm{q}, \mathcal{T}_h.\mathrm{me}, \mathcal{T}_h.\mathrm{vols})$
  for $\alpha \leftarrow 1$ **to** $\mathrm{Hop.m}$ **do**
    $\mathcal{I} \leftarrow [\![1, \mathcal{T}_h.\mathrm{n_q}]\!] + (\alpha - 1)\mathcal{T}_h.\mathrm{n_q}$
    for $\beta \leftarrow 1$ **to** $\mathrm{Hop.m}$ **do** ▷ computation of block matrix (α, β)
      $\mathcal{J} \leftarrow [\![1, \mathcal{T}_h.\mathrm{n_q}]\!] + (\beta - 1)\mathcal{T}_h.\mathrm{n_q}$
      $\mathbb{H}(\mathcal{I}, \mathcal{J}) \leftarrow$ DAssemblyP1_OptV3$(\mathcal{T}_h, \mathrm{Hop.H}(\alpha, \beta), \mathbb{G})$
    end for
  end for
end Function

One can also use DAssemblyP1_base instead of DAssemblyP1_OptV3!

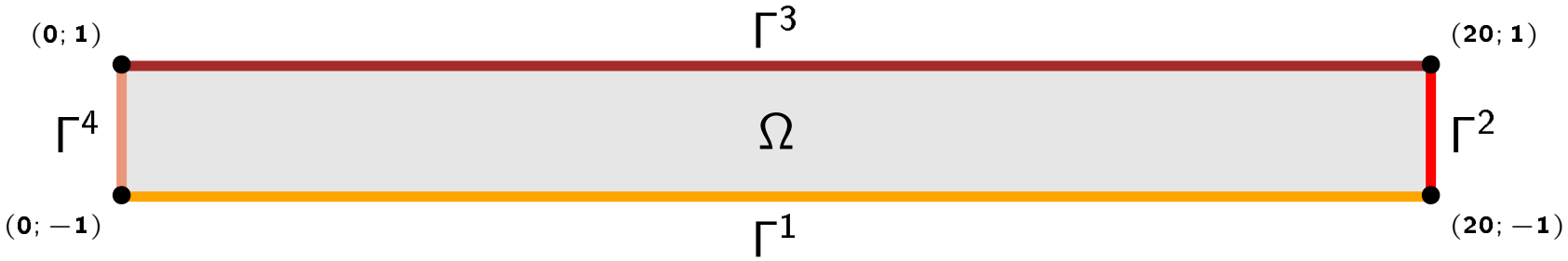
Vectorized assembly of $\mathcal{H}$-operator

$\mathbb{H}$ m-by-m block matrix, $\mathbb{H}_{i,j}^{\alpha,\beta} = \int_{\Omega_h} \mathcal{D}_{\mathcal{H}_{\alpha,\beta}}(\varphi_j, \varphi_i) d\mathrm{q}$

Algorithm 17 Function HAssemblyP1_OptV3 - Matrix $\mathbb{H}$ assembly *(version 2)*

Function $\mathbb{H} \leftarrow$ HAssemblyP1_OptV3$(\mathcal{T}_h, \mathrm{Hop})$
  $\mathrm{n_{dof}} \leftarrow \mathrm{Hop.m} * \mathcal{T}_h.\mathrm{n_q}$
  $\mathbb{H} \leftarrow$ Sparse$(\mathrm{n_{dof}}, \mathrm{n_{dof}})$
  $\mathbb{G} \leftarrow$ GradientsVec$(\mathcal{T}_h.\mathrm{q}, \mathcal{T}_h.\mathrm{me}, \mathcal{T}_h.\mathrm{vols})$
  $[\mathbb{I}_g, \mathbb{J}_g] \leftarrow$ IgJgP1_OptV3$(d, \mathrm{n_{me}}, \mathrm{me})$
  for $\alpha \leftarrow 1$ **to** Hop.m **do**
    $\mathbb{I} \leftarrow \mathbb{I}_g + (\alpha - 1)\mathcal{T}_h.\mathrm{n_q}$
    for $\beta \leftarrow 1$ **to** Hop.m **do**
      $\mathbb{J} \leftarrow \mathbb{J}_g + (\beta - 1)\mathcal{T}_h.\mathrm{n_q}$
      $\mathbb{K} \leftarrow$ KgP1_OptV3$(\mathcal{T}_h, \mathrm{Hop.H}(\alpha, \beta), \mathbb{G})$
      $\mathbb{H} \leftarrow \mathbb{H} +$ Sparse$(\mathbb{I}(:), \mathbb{J}(:), \mathbb{K}(:), \mathrm{n_{dof}}, \mathrm{n_{dof}})$
    end for
  end for
end Function

Benchmarks : 2d elasticity problem



Algorithm 18 2D elasticity

$\mathcal{T}_h \leftarrow$ getMesh(...) ▷ Load 2D mesh
$\lambda \leftarrow \frac{E\nu}{(1+\nu)(1-2\nu)}$
$\mu \leftarrow \frac{E}{2(1+\nu)}$
$\mathtt{Hop} \leftarrow$ InitHoperator$(2,2)$
$\mathtt{Hop}.H(1,1) \leftarrow$ Loperator$(2,[2\mu+\lambda,0;0,\mu],\mathbf{0},\mathbf{0},0)$
$\mathtt{Hop}.H(2,1) \leftarrow$ Loperator$(2,[0,\lambda;\mu,0],\mathbf{0},\mathbf{0},0)$
$\mathtt{Hop}.H(1,2) \leftarrow$ Loperator$(2,[0,\mu;\lambda,0],\mathbf{0},\mathbf{0},0)$
$\mathtt{Hop}.H(2,2) \leftarrow$ Loperator$(2,[\mu,0;0,2\mu+\lambda],\mathbf{0},\mathbf{0},0)$
$\mathrm{pde} \leftarrow$ initPDE$(\mathtt{Hop},\mathcal{T}_h)$
$\mathrm{pde} \leftarrow$ setBC_PDE$(\mathrm{pde},4,1:2,\mathtt{'Dirichlet'},\boldsymbol{x}\rightarrow\mathbf{0})$
$\mathrm{pde}.f \leftarrow \boldsymbol{x}\rightarrow[0,-1]$
$\boldsymbol{x} \leftarrow$ SolvePDE(pde)

Benchmarks : 2d elasticity problem

n_{dof}	Matlab		Octave		Python		FreeFEM++	
Classic	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)
8442	62.116	0.084	121.577	0.050	55.301	0.055	0.156	0.004
18662	143.297	0.180	276.281	0.130	123.622	0.137	0.269	0.012
32882	258.444	0.318	504.902	0.282	219.901	0.294	0.471	0.024
51102	414.511	0.866	812.632	0.508	343.043	0.521	0.740	0.040
73322	611.897	1.096	1214.158	0.795	491.822	0.819	1.063	0.059
99542	871.479	1.336	1750.689	1.025	669.178	1.134	1.448	0.084

n_{dof}	Matlab		Octave		Python		FreeFEM++	
Opt V3	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)
51102	0.547	0.300	0.192	0.237	0.233	0.676	0.740	0.040
202202	2.180	1.023	0.695	1.279	1.067	4.563	2.961	0.181
453302	4.080	2.195	1.760	3.414	2.473	15.071	6.790	0.447
804402	6.795	3.826	3.453	6.837	4.703	44.976	11.871	0.809
1255502	10.799	5.854	5.611	12.303	7.518	87.745	18.546	1.605

Benchmarks : 3d elasticity problem

n_{dof} classical	Matlab		Octave		Python		FreeFEM++	
	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)
624	20.352	0.018	39.974	0.006	16.447	0.008	0.067	0.023
2268	91.260	0.070	182.602	0.046	92.118	0.049	0.134	0.068
5568	251.202	0.301	500.730	0.197	337.883	0.234	0.371	0.251
14883	734.156	1.063	1468.289	1.926	1640.140	1.959	1.180	1.034
31164	1594.150	3.480	3279.344	8.192	6032.708	11.114	2.380	2.982

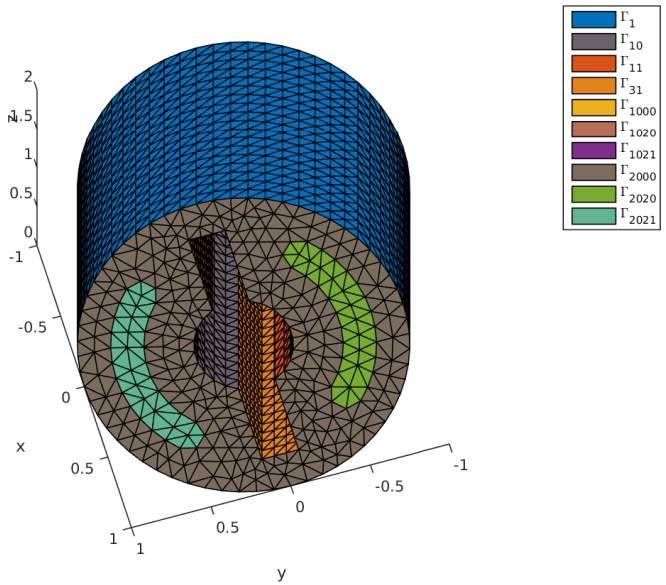
n_{dof} OptV3	Matlab		Octave		Python		FreeFEM++	
	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)	System(s)	Solve(s)
14883	0.892	1.634	0.363	2.933	0.386	1.627	1.082	1.031
107163	7.327	32.043	2.336	43.451	4.062	22.652	8.579	17.278
348843	19.199	113.125	10.422	210.357	16.062	114.045	29.504	84.889
811923	47.138	341.732	25.878	662.872	39.185	371.655	69.304	265.619

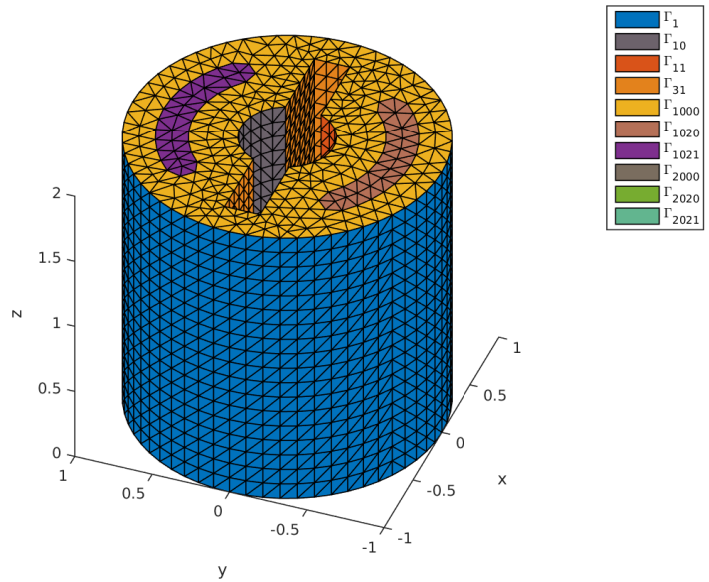
Outline

3D problem : stationary heat with potential flow

Find $u \in \mathrm{H}^2(\Omega)$ such that

$$
\begin{aligned}
-\operatorname{div}(\alpha \nabla u) + \langle \boldsymbol{V}, \nabla u \rangle + \beta u &= 0 \quad \text{in } \Omega \subset \mathbb{R}^3, && (4.1)\\
u &= 30 \quad \text{on } \Gamma_{1020} \cup \Gamma_{2020}, && (4.2)\\
u &= 10\delta_{|z-1|>0.5} \quad \text{on } \Gamma_{10}, && (4.3)\\
\frac{\partial u}{\partial n} &= 0 \quad \text{otherwise} && (4.4)
\end{aligned}
$$





3D problem : stationary heat with potential flow

Find $u \in \mathrm{H}^2(\Omega)$ such that

$$
\begin{aligned}
-\operatorname{div}(\alpha \nabla u) + \langle \boldsymbol{V}, \nabla u \rangle + \beta u &= 0 \ \text{ in } \Omega \subset \mathbb{R}^3, && (4.1)\\
u &= 30 \ \text{ on } \Gamma_{1020} \cup \Gamma_{2020}, && (4.2)\\
u &= 10\delta_{|z-1|>0.5} \ \text{ on } \Gamma_{10}, && (4.3)\\
\frac{\partial u}{\partial n} &= 0 \ \text{ otherwise} && (4.4)
\end{aligned}
$$

$$\mathcal{L}_{\alpha\mathbb{I},\mathbf{0},\boldsymbol{V},\beta}(u) = 0$$

$$\frac{\partial u}{\partial n_{\mathcal{L}}} := \langle \mathbb{A} \nabla u, \boldsymbol{n} \rangle - \langle \boldsymbol{b} u, \boldsymbol{n} \rangle = \alpha \frac{\partial u}{\partial n}.$$

Velocity potential in 3d : $\boldsymbol{V} = \nabla \phi$

Find $\phi \in \mathrm{H}^2(\Omega)$ such that

$$
\begin{aligned}
-\Delta\phi &= 0 \ \text{ in } \Omega, && (4.5)\\
\phi &= 1 \ \text{ on } \Gamma_{1021} \cup \Gamma_{2021}, && (4.6)\\
\phi &= -1 \ \text{ on } \Gamma_{1020} \cup \Gamma_{2020}, && (4.7)\\
\frac{\partial \phi}{\partial n} &= 0 \ \text{ otherwise} && (4.8)
\end{aligned}
$$

Velocity potential and velocity field in 3d

Find $\phi \in \mathrm{H}^2(\Omega)$ and $\boldsymbol{V} \in \mathrm{H}^1(\Omega)^3$ such that

$$
\begin{aligned}
-\left(\frac{\partial \boldsymbol{V}_1}{\partial x} + \frac{\partial \boldsymbol{V}_2}{\partial y} + \frac{\partial \boldsymbol{V}_3}{\partial z}\right) &= 0 \ \text{ in } \Omega, && (4.9)\\
\boldsymbol{V}_1 - \frac{\partial \phi}{\partial x} &= 0 \ \text{ in } \Omega, && (4.10)\\
\boldsymbol{V}_2 - \frac{\partial \phi}{\partial y} &= 0 \ \text{ in } \Omega, && (4.11)\\
\boldsymbol{V}_3 - \frac{\partial \phi}{\partial z} &= 0 \ \text{ in } \Omega, && (4.12)
\end{aligned}
$$

with boundary conditions (4.6) to (4.8).

$$-\left(\frac{\partial V_1}{\partial x}+\frac{\partial V_2}{\partial y}+\frac{\partial V_3}{\partial z}\right)=0$$

$$V_1-\frac{\partial \phi}{\partial x}=0$$

$$V_2-\frac{\partial \phi}{\partial y}=0$$

$$V_3-\frac{\partial \phi}{\partial z}=0$$

Let $\boldsymbol{w}=(\phi,\ \boldsymbol{V}_1,\ \boldsymbol{V}_2,\ \boldsymbol{V}_3)^t$, and $\boldsymbol{e}_1=(1,0,0)^t$, $\boldsymbol{e}_2=(0,1,0)^t$, $\boldsymbol{e}_3=(0,0,1)^t$.

$$\mathcal{H}(\boldsymbol{w})=0 \tag{4.13}$$

with

$$\mathcal{H}_{1,1}=0, \quad \mathcal{H}_{1,2}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_1},\boldsymbol{0},0}, \quad \mathcal{H}_{1,3}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_2},\boldsymbol{0},0}, \quad \mathcal{H}_{1,4}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_3},\boldsymbol{0},0},$$
$$\mathcal{H}_{2,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_1},0}, \quad \mathcal{H}_{2,2}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1}, \quad \mathcal{H}_{2,3}=0, \quad \mathcal{H}_{2,4}=0,$$
$$\mathcal{H}_{3,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_2},0}, \quad \mathcal{H}_{3,2}=0, \quad \mathcal{H}_{3,3}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1}, \quad \mathcal{H}_{3,4}=0,$$
$$\mathcal{H}_{4,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_3},0}, \quad \mathcal{H}_{4,2}=0, \quad \mathcal{H}_{4,3}=0, \quad \mathcal{H}_{4,4}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1},$$

$$-\left(\frac{\partial \boldsymbol{V}_1}{\partial x}+\frac{\partial \boldsymbol{V}_2}{\partial y}+\frac{\partial \boldsymbol{V}_3}{\partial z}\right)=0$$

$$\boldsymbol{V}_1-\frac{\partial \phi}{\partial x}=0$$

$$\boldsymbol{V}_2-\frac{\partial \phi}{\partial y}=0$$

$$\boldsymbol{V}_3-\frac{\partial \phi}{\partial z}=0$$

Let $\boldsymbol{w}=(\phi,\ \boldsymbol{V}_1,\ \boldsymbol{V}_2,\ \boldsymbol{V}_3)^t$, and $\boldsymbol{e}_1=(1,0,0)^t$, $\boldsymbol{e}_2=(0,1,0)^t$, $\boldsymbol{e}_3=(0,0,1)^t$.

$$\mathcal{H}(\boldsymbol{w})=0 \tag{4.13}$$

with

$$\mathcal{H}_{1,1}=0, \quad \mathcal{H}_{1,2}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_1},\boldsymbol{0},0}, \quad \mathcal{H}_{1,3}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_2},\boldsymbol{0},0}, \quad \mathcal{H}_{1,4}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_3},\boldsymbol{0},0},$$
$$\mathcal{H}_{2,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_1},0}, \quad \mathcal{H}_{2,2}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1}, \quad \mathcal{H}_{2,3}=0, \quad \mathcal{H}_{2,4}=0,$$
$$\mathcal{H}_{3,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_2},0}, \quad \mathcal{H}_{3,2}=0, \quad \mathcal{H}_{3,3}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1}, \quad \mathcal{H}_{3,4}=0,$$
$$\mathcal{H}_{4,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_3},0}, \quad \mathcal{H}_{4,2}=0, \quad \mathcal{H}_{4,3}=0, \quad \mathcal{H}_{4,4}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1},$$

$$-\left(\frac{\partial \boldsymbol{V}_1}{\partial x}+\frac{\partial \boldsymbol{V}_2}{\partial y}+\frac{\partial \boldsymbol{V}_3}{\partial z}\right)=0$$

$$\boldsymbol{V}_1-\frac{\partial \phi}{\partial x}=0$$

$$\boldsymbol{V}_2-\frac{\partial \phi}{\partial y}=0$$

$$\boldsymbol{V}_3-\frac{\partial \phi}{\partial z}=0$$

Let $\boldsymbol{w}=(\phi,\ \boldsymbol{V}_1,\ \boldsymbol{V}_2,\ \boldsymbol{V}_3)^t$, and $\boldsymbol{e}_1=(1,0,0)^t$, $\boldsymbol{e}_2=(0,1,0)^t$, $\boldsymbol{e}_3=(0,0,1)^t$.

$$\mathcal{H}(\boldsymbol{w})=0 \tag{4.13}$$

with

$$\mathcal{H}_{1,1}=0, \quad \mathcal{H}_{1,2}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_1},\boldsymbol{0},0}, \quad \mathcal{H}_{1,3}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_2},\boldsymbol{0},0}, \quad \mathcal{H}_{1,4}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_3},\boldsymbol{0},0},$$

$$\mathcal{H}_{2,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_1},0}, \quad \mathcal{H}_{2,2}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1}, \quad \mathcal{H}_{2,3}=0, \quad \mathcal{H}_{2,4}=0,$$

$$\mathcal{H}_{3,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_2},0}, \quad \mathcal{H}_{3,2}=0, \quad \mathcal{H}_{3,3}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1}, \quad \mathcal{H}_{3,4}=0,$$

$$\mathcal{H}_{4,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_3},0}, \quad \mathcal{H}_{4,2}=0, \quad \mathcal{H}_{4,3}=0, \quad \mathcal{H}_{4,4}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1},$$

$$-\left(\frac{\partial V_1}{\partial x}+\frac{\partial V_2}{\partial y}+\frac{\partial V_3}{\partial z}\right)=0$$

$$V_1-\frac{\partial \phi}{\partial x}=0$$

$$V_2-\frac{\partial \phi}{\partial y}=0$$

$$V_3-\frac{\partial \phi}{\partial z}=0$$

Let $\boldsymbol{w}=(\phi,\ \boldsymbol{V}_1,\ \boldsymbol{V}_2,\ \boldsymbol{V}_3)^t$, and $\boldsymbol{e}_1=(1,0,0)^t$, $\boldsymbol{e}_2=(0,1,0)^t$, $\boldsymbol{e}_3=(0,0,1)^t$.

$$\mathcal{H}(\boldsymbol{w})=0 \tag{4.13}$$

with

$$\mathcal{H}_{1,1}=0, \quad \mathcal{H}_{1,2}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e}_1,\boldsymbol{0},0}, \quad \mathcal{H}_{1,3}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e}_2,\boldsymbol{0},0}, \quad \mathcal{H}_{1,4}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e}_3,\boldsymbol{0},0},$$

$$\mathcal{H}_{2,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e}_1,0}, \quad \mathcal{H}_{2,2}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1}, \quad \mathcal{H}_{2,3}=0, \quad \mathcal{H}_{2,4}=0,$$

$$\mathcal{H}_{3,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e}_2,0}, \quad \mathcal{H}_{3,2}=0, \quad \mathcal{H}_{3,3}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1}, \quad \mathcal{H}_{3,4}=0,$$

$$\mathcal{H}_{4,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e}_3,0}, \quad \mathcal{H}_{4,2}=0, \quad \mathcal{H}_{4,3}=0, \quad \mathcal{H}_{4,4}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1},$$

$$-\left(\frac{\partial \boldsymbol{V}_1}{\partial x}+\frac{\partial \boldsymbol{V}_2}{\partial y}+\frac{\partial \boldsymbol{V}_3}{\partial z}\right)=0$$

$$\boldsymbol{V}_1-\frac{\partial \phi}{\partial x}=0$$

$$\boldsymbol{V}_2-\frac{\partial \phi}{\partial y}=0$$

$$\boldsymbol{V}_3-\frac{\partial \phi}{\partial z}=0$$

Let $\boldsymbol{w}=(\phi,\ \boldsymbol{V}_1,\ \boldsymbol{V}_2,\ \boldsymbol{V}_3)^t$, and $\boldsymbol{e}_1=(1,0,0)^t$, $\boldsymbol{e}_2=(0,1,0)^t$, $\boldsymbol{e}_3=(0,0,1)^t$.

$$\mathcal{H}(\boldsymbol{w})=0 \tag{4.13}$$

with

$$\mathcal{H}_{1,1}=0, \quad \mathcal{H}_{1,2}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_1},\boldsymbol{0},0}, \quad \mathcal{H}_{1,3}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_2},\boldsymbol{0},0}, \quad \mathcal{H}_{1,4}=\mathcal{L}_{\mathbb{O},-\boldsymbol{e_3},\boldsymbol{0},0},$$
$$\mathcal{H}_{2,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_1},0}, \quad \mathcal{H}_{2,2}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1}, \quad \mathcal{H}_{2,3}=0, \quad \mathcal{H}_{2,4}=0,$$
$$\mathcal{H}_{3,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_2},0}, \quad \mathcal{H}_{3,2}=0, \quad \mathcal{H}_{3,3}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1}, \quad \mathcal{H}_{3,4}=0,$$
$$\mathcal{H}_{4,1}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},-\boldsymbol{e_3},0}, \quad \mathcal{H}_{4,2}=0, \quad \mathcal{H}_{4,3}=0, \quad \mathcal{H}_{4,4}=\mathcal{L}_{\mathbb{O},\boldsymbol{0},\boldsymbol{0},1},$$

$$\frac{\partial \boldsymbol{w}_1}{\partial n_{\mathcal{H}_{1,1}}} = 0, \quad \frac{\partial \boldsymbol{w}_1}{\partial n_{\mathcal{H}_{2,1}}} = 0, \quad \frac{\partial \boldsymbol{w}_1}{\partial n_{\mathcal{H}_{3,1}}} = 0, \quad \frac{\partial \boldsymbol{w}_1}{\partial n_{\mathcal{H}_{4,1}}} = 0,$$
$$\frac{\partial \boldsymbol{w}_2}{\partial n_{\mathcal{H}_{1,2}}} = \boldsymbol{V}_1 \boldsymbol{n}_1, \quad \frac{\partial \boldsymbol{w}_2}{\partial n_{\mathcal{H}_{2,2}}} = 0, \quad \frac{\partial \boldsymbol{w}_2}{\partial n_{\mathcal{H}_{3,2}}} = 0, \quad \frac{\partial \boldsymbol{w}_2}{\partial n_{\mathcal{H}_{4,2}}} = 0,$$
$$\frac{\partial \boldsymbol{w}_3}{\partial n_{\mathcal{H}_{1,3}}} = \boldsymbol{V}_2 \boldsymbol{n}_2, \quad \frac{\partial \boldsymbol{w}_3}{\partial n_{\mathcal{H}_{2,3}}} = 0, \quad \frac{\partial \boldsymbol{w}_3}{\partial n_{\mathcal{H}_{3,3}}} = 0, \quad \frac{\partial \boldsymbol{w}_3}{\partial n_{\mathcal{H}_{4,3}}} = 0,$$
$$\frac{\partial \boldsymbol{w}_4}{\partial n_{\mathcal{H}_{1,4}}} = \boldsymbol{V}_3 \boldsymbol{n}_3, \quad \frac{\partial \boldsymbol{w}_4}{\partial n_{\mathcal{H}_{2,4}}} = 0, \quad \frac{\partial \boldsymbol{w}_4}{\partial n_{\mathcal{H}_{3,4}}} = 0, \quad \frac{\partial \boldsymbol{w}_4}{\partial n_{\mathcal{H}_{4,4}}} = 0,$$

So

$$\sum_{\alpha=1}^{4} \frac{\partial \boldsymbol{w}_\alpha}{\partial n_{\mathcal{H}_{1,\alpha}}} = \langle \boldsymbol{V}, \boldsymbol{n} \rangle = \langle \nabla \phi, \boldsymbol{n} \rangle, \tag{4.14}$$

and

$$\sum_{\alpha=1}^{4} \frac{\partial \boldsymbol{w}_\alpha}{\partial n_{\mathcal{H}_{2,\alpha}}} = \sum_{\alpha=1}^{4} \frac{\partial \boldsymbol{w}_\alpha}{\partial n_{\mathcal{H}_{3,\alpha}}} = \sum_{\alpha=1}^{4} \frac{\partial \boldsymbol{w}_\alpha}{\partial n_{\mathcal{H}_{4,\alpha}}} = 0. \tag{4.15}$$

Algorithm 19 Stationary heat with potential velocity problem

$\mathcal{T}_h \leftarrow$ getMesh(...) ▷ Load FreeFEM++ mesh

$\boldsymbol{e}_1 \leftarrow \begin{pmatrix}1\\0\\0\end{pmatrix}, \quad \boldsymbol{e}_2 \leftarrow \begin{pmatrix}0\\1\\0\end{pmatrix}, \quad \boldsymbol{e}_3 \leftarrow \begin{pmatrix}0\\0\\1\end{pmatrix}$

Hop $\leftarrow$ Hoperator$(4,4)$

Hop.H$(1,2) \leftarrow$ Loperator$(\mathbb{O}_3, -\boldsymbol{e}_1, \mathbf{0}, 0)$

Hop.H$(1,3) \leftarrow$ Loperator$(\mathbb{O}_3, -\boldsymbol{e}_2, \mathbf{0}, 0)$

Hop.H$(1,4) \leftarrow$ Loperator$(\mathbb{O}_3, -\boldsymbol{e}_3, \mathbf{0}, 0)$

Hop.H$(2,1) \leftarrow$ Loperator$(\mathbb{O}_3, \mathbf{0}, -\boldsymbol{e}_1, 0)$, Hop.H$(2,2) \leftarrow$ Loperator$(\mathbb{O}_3, \mathbf{0}, \mathbf{0}, 1)$

Hop.H$(3,1) \leftarrow$ Loperator$(\mathbb{O}_3, \mathbf{0}, -\boldsymbol{e}_2, 0)$, Hop.H$(3,3) \leftarrow$ Loperator$(\mathbb{O}_3, \mathbf{0}, \mathbf{0}, 1)$

Hop.H$(4,1) \leftarrow$ Loperator$(\mathbb{O}_3, \mathbf{0}, -\boldsymbol{e}_3, 0)$, Hop.H$(4,4) \leftarrow$ Loperator$(\mathbb{O}_3, \mathbf{0}, \mathbf{0}, 1)$

PDEflow $\leftarrow$ initPDE(Hop, $\mathcal{T}_h$)

PDEflow $\leftarrow$ setBC_PDE(PDEflow, $[1021, 2021]$, 1, 'Dirichlet', 1., $\varnothing$)

PDEflow $\leftarrow$ setBC_PDE(PDEflow, $[1020, 2020]$, 1, 'Dirichlet', $-1.$, $\varnothing$)

$[\boldsymbol{\phi}, \boldsymbol{V}_1, \boldsymbol{V}_2, \boldsymbol{V}_3] \leftarrow$ SolvePDE(PDEflow)

$\alpha \leftarrow (x,y,z) \longmapsto 1$

$g_{20} \leftarrow (x,y,z) \longmapsto 30, \quad g_{10} \leftarrow (x,y,z) \longmapsto 10 * (|z-1| > 0.5)$

$\beta \leftarrow 0.01$

Dop $\leftarrow$ Loperator$\left(\begin{pmatrix}\alpha & 0 & 0\\ 0 & \alpha & 0\\ 0 & 0 & \alpha\end{pmatrix}, \mathbf{0}, \begin{pmatrix}\boldsymbol{V}_1\\ \boldsymbol{V}_2\\ \boldsymbol{V}_3\end{pmatrix}, \beta\right)$

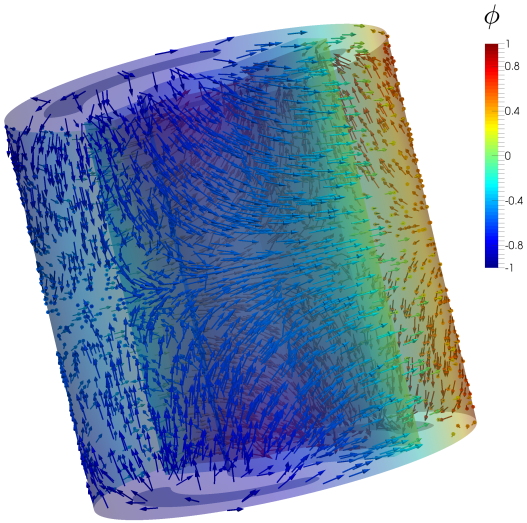
PDE $\leftarrow$ initPDE(Dop, $\mathcal{T}_h$) ▷ Set homogeneous 'Neumann' condition on all boundaries

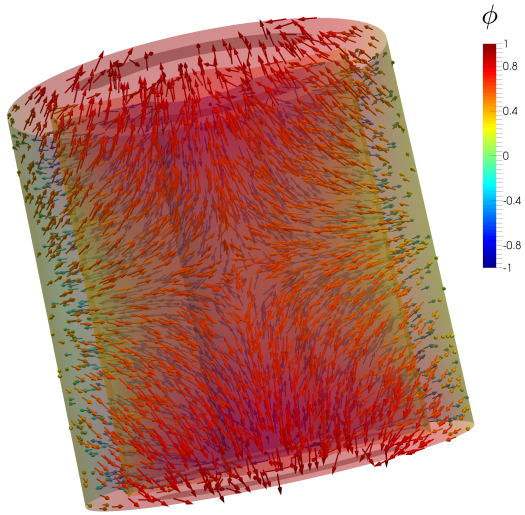
PDE $\leftarrow$ setBC_PDE(PDE, 1020, 1, 'Dirichlet', g_{20}, $\varnothing$)

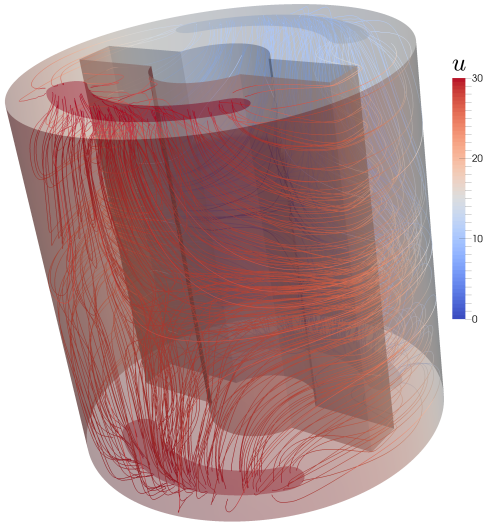
PDE $\leftarrow$ setBC_PDE(PDE, 2020, 1, 'Dirichlet', g_{20}, $\varnothing$)

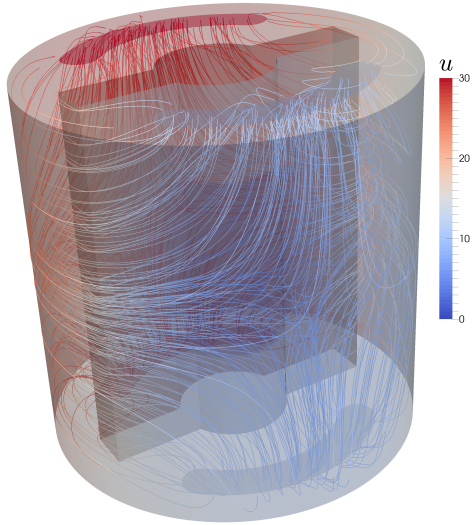
PDE $\leftarrow$ setBC_PDE(PDE, 10, 1, 'Dirichlet', g_{10}, $\varnothing$)

$\boldsymbol{u} \leftarrow$ SolvePDE(PDE)


ϕ
1
0.8
0.4
0
-0.4
-0.8
-1


ϕ
1
0.8
0.4
0
-0.4
-0.8
-1


u
30
20
10
0


u
30
20
10
0

Clamped plate

From the thesis of T. Gerasimov[2009] (page 138). Let $\Omega = [-1, 6] \times [-1, 1]$ and $f := \exp(-100((x + 0.75)^2 + (y - 0.75)^2))$.

$$\begin{cases} \Delta^2 u = f, & \text{in } \Omega \\ u = 0, & \text{on } \Gamma \\ \frac{\partial u}{\partial \boldsymbol{n}} = 0. & \text{on } \Gamma \end{cases} \iff (\mathcal{P}) \begin{cases} v = -\Delta u, & \text{in } \Omega \\ -\Delta v = f, & \text{in } \Omega \\ u = 0, & \text{on } \Gamma \\ \frac{\partial u}{\partial \boldsymbol{n}} = 0. & \text{on } \Gamma \end{cases}$$

Let $\boldsymbol{w} = (u, v) \in \mathrm{H}_0^1(\Omega) \times \mathrm{H}^1(\Omega)$, the $\mathcal{H}$-operator such that

$$(\mathcal{P}_1), (\mathcal{P}_2) \iff \mathcal{H} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix} \text{ where } \mathcal{H} = \begin{pmatrix} 0 & \mathcal{L}_{\mathbb{I},\mathbf{0},\mathbf{0},0} \\ \mathcal{L}_{\mathbb{I},\mathbf{0},\mathbf{0},0} & \mathcal{L}_{\mathbb{O},\mathbf{0},\mathbf{0},-1} \end{pmatrix}$$

and

$$\frac{\partial \boldsymbol{w}}{\partial_{\mathcal{H}_1}} = \frac{\partial \boldsymbol{w}_2}{\partial \boldsymbol{n}} = \frac{\partial v}{\partial \boldsymbol{n}} \text{ and } \frac{\partial \boldsymbol{w}}{\partial_{\mathcal{H}_2}} = \frac{\partial \boldsymbol{w}_1}{\partial \boldsymbol{n}} = \frac{\partial u}{\partial \boldsymbol{n}}$$

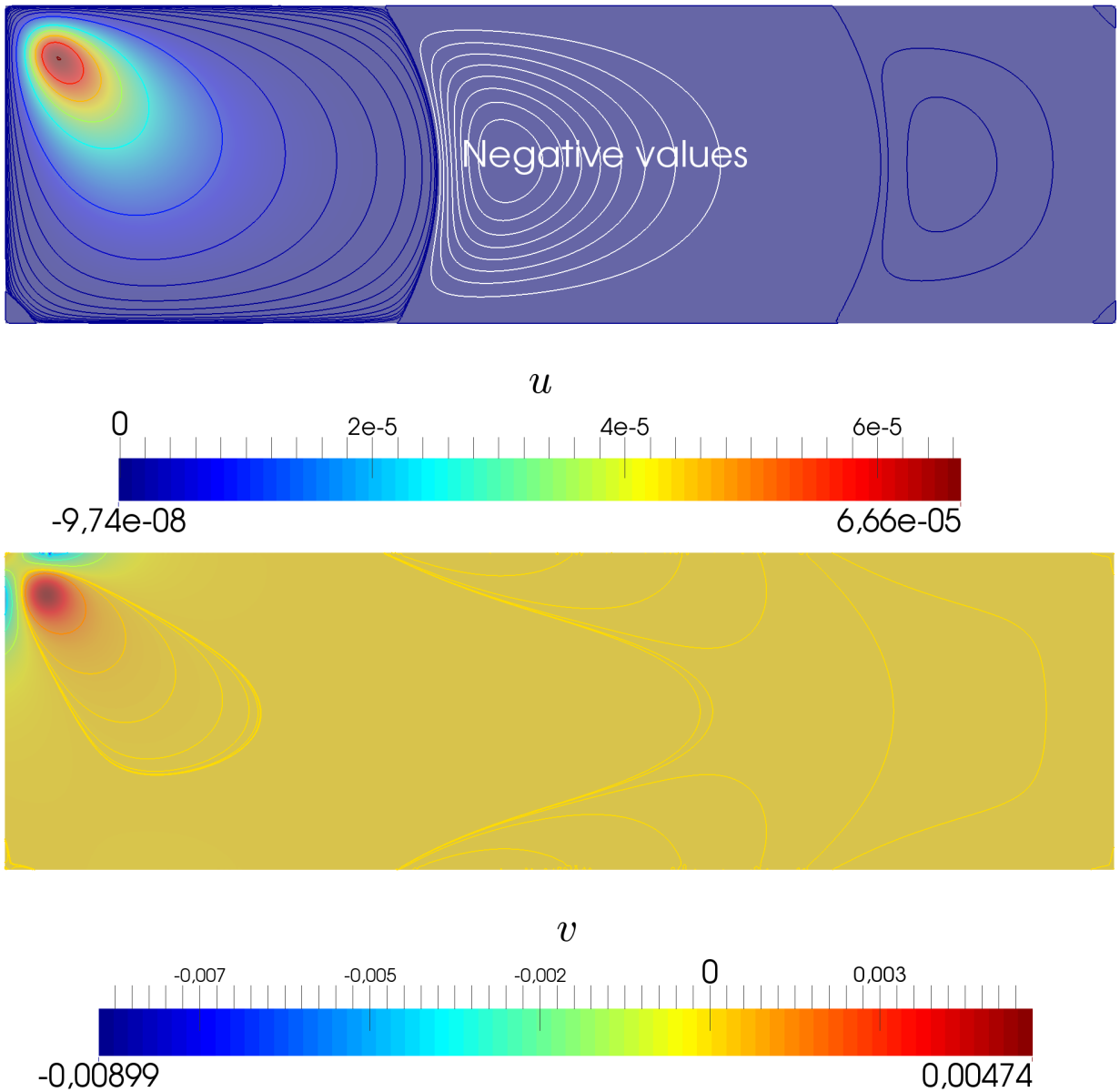
Clamped plate

$$\begin{cases} \mathcal{H}(\boldsymbol{w}) & = \begin{pmatrix} f \\ 0 \end{pmatrix}, \quad \text{in } \Omega \\ \boldsymbol{w}_1 & = 0, \quad \text{on } \Gamma_1^D = \Gamma \\ \frac{\partial \boldsymbol{w}}{\partial_{\mathcal{H}_2}} & = 0, \quad \text{on } \Gamma_2^R = \Gamma \end{cases} \qquad \text{where } \mathcal{H} = \begin{pmatrix} 0 & \mathcal{L}_{\mathbb{I},\mathbf{0},\mathbf{0},0} \\ \mathcal{L}_{\mathbb{I},\mathbf{0},\mathbf{0},0} & \mathcal{L}_{\mathbb{O},\mathbf{0},\mathbf{0},-1} \end{pmatrix}$$

Algorithm 20 Clamped plate problem

1: $\mathcal{T}_h \leftarrow \text{getMesh}(...)$ ▷ Load FreeFEM++ mesh
2: $\texttt{Hop} \leftarrow \text{Hoperator}(2,2)$
3: $\texttt{Hop.H}(1,2) \leftarrow \text{Loperator}(2, \mathbb{I}_2, \mathbf{0}, \mathbf{0}, 0)$
4: $\texttt{Hop.H}(2,1) \leftarrow \text{Loperator}(2, \mathbb{I}_2, \mathbf{0}, \mathbf{0}, 0)$
5: $\texttt{Hop.H}(2,2) \leftarrow \text{Loperator}(2, \mathbb{O}_2, \mathbf{0}, \mathbf{0}, -1)$
6: $\text{pde} \leftarrow \text{initPDE}(\texttt{Hop}, \mathcal{T}_h)$
7: $\text{pde.f} \leftarrow (x,y) \mapsto \begin{pmatrix} \exp(-100((x+0.75)^2 + (y-0.75)^2)) \\ 0 \end{pmatrix}$
8: $\text{pde} \leftarrow \text{setBC\_PDE}(\text{pde}, 1, 1, \texttt{'Dirichlet'}, 0.)$
9: $[\boldsymbol{u}, \boldsymbol{v}] \leftarrow \text{SolvePDE}(\text{PDE})$

Clamped plate



Eigenvalue problems

Scalar eigenvalue problem

Find $(\lambda, u) \in \mathbb{K} \times \mathrm{H}^2(\Omega)$ such that

$$\mathcal{L}(u) = \lambda \mathcal{B}(u) \quad \text{in } \Omega, \tag{4.16}$$

$$u = 0 \quad \text{on } \Gamma^D, \tag{4.17}$$

$$\frac{\partial u}{\partial n_{\mathcal{L}}} + a^R u = 0 \quad \text{on } \Gamma^R. \tag{4.18}$$

Algorithm 21 function EigsPDE : solving *scalar* or *vector* eigenvalue problems

1: **Function** $[\boldsymbol{U}, \boldsymbol{\lambda}] \leftarrow \mathrm{EigsPDE}(\mathrm{pde},\ \mathrm{Bop},\ N_e)$
2: $\quad \mathrm{n_{dof}} \leftarrow \mathrm{pde.m} \times \mathrm{pde}.\mathcal{T}_h.\mathrm{n_q}$
3: $\quad \mathbb{A} \leftarrow \mathrm{AssemblyP1\_OptV3}(\mathrm{pde}.\mathcal{T}_h, \mathrm{pde.Op})$
4: $\quad [\mathbb{M}^R, \boldsymbol{F}^R] \leftarrow \mathrm{RobinBC}(\mathrm{pde})$
5: $\quad \mathbb{A} \leftarrow \mathbb{A} + \mathbb{M}^R$
6: $\quad \mathbb{B} \leftarrow \mathrm{AssemblyP1\_OptV3}(\mathrm{pde}.\mathcal{T}_h, \mathrm{Bop})$
7: $\quad [\boldsymbol{R}^D, \boldsymbol{I}_D, \boldsymbol{I}_D^c] \leftarrow \mathrm{DirichletBC}(\mathrm{pde})$
8: $\quad \boldsymbol{U} \leftarrow \mathbb{O}_{\mathrm{n_{dof}}, N_e}$
9: $\quad [\boldsymbol{U}(\boldsymbol{I}_D^c, :), \boldsymbol{\lambda}] \leftarrow \mathrm{Eigs}(\mathbb{A}(\boldsymbol{I}_D^c, \boldsymbol{I}_D^c), \mathbb{B}(\boldsymbol{I}_D^c, \boldsymbol{I}_D^c), N_e)$
10: **end Function**

Laplace problems

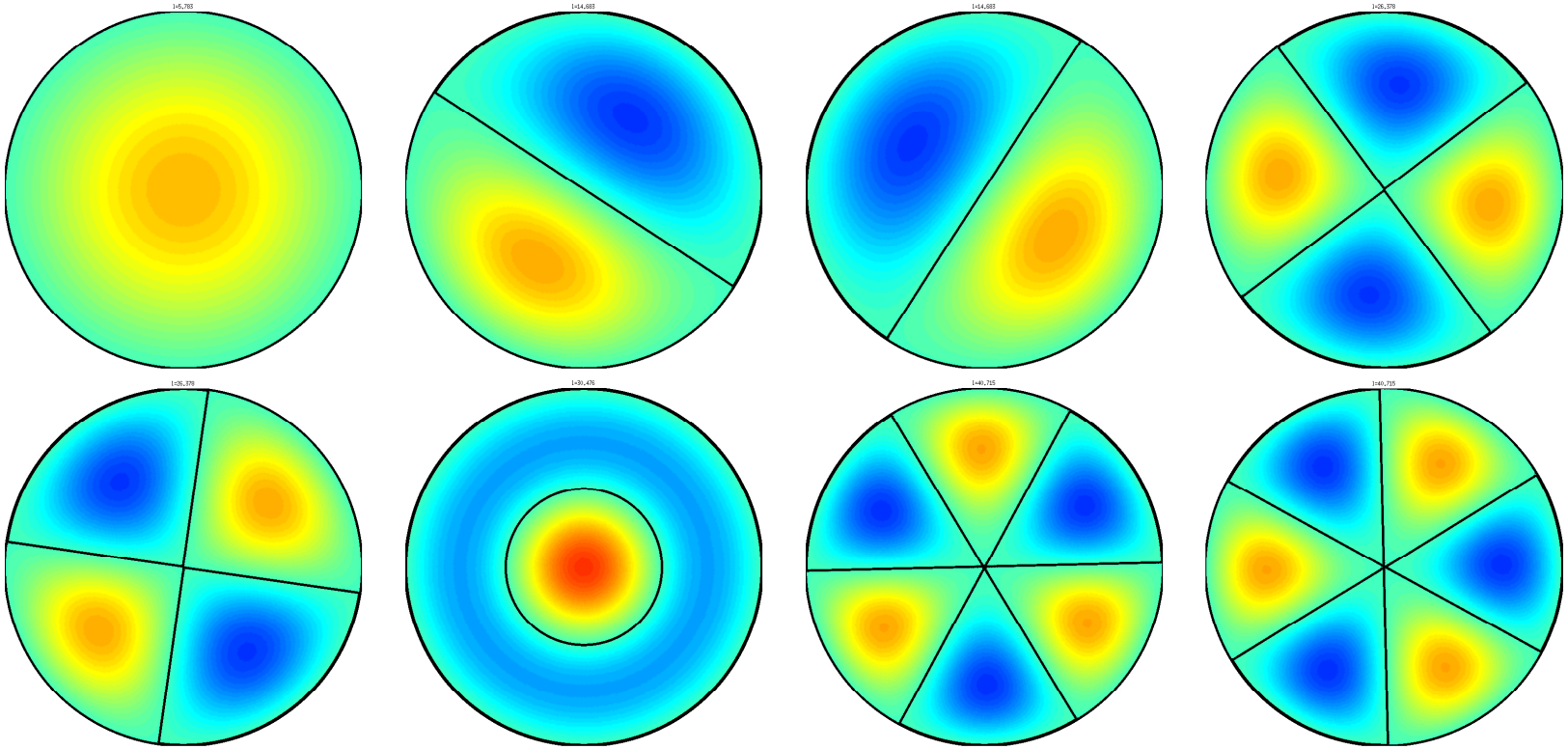
$$-\Delta u = \lambda u \qquad \text{in } \Omega$$
$$u = 0 \qquad \text{on } \Gamma$$

$\texttt{Lop} \leftarrow \text{Loperator}(2, \mathbb{I}_2, \mathbf{0}, \mathbf{0}, 0)$
$\text{pde} \leftarrow \text{initPDE}(\texttt{Lop}, \mathcal{T}_h)$
for $i \leftarrow 1$ **to** $\text{pde.n}_{\text{lab}}$ **do**
  $\text{pde} \leftarrow \text{setBC\_PDE}(\text{pde}, i, 1, \texttt{'Dirichlet'}, 0)$
end for
$\texttt{LMass} \leftarrow \text{Loperator}(2, \mathbb{O}_2, \mathbf{0}, \mathbf{0}, 1)$
$[\boldsymbol{u}, \boldsymbol{\lambda}] \leftarrow \text{EigsPDE}(\text{pde}, \texttt{LMass}, 16)$

Laplace problems

$$-\Delta u = \lambda u \quad \text{in } \Omega$$
$$u = 0 \quad \text{on } \Gamma$$

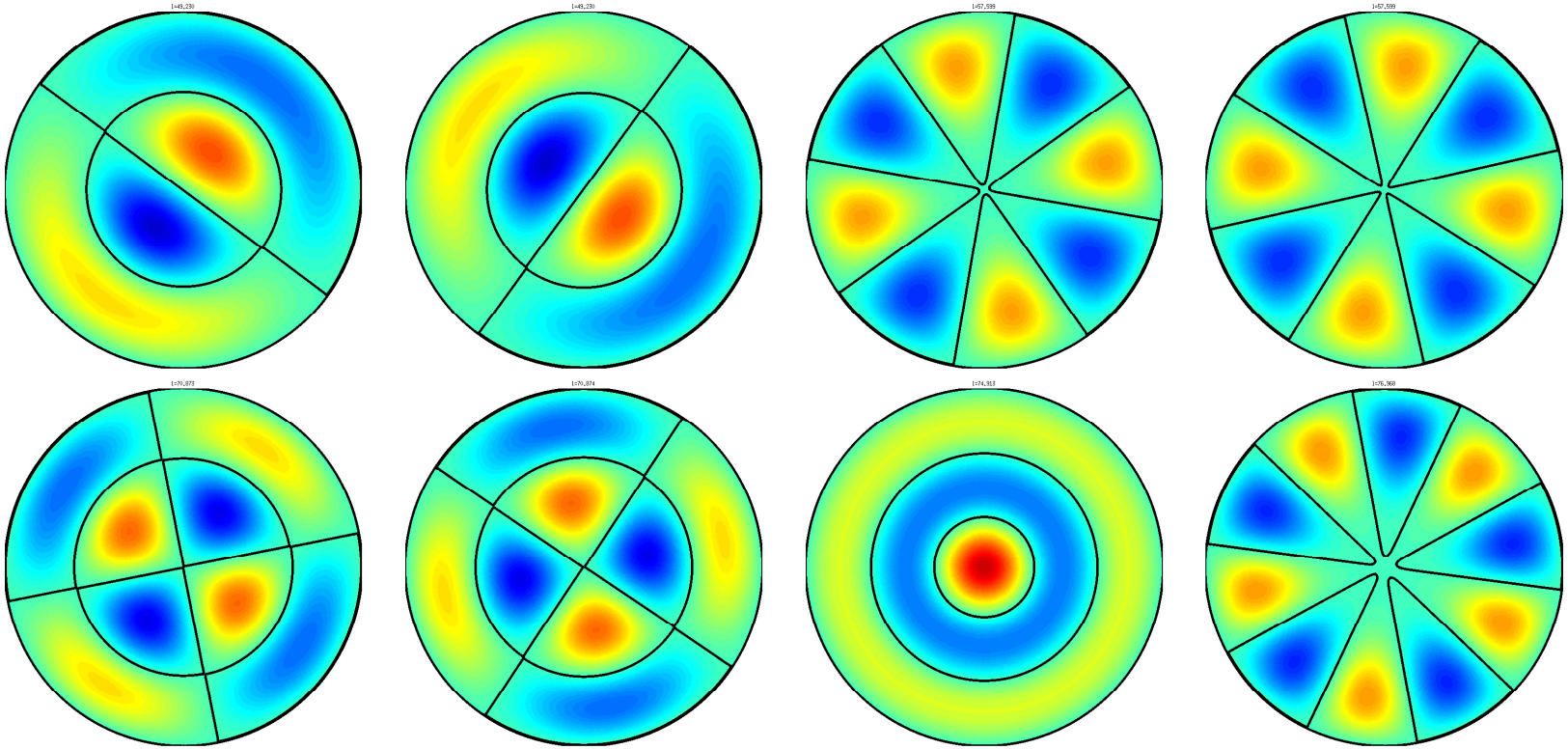
$\texttt{Lop} \leftarrow \text{Loperator}(2, \mathbb{I}_2, \mathbf{0}, \mathbf{0}, 0)$
$\text{pde} \leftarrow \text{initPDE}(\texttt{Lop}, \mathcal{T}_h)$
for $i \leftarrow 1$ **to** $\text{pde.n}_{\text{lab}}$ **do**
  $\text{pde} \leftarrow \text{setBC\_PDE}(\text{pde}, i, 1, \texttt{'Dirichlet'}, 0)$
end for
$\texttt{LMass} \leftarrow \text{Loperator}(2, \mathbb{O}_2, \mathbf{0}, \mathbf{0}, 1)$
$[\boldsymbol{u}, \boldsymbol{\lambda}] \leftarrow \text{EigsPDE}(\text{pde}, \texttt{LMass}, 16)$



Laplace problems

$$-\Delta u = \lambda u \quad \text{in } \Omega$$
$$u = 0 \quad \text{on } \Gamma$$

$\texttt{Lop} \leftarrow \text{Loperator}(2, \mathbb{I}_2, \mathbf{0}, \mathbf{0}, 0)$
$\text{pde} \leftarrow \text{initPDE}(\texttt{Lop}, \mathcal{T}_h)$
for $i \leftarrow 1$ **to** $\text{pde.n}_{\text{lab}}$ **do**
  $\text{pde} \leftarrow \text{setBC\_PDE}(\text{pde}, i, 1, \texttt{'Dirichlet'}, 0)$
end for
$\texttt{LMass} \leftarrow \text{Loperator}(2, \mathbb{O}_2, \mathbf{0}, \mathbf{0}, 1)$
$[\boldsymbol{u}, \boldsymbol{\lambda}] \leftarrow \text{EigsPDE}(\text{pde}, \texttt{LMass}, 16)$



Outline

Conclusion and prospects

- Very short algorithms to solve BVP's in any dimension, ...
  Less than 400 lines in Matlab, Octave, Python, Scilab
- Easy to read (so easy to write and to maintain)
- Efficient

To do :

- Add time-dependent PDE's,
- Add various types of finite element spaces (P_k, P_1-bubble, RT0, ...),
- Add quadrature formulas,
- Add samples using Domain Decomposition Methods,
- Parallel computing, GPU, ...

http://www.math.univ-paris13.fr/∼cuvelier/software

Conclusion and prospects

- Very short algorithms to solve BVP's in any dimension, ...
  Less than 400 lines in Matlab, Octave, Python, Scilab
- Easy to read (so easy to write and to maintain)
- Efficient

To do :

- Add time-dependent PDE's,
- Add various types of finite element spaces (P_k, P_1-bubble, RT0, ...),
- Add quadrature formulas,
- Add samples using Domain Decomposition Methods,
- Parallel computing, GPU, ...

http://www.math.univ-paris13.fr/∼cuvelier/software