FC-VFEM$\mathbb{P}_1$-SURFACE Python package, User's Guide [1]

François Cuvelier[2]

2017/08/14

[1]Compiled with Python 3.6.0

[2]Université Paris 13, Sorbonne Paris Cité, LAGA, CNRS UMR 7539, 99 Avenue J-B Clément, F-93430 Villetaneuse, France, cuvelier@math.univ-paris13.fr.

This work was partially supported by ANR Dedales.

Abstract

FC-VFEM$\mathbb{P}_1$-SURFACE is an **experimental** object-oriented Python package dedicated to solve **surface** boundary value problems (BVP) by using $\mathbb{P}^1$-Lagrange finite element method in any space dimension. This package is an add-on to the FC-VFEM$\mathbb{P}_1$ package [**?**]. It uses the FC-SIMESH package [1] and the `siMesh` class which allows to use simplices meshes generated from gmsh (in dimension 2 or 3) or an hypercube triangulation (in any dimension).

The two FC-SIMESH add-ons FC-SIMESH-MATPLOTLIB [2] and FC-SIMESH-MAYAVI [3] allows a great flexibility in graphical representations of the meshes and datas on the meshes by using respectively the MATPLOTLIB and the MAYAVI packages.

The FC-VFEM$\mathbb{P}_1$ package also contains the techniques of vectorization presented in [5] and extended in [4] and allows good performances when using $\mathbb{P}^1$-Lagrange finite element method.

Contents

Let $\Gamma \subset \mathbb{R}^{n+1}$ be a regular surface. For all $x \in \Gamma$, $\mathbb{A}(x) = (A_{i,j}(x))_{i,j=1,\dots,n+1}$ be a matrix which maps the tangent space $T_x\Gamma$ into itself (i.e. $\mathbb{A}(x) : T_x\Gamma \longrightarrow T_x\Gamma$) and let $\boldsymbol{b}(x) \in T_x\Gamma$, $\boldsymbol{c} : \Gamma \longrightarrow \mathbb{R}^{n+1}$ and $a_0 : \Gamma \longrightarrow \mathbb{R}$. We denote by $\mathcal{L}$ the second order linear differential operator acting on scalar fields defined, $\forall u \in H^2(\Gamma)$, by

$$\mathcal{L}(u) \stackrel{\text{def}}{=} -\operatorname{div}_\Gamma(\mathbb{A}\,\boldsymbol{\nabla}_\Gamma\, u) + \operatorname{div}_\Gamma(\boldsymbol{b}u) + \langle \boldsymbol{\nabla}_\Gamma\, u, \boldsymbol{c}\rangle + a_0 u. \tag{1}$$

The main of the **Surface BVP Addon** is to solve generic *scalar* surface BVP given by

Scalar surface BVP 1 : generic problem

Find $u \in H^2(\Gamma)$ such that

$$\mathcal{L}(u) = f \qquad \text{in } \Gamma, \tag{2}$$

$$u = g^D \qquad \text{on } \partial\Gamma^D, \tag{3}$$

$$\frac{\partial u}{\partial n_{\mathcal{L}}} + a^R u = g^R \qquad \text{on } \partial\Gamma^R. \tag{4}$$

The **conormal derivative** of u is defined by

$$\frac{\partial u}{\partial n_{\mathcal{L}}} \stackrel{\text{def}}{=} \langle \mathbb{A}\,\boldsymbol{\nabla}_\Gamma\, u, \boldsymbol{\mu}\rangle - \langle \boldsymbol{b}u, \boldsymbol{\mu}\rangle \tag{5}$$

Before getting to the heart of the matter, the first chapter rapidly presents the FC-VFEM$\mathbb{P}_1$ package. For a more complete description one can refers to []. Thereafter, notations and results on regular surfaces given in [7] are recalled. Finally some surface BVPs are presented and numericaly solved by using the **Surface BVP Addon** for the FC-VFEM$\mathbb{P}_1$ Python package.

Chapter 1

Generic Boundary Value Problems

The notations of [8] are employed in this section and extended to the vector case.

1.1 Scalar boundary value problem

Let Ω be a bounded open subset of $\mathbb{R}^d$, $d \geqslant 1$. The boundary of Ω is denoted by Γ.

We denote by $\mathcal{L}_{\mathbb{A},\boldsymbol{b},\boldsymbol{c},a_0} = \mathcal{L} : \mathrm{H}^2(\Omega) \longrightarrow L^2(\Omega)$ the second order linear differential operator acting on *scalar fields* defined, $\forall u \in \mathrm{H}^2(\Omega)$, by

$$\mathcal{L}_{\mathbb{A},\boldsymbol{b},\boldsymbol{c},a_0}(u) \stackrel{\mathsf{def}}{=} -\operatorname{div}\left(\mathbb{A}\,\boldsymbol{\nabla}\, u\right) + \operatorname{div}\left(\boldsymbol{b}u\right) + \langle \boldsymbol{\nabla}\, u, \boldsymbol{c}\rangle + a_0 u \tag{1.1}$$

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 $a_0 \in L^\infty(\Omega)$ are given functions and $\langle \cdot, \cdot \rangle$ is the usual scalar product in $\mathbb{R}^d$. We use the same notations as in the chapter 6 of [8] and we note that we can omit either $\operatorname{div}(\boldsymbol{b}u)$ or $\langle \boldsymbol{\nabla}\, u, \boldsymbol{c}\rangle$ if $\boldsymbol{b}$ and $\boldsymbol{c}$ are sufficiently regular functions. We keep both terms with $\boldsymbol{b}$ and $\boldsymbol{c}$ to deal with more boundary conditions. It should be also noted that it is important to preserve the two terms $\boldsymbol{b}$ and $\boldsymbol{c}$ in the generic formulation to enable a greater flexibility in the choice of the boundary conditions.

Let Γ^D, Γ^R be open subsets of Γ, possibly empty and $f \in L^2(\Omega)$, $g^D \in \mathrm{H}^{1/2}(\Gamma^D)$, $g^R \in L^2(\Gamma^R)$, $a^R \in L^\infty(\Gamma^R)$ be given data.

A *scalar* boundary value problem is given by

Scalar BVP 1 : generic problem

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

$$\mathcal{L}(u) = f \qquad \text{in } \Omega, \tag{1.2}$$

$$u = g^D \qquad \text{on } \Gamma^D, \tag{1.3}$$

$$\frac{\partial u}{\partial n_{\mathcal{L}}} + a^R u = g^R \qquad \text{on } \Gamma^R. \tag{1.4}$$

The **conormal derivative** of u is defined by

$$\frac{\partial u}{\partial n_{\mathcal{L}}} \stackrel{\mathsf{def}}{=} \langle \mathbb{A}\,\boldsymbol{\nabla}\, u, \boldsymbol{n}\rangle - \langle \boldsymbol{b}u, \boldsymbol{n}\rangle \tag{1.5}$$

The boundary conditions (1.3) and (1.4) are respectively **Dirichlet** and **Robin** boundary conditions. **Neumann** boundary conditions are particular Robin boundary conditions with $a^R \equiv 0$.

To have an outline of the FC-VFEM$\mathbb{P}_1$ package, a first and simple problem is quickly present. Explanations will be given in next sections.

The problem to solve is the Laplace problem for a condenser.

Usual BVP 1 : 2D condenser problem

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

$$-\Delta u = 0 \text{ in } \Omega \subset \mathbb{R}^2, \tag{1.6}$$
$$u = 0 \text{ on } \Gamma_1, \tag{1.7}$$
$$u = -12 \text{ on } \Gamma_{98}, \tag{1.8}$$
$$u = 12 \text{ on } \Gamma_{99}, \tag{1.9}$$

where Ω and its boundaries are given in Figure 1.1.

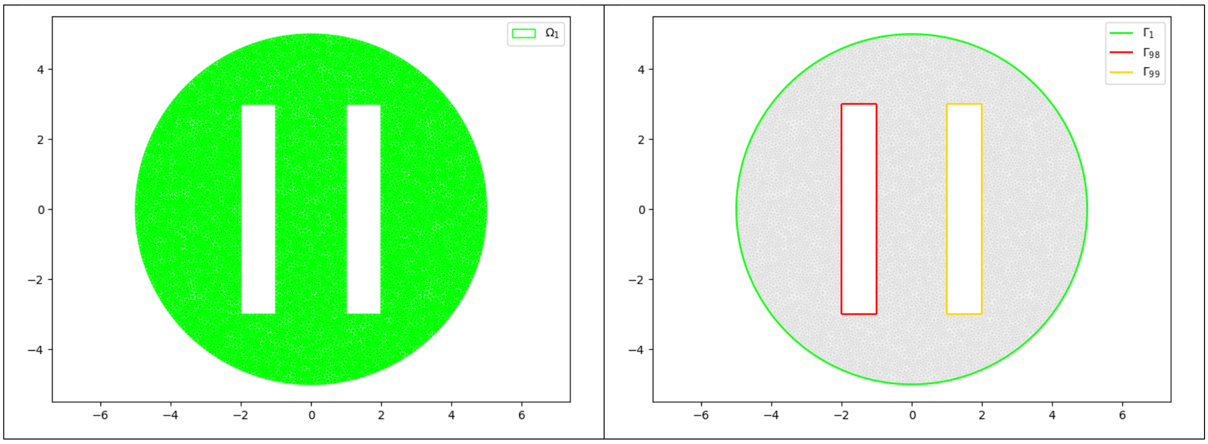


Figure 1.1: 2D condenser mesh and boundaries (left) and numerical solution (right)

The problem (1.6)-(1.9) can be equivalently expressed as the scalar BVP (1.2)-(1.4) :

Scalar BVP 2 : 2D condenser problem

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

$$\mathcal{L}(u) = f \quad \text{in } \Omega,$$
$$u = g^D \quad \text{on } \Gamma^D = \Gamma_1 \cup \Gamma_{98} \cup \Gamma_{99}.$$

where $\mathcal{L} := \mathcal{L}_{\mathbb{I},\mathbf{0},\mathbf{0},0}$, $f \equiv 0$, and

$$g^D := 0 \text{ on } \Gamma_1, \; g^D := -12 \text{ on } \Gamma_{98}, \; g^D := +12 \text{ on } \Gamma_{99}$$

In Listing 27 a complete code is given to solve this problem.

```
meshfile=gmsh.buildmesh2d('condenser',10) # generate mesh
Th=siMesh(meshfile) # read mesh
Lop=Loperator(dim=2,d=2,A=[[1,0],[0,1]])
pde=PDE(Op=Lop)
bvp=BVP(Th,pde=pde)
bvp.setDirichlet( 1, 0.)
bvp.setDirichlet( 98, -12.)
bvp.setDirichlet( 99, +12.)
u=bvp.solve();
# Graphic parts
plt.figure(1)
siplt.plotmesh(Th,legend=True)
set_axes_equal()
plt.figure(2)
siplt.plotmesh(Th,color='LightGray',alpha=0.3)
siplt.plotmesh(Th,d=1,legend=True)
```

```
set_axes_equal()
plt.figure(3)
siplt.plot(Th,u)
plt.colorbar(label='u')
set_axes_equal()
plt.figure(4)
siplt.plotiso(Th,u,contours=15)
plt.colorbar(label='u')
siplt.plotmesh(Th,color='LightGray',alpha=0.3)
plt.axis('off');set_axes_equal()
```

Listing 1.1: Complete Python code to solve the 2D condenser problem with graphical representations

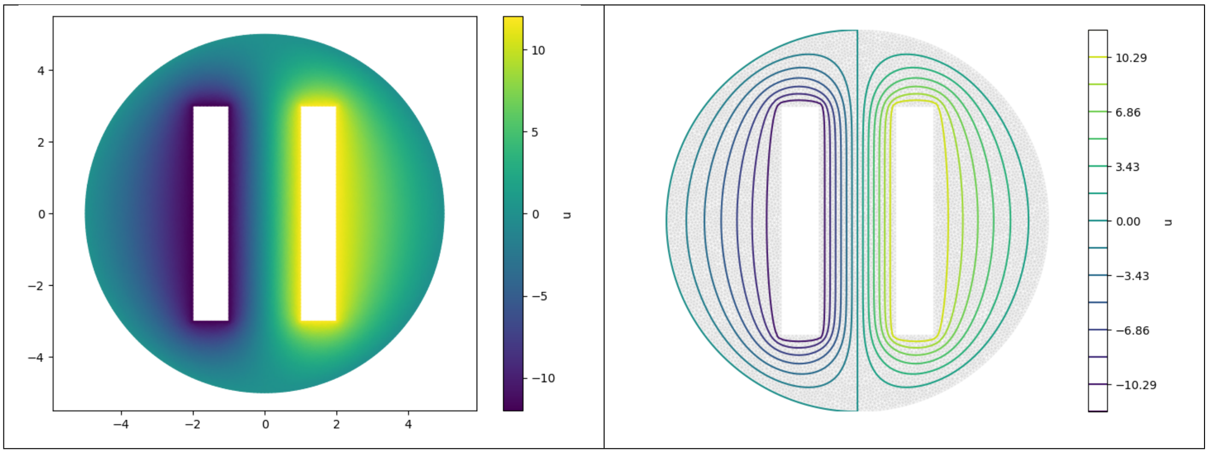


Figure 1.2: 2D condenser numerical solution

1.2 Vector boundary value problem

Let $m \geqslant 1$ and $\mathcal{H}$ be the m-by-m matrix of second order linear differential operators defined by

$$\left\{ \begin{array}{llcl} \mathcal{H} & : & (\mathrm{H}^2(\Omega))^m & \longrightarrow & (L^2(\Omega))^m \\ & & \boldsymbol{u} = (\boldsymbol{u}_1, \ldots, \boldsymbol{u}_m) & \longmapsto & \boldsymbol{f} = (\boldsymbol{f}_1, \ldots, \boldsymbol{f}_m) \stackrel{\textsf{def}}{=} \mathcal{H}(\boldsymbol{u}) \end{array} \right. \tag{1.10}$$

where

$$\boldsymbol{f}_\alpha = \sum_{\beta=1}^{m} \mathcal{H}_{\alpha,\beta}(\boldsymbol{u}_\beta), \quad \forall \alpha \in [\![1, m]\!], \tag{1.11}$$

with, for all $(\alpha, \beta) \in [\![1, m]\!]^2$,

$$\mathcal{H}_{\alpha,\beta} \stackrel{\textsf{def}}{=} \mathcal{L}_{\mathbb{A}^{\alpha,\beta},\boldsymbol{b}^{\alpha,\beta},\boldsymbol{c}^{\alpha,\beta},a_0^{\alpha,\beta}} \tag{1.12}$$

and $\mathbb{A}^{\alpha,\beta} \in (L^\infty(\Omega))^{d\times d}$, $\boldsymbol{b}^{\alpha,\beta} \in (L^\infty(\Omega))^d$, $\boldsymbol{c}^{\alpha,\beta} \in (L^\infty(\Omega))^d$ and $a_0^{\alpha,\beta} \in L^\infty(\Omega)$ are given functions. We can also write in matrix form

$$\mathcal{H}(\boldsymbol{u}) = \begin{pmatrix} \mathcal{L}_{\mathbb{A}^{1,1},\boldsymbol{b}^{1,1},\boldsymbol{c}^{1,1},a_0^{1,1}} & \cdots & \mathcal{L}_{\mathbb{A}^{1,m},\boldsymbol{b}^{1,m},\boldsymbol{c}^{1,m},a_0^{1,m}} \\ \vdots & \ddots & \vdots \\ \mathcal{L}_{\mathbb{A}^{m,1},\boldsymbol{b}^{m,1},\boldsymbol{c}^{m,1},a_0^{m,1}} & \cdots & \mathcal{L}_{\mathbb{A}^{m,m},\boldsymbol{b}^{m,m},\boldsymbol{c}^{m,m},a_0^{m,m}} \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_1 \\ \vdots \\ \boldsymbol{u}_m \end{pmatrix}. \tag{1.13}$$

We remark that the $\mathcal{H}$ operator for $m = 1$ is equivalent to the $\mathcal{L}$ operator.

For $\alpha \in [\![1, m]\!]$, we define Γ_α^D and Γ_α^R as open subsets of Γ, possibly empty, such that $\Gamma_\alpha^D \cap \Gamma_\alpha^R = \emptyset$. Let $\boldsymbol{f} \in (L^2(\Omega))^m$, $g_\alpha^D \in \mathrm{H}^{1/2}(\Gamma_\alpha^D)$, $g_\alpha^R \in L^2(\Gamma_\alpha^R)$, $a_\alpha^R \in L^\infty(\Gamma_\alpha^R)$ be given data.

A *vector* boundary value problem is given by

Vector BVP 1 : generic problem

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

$$\mathcal{H}(\boldsymbol{u}) = \boldsymbol{f} \quad \text{in } \Omega, \tag{1.14}$$

$$\boldsymbol{u}_\alpha = g_\alpha^D \quad \text{on } \Gamma_\alpha^D,\ \forall \alpha \in [\![1, m]\!], \tag{1.15}$$

$$\frac{\partial \boldsymbol{u}}{\partial n_{\mathcal{H}_\alpha}} + a_\alpha^R \boldsymbol{u}_\alpha = g_\alpha^R \quad \text{on } \Gamma_\alpha^R,\ \forall \alpha \in [\![1, m]\!], \tag{1.16}$$

where the α-th component of the **conormal derivative** of $\boldsymbol{u}$ is defined by

$$\frac{\partial \boldsymbol{u}}{\partial n_{\mathcal{H}_\alpha}} \stackrel{\mathsf{def}}{=} \sum_{\beta=1}^{m} \frac{\partial \boldsymbol{u}_\beta}{\partial n_{\mathcal{H}_{\alpha,\beta}}} = \sum_{\beta=1}^{m} \left(\left\langle \mathbb{A}^{\alpha,\beta}\, \boldsymbol{\nabla} \boldsymbol{u}_\beta, \boldsymbol{n} \right\rangle - \left\langle \boldsymbol{b}^{\alpha,\beta} \boldsymbol{u}_\beta, \boldsymbol{n} \right\rangle \right). \tag{1.17}$$

The boundary conditions (1.16) are the **Robin** boundary conditions and (1.15) is the **Dirichlet** boundary condition. The **Neumann** boundary conditions are particular Robin boundary conditions with $a_\alpha^R \equiv 0$.

In this problem, we may consider on a given boundary some conditions which can vary depending on the component. For example we may have a Robin boundary condition satisfying $\frac{\partial \boldsymbol{u}}{\partial n_{\mathcal{H}_1}} + a_1^R \boldsymbol{u}_1 = g_1^R$ and a Dirichlet one with $\boldsymbol{u}_2 = g_2^D$.

To have an outline of the FC-VFEM$\mathbb{P}_1$ package, a second and simple problem is quickly present.

Usual vector BVP 1 : 2D simple vector problem

Find $\boldsymbol{u} = (u_1, u_2) \in (\mathrm{H}^2(\Omega))^2$ such that

$$-\Delta u_1 + u_2 = 0 \quad \text{in } \Omega \subset \mathbb{R}^2, \tag{1.18}$$

$$-\Delta u_2 + u_1 = 0 \quad \text{in } \Omega \subset \mathbb{R}^2, \tag{1.19}$$

$$(u_1, u_2) = (0, 0) \quad \text{on } \Gamma_1, \tag{1.20}$$

$$(u_1, u_2) = (-12., +12.) \quad \text{on } \Gamma_{98}, \tag{1.21}$$

$$(u_1, u_2) = (+12., -12.) \quad \text{on } \Gamma_{99}, \tag{1.22}$$

where Ω and its boundaries are given in Figure 1.1.

The problem (1.18)-(1.22) can be equivalently expressed as the vector BVP (1.2)-(1.4) :

Vector BVP 2 : 2D simple vector problem

Find $\boldsymbol{u} = (u_1, u_2) \in (\mathrm{H}^2(\Omega))^2$ such that

$$\mathcal{H}(\boldsymbol{u}) = \boldsymbol{f} \quad \text{in } \Omega,$$

$$u_1 = g_1^D \quad \text{on } \Gamma^D = \Gamma_1 \cup \Gamma_{98} \cup \Gamma_{99},$$

$$u_2 = g_2^D \quad \text{on } \Gamma^D = \Gamma_1 \cup \Gamma_{98} \cup \Gamma_{99},$$

where

$$\mathcal{H} := \begin{pmatrix} \mathcal{L}_{\mathbb{I},\boldsymbol{O},\boldsymbol{O},0} & \mathcal{L}_{\mathbb{O},\boldsymbol{O},\boldsymbol{O},1} \\ \mathcal{L}_{\mathbb{O},\boldsymbol{O},\boldsymbol{O},1} & \mathcal{L}_{\mathbb{I},\boldsymbol{O},\boldsymbol{O},0} \end{pmatrix}, \text{ as } \mathcal{H}\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} -\Delta & 1 \\ 1 & -\Delta \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$$

$$f \equiv 0,$$

and

$$g_1^D = g_2^D := 0 \text{ on } \Gamma_1,\ g_1^D := -12,\ g_2^D := +12 \text{ on } \Gamma_{98},\ g_1^D := +12,\ g_2^D := -12 \text{ on } \Gamma_{99}$$

In Listing 21 a complete code is given to solve this problem. Numerical solutions are given in Figure 1.3.

```
meshfile=gmsh.buildmesh2d('condenser',10); # generate mesh
Th=siMesh(meshfile) # read mesh
Hop1=Loperator(dim=2,A=[[1,None],[None,1]])
```

```
Hop2=Loperator(dim=2,a0=1)
Hop=Hoperator(dim=2,m=2,H=[[Hop1,Hop2],[Hop2,Hop1]])
pde=PDE(Op=Hop)
bvp=BVP(Th,pde=pde)
bvp.setDirichlet( 1, 0,comps=[0,1])
bvp.setDirichlet( 98, [-12,+12],comps=[0,1]);
bvp.setDirichlet( 99, [+12,-12],comps=[0,1]);
U=bvp.solve(split=True)
# Graphic parts
plt.figure(1)
siplt.plot(Th,U[0])
plt.axis('off');set_axes_equal()
plt.colorbar(label='$u_1$',orientation='horizontal')
plt.figure(2)
siplt.plot(Th,U[1])
plt.axis('off');set_axes_equal()
plt.colorbar(label='$u_2$',orientation='horizontal')
```

Listing 1.2: Complete Python code to solve the funny 2D vector problem with graphical representations

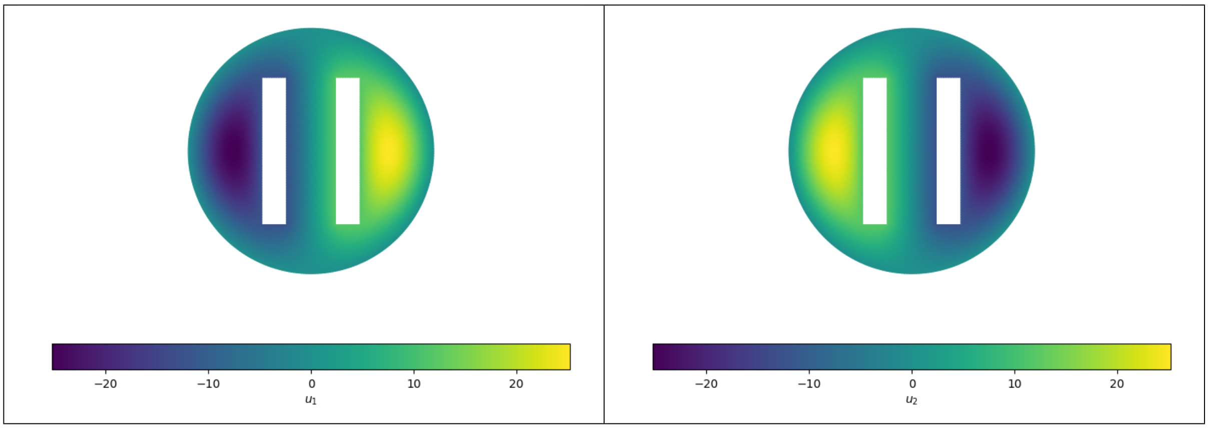


Figure 1.3: Funny vector BVP, u_1 numerical solution (left) and u_2 numerical solution (right)

Chapter 2

Notations and results on regular surfaces

All the notations and results of this chapter are directly obtain from [7].

2.1 Elementary geometric analysis

Definition 2.1: $\mathcal{C}^k$-hypersurface

Let $k \in \mathbb{N} \cup \infty$. $\Gamma \subset \mathbb{R}^{n+1}$ is called a $\mathcal{C}^k$-hypersurface if, for each point $x_0 \in \Gamma$, there exists an open set $U \in \mathbb{R}^{n+1}$ containing x_0 and a function $\Phi \in \mathcal{C}^k(U;\mathbb{R})$ with the property that $\nabla \Phi \neq 0$ on $\Gamma \cap U$ and such that

$$\Gamma \cap U = \{x \in U \mid \Phi(x) = 0\}. \tag{2.1}$$

The linear space

$$T_x\Gamma = [\nabla \Phi(x)]^{\perp} \tag{2.2}$$

is called the *tangent space* to Γ at $x \in \Gamma$.

A vector $\boldsymbol{\nu}(x) \in \mathbb{R}^{n+1}$ is called a *unit normal vector* at $x \in \Gamma$ if $\boldsymbol{\nu}(x) \perp T_x\Gamma$ and $\|\boldsymbol{\nu}(x)\| = 1$. We have

$$\boldsymbol{\nu}(x) = \frac{\nabla \Phi(x)}{\|\nabla \Phi(x)\|} \quad \text{or} \quad \boldsymbol{\nu}(x) = -\frac{\nabla \Phi(x)}{\|\nabla \Phi(x)\|} \tag{2.3}$$

A $\mathcal{C}^1$-hypersurface is called *orientable* if there exists a continuous vector field $\boldsymbol{\nu} : \Gamma \longrightarrow \mathbb{R}^{n+1}$ such that $\boldsymbol{\nu}(x)$ is a unit normal vector to Γ for all $x \in \Gamma$.

Definition 2.2: Tangential gradient or surface gradient

Let $\Gamma \subset \mathbb{R}^{n+1}$ be a $\mathcal{C}^1$-hypersurface and let $f : \Gamma \longrightarrow \mathbb{R}$ be differentiable at $x \in \Gamma$. we define the tangential gradient or surface gradient of f at $x \in \Gamma$ by

$$\nabla_\Gamma f(x) = \nabla \bar{f}(x) - \langle \nabla \bar{f}(x), \boldsymbol{\nu}(x)\rangle \boldsymbol{\nu}(x) = \mathbb{P}(x) \nabla \bar{f}(x), \tag{2.4}$$

where $(\mathbb{P}(x))_{i,j} = \delta_{i,j} - \boldsymbol{\nu}_i(x)\boldsymbol{\nu}_j x$, $\forall (i,j) \in [\![1, n+1]\!]^2$. Here $\bar{f}$ is a smooth extension of f to a neighbourhood $U \subset \mathbb{R}^{n+1}$ of the surface Γ, so that $\bar{f}_{|\Gamma} = f$.

We shall use the notation

$$\nabla_\Gamma f(x) = (\underline{D}_1 f(x), \ldots, \underline{D}_{n+1} f(x))^t \tag{2.5}$$

for the $n+1$ components of the surface gradient. Note that $\langle \nabla_\Gamma f(x), \boldsymbol{\nu}(x)\rangle = 0$ and hence $\nabla_\Gamma f(x) \in T_x\Gamma$.

We denote by :

- $\mathcal{C}^0(\Gamma;\mathbb{R})$ the set of functions $f : \Gamma \longrightarrow \mathbb{R}$ which are continous.
- $\mathcal{C}^1(\Gamma;\mathbb{R})$ the set of functions in $\mathcal{C}^0(\Gamma;\mathbb{R})$ which are differentiable at every point $x \in \Gamma$ and for which $\underline{D}_i \in \mathcal{C}^0(\Gamma;\mathbb{R})$, for all $i \in [\![1, n+1]\!]$.
- $\mathcal{C}^l(\Gamma;\mathbb{R})$, $l \in \mathbb{N}$ provide that Γ is a $\mathcal{C}^k$-hypersurface with $k \geqslant l$.

Let $\boldsymbol{g} : \Gamma \longrightarrow \mathbb{R}^{n+1}$ be differentiable at $x \in \Gamma$ ($\boldsymbol{g}$ is a vector field) and $f : \Gamma \longrightarrow \mathbb{R}$ be twice differentiable at $x \in \Gamma$. The surface divergence is given by

$$\operatorname{div}_\Gamma \boldsymbol{g}(x) = \sum_{i=1}^{n+1} \underline{D}_i \boldsymbol{g}_i(x) \tag{2.6}$$

and, the *Laplace-Beltrami operator* is given by

$$\Delta_\Gamma f(x) = \operatorname{div}_\Gamma \nabla_\Gamma f(x) = \sum_{i=1}^{n+1} \underline{D}_i \underline{D}_i f(x) \tag{2.7}$$

Definition 2.3: Extend Weingarten map

Let Γ be a $\mathcal{C}^2$-hypersurface. The extend Weingarten map is the matrix $\mathbb{H} \in \mathcal{M}_{n+1}(\mathcal{C}^0(\Gamma;\mathbb{R}))$ define by

$$\mathbb{H}_{i,j} = \underline{D}_i \boldsymbol{\nu}_j, \quad \forall (i,j) \in [\![1, n+1]\!]^2. \tag{2.8}$$

The matrix $\mathbb{H}$ is symmetric and $\mathbb{H}\boldsymbol{\nu} = 0$. The restriction of $\mathbb{H}$ to the tangent space is called the *Weingarten map.*

The *mean curvature* of Γ at point $x \in \Gamma$ is the quantity

$$H(x) = \operatorname{trace} \mathbb{H}(x) = \sum_{i=1}^{n+1} \mathbb{H}_{i,i}(x). \tag{2.9}$$

It differs from the common definition by a factor of n.

Lemme 2.4

Let Γ be a $\mathcal{C}^2$-hypersurface and $u \in \mathcal{C}^2(\Gamma;\mathbb{R})$. Then, we have

$$\underline{D}_i \underline{D}_j u - \underline{D}_j \underline{D}_i u = (\mathbb{H}\nabla_\Gamma u)_j \boldsymbol{\nu}_i - (\mathbb{H}\nabla_\Gamma u)_i \boldsymbol{\nu}_j, \quad \forall (i,j) \in [\![1, n+1]\!]^2. \tag{2.10}$$

Theorem 2.5: integration by parts on surfaces

Assume that Γ is a hypersurface in $\mathbb{R}^{n+1}$ with smooth boundary $\partial\Gamma$ and that $f \in \mathcal{C}^1(\overline{\Gamma};\mathbb{R})$. Then

$$\int_\Gamma \underline{D}_i f \mathrm{d}A = \int_\Gamma f H \boldsymbol{\nu}_i \mathrm{d}A + \int_{\partial\Gamma} f \boldsymbol{\mu}_i \mathrm{d}A, \quad \forall i \in [\![1, n+1]\!] \tag{2.11}$$

and so

$$\int_\Gamma \nabla_\Gamma f \mathrm{d}A = \int_\Gamma f H \boldsymbol{\nu} \mathrm{d}A + \int_{\partial\Gamma} f \boldsymbol{\mu} \mathrm{d}A. \tag{2.12}$$

Here, $\boldsymbol{\mu}$ denotes the co-normal vector which is normal to $\partial\Gamma$ and tangent to Γ.
A compact hypersurface Γ does not have boundary, $\partial\Gamma = \varnothing$, and the last term on the right-hand side vanishes.

Note that $\mathrm{d}A$ in connection with an integral over Γ denotes the n-dimensional surface measure, while $\mathrm{d}A$ in connection with an integral over $\partial\Gamma$ is the $n-1$-dimensional surface measure.

2.2 Sobolev space on surface

Let Γ be a $\mathcal{C}^2$-hypersurface for the following.

For $p \in [\![1, \infty]\!]$ we let $L^p(\Gamma)$ denote the space of functions $f : \Gamma \longrightarrow \mathbb{R}$ which are measurable with respect to the surface measure $\mathrm{d}A$ and have a finite norm, where the norm is defined by

$$\|f\|_{L^p(\Gamma)} = \left(\int_\Gamma |f|^p \mathrm{d}A\right)^{\frac{1}{p}}$$

for $p < \infty$, and for $p = \infty$ we mean the esential supremum norm.

$L^p(\Gamma)$ is a Banach space and for $p = 2$ a Hilbert space. For $1 \leqslant p < \infty$ the spaces $\mathcal{C}^0(\Gamma; \mathbb{R})$ and $\mathcal{C}^1(\Gamma; \mathbb{R})$ are dense in $L^p(\Gamma)$.

Definition 2.6

A function $f \in L^p(\Gamma)$ has the weak derivative $v_i = \underline{D}_i f \in L^p(\Gamma)$ if, for every function $\varphi \in \mathcal{C}^1(\Gamma; \mathbb{R})$ with compact support $\overline{\{x \in \Gamma \mid \varphi(x) \neq 0\}} \subset \Gamma$, we have the relation

$$\int_\Gamma f \underline{D}_i \varphi \mathrm{d}A = -\int_\Gamma f v_i \mathrm{d}A + \int_\Gamma f \varphi H \boldsymbol{\nu}_i \mathrm{d}A.$$

The Sobolev space $H^{1,p}(\Gamma)$ is defined by

$$H^{1,p}(\Gamma) = \{f \in L^p(\Gamma) \mid \underline{D}_i f \in L^p(\Gamma), i \in [\![1, n+1]\!]\} \tag{2.13}$$

with norm

$$\|f\|_{H^{1,p}(\Gamma)} = \left(\|f\|^p_{L^p(\Gamma)} + \|\boldsymbol{\nabla}_\Gamma f\|^p_{L^p(\Gamma)}\right)^{\frac{1}{p}} \tag{2.14}$$

Theorem 2.7: Poincaré's inequality

Assume that Γ is a $\mathcal{C}^3$-hypersurface and $1 \leqslant p < \infty$. Then there is a constant $C > 0$ such that, for every function $f \in H^{1,p}(\Gamma)$ with $\int_\Gamma d\mathrm{d}A = 0$, we have the inequality

$$\|f\|_{L^p(\Gamma)} \leqslant C \|\boldsymbol{\nabla}_\Gamma f\|_{L^p(\Gamma)}. \tag{2.15}$$

Theorem 2.8: Green's formula

Let Γ be a $\mathcal{C}^2$-hypersurface. Then for all $f \in H^{1,p}(\Gamma)$ and $g \in H^{2,p}(\Gamma)$ we have

$$\int_\Gamma \langle \boldsymbol{\nabla}_\Gamma f, \boldsymbol{\nabla}_\Gamma g\rangle \mathrm{d}A = -\int_\Gamma f \Delta_\Gamma g \mathrm{d}A + \int_{\partial\Gamma} f \langle \boldsymbol{\nabla}_\Gamma g, \boldsymbol{\mu}\rangle \mathrm{d}A. \tag{2.16}$$

A compact hypersurface Γ does not have boundary, $\partial\Gamma = \varnothing$, and the last term on the right-hand side vanishes.

Proof. Let $i \in [\![1, n]\!]$, we have

$$\underline{D}_i \left(f \underline{D}_i(g)\right) = \underline{D}_i(f)\underline{D}_i(g) + f \underline{D}_i \underline{D}_i(g)$$

and then

$$\int_\Gamma \underline{D}_i(f)\underline{D}_i(g)\mathrm{d}A = \int_\Gamma \underline{D}_i \left(f \underline{D}_i(g)\right)\mathrm{d}A - \int_\Gamma f \underline{D}_i \underline{D}_i(g)\mathrm{d}A$$

Using integration by parts formula (2.11) with f identify to $f\underline{D}_i(g)$ we obtain

$$\int_\Gamma \underline{D}_i(f)\underline{D}_i(g)\mathrm{d}A = \int_\Gamma f\underline{D}_i(g)H\boldsymbol{\nu}_i\mathrm{d}A + \int_{\partial\Gamma} f\underline{D}_i(g)\boldsymbol{\mu}_i\mathrm{d}A - \int_\Gamma f\underline{D}_i\underline{D}_i(g)\mathrm{d}A$$

By summing in i, we obtain

$$\int_\Gamma \langle \boldsymbol{\nabla}_\Gamma f, \boldsymbol{\nabla}_\Gamma g\rangle \mathrm{d}A = \int_\Gamma f\langle \boldsymbol{\nabla}_\Gamma g, \boldsymbol{\nu}\rangle H\mathrm{d}A + \int_{\partial\Gamma} f\langle \boldsymbol{\nabla}_\Gamma g, \boldsymbol{\mu}\rangle \mathrm{d}A - \int_\Gamma f\Delta_\Gamma g\mathrm{d}A$$

Since $\langle \boldsymbol{\nabla}_\Gamma g, \boldsymbol{\nu}\rangle = 0$, the formula is prooved. □

2.3 Variational formulation of a surface BVP

In this section, the variational formulation of the generic *scalar* surface BVP (1.2)-(1.4) is established by using previous results.

Lemme 2.9

Let Γ be a $\mathcal{C}^2$-hypersurface. Then, for all $f, g \in H^1(\Gamma)$ and for all $i \in [\![1, n+1]\!]$, we have

$$\int_\Gamma \underline{D}_i(f)g\mathrm{d}A = \int_\Gamma fgH\boldsymbol{\nu}_i\mathrm{d}A - \int_\Gamma f\underline{D}_i(g)\mathrm{d}A + \int_{\partial\Gamma} fg\boldsymbol{\mu}_i\mathrm{d}A. \tag{2.17}$$

Proof. We have $\underline{D}_i(f)g = \underline{D}_i(fg) - f\underline{D}_i(g)$ and by integration we obtain using ...

$$\begin{aligned}\int_\Gamma \underline{D}_i(f)g\mathrm{d}A &= \int_\Gamma \underline{D}_i(fg)\mathrm{d}A - \int_\Gamma f\underline{D}_i(g)\mathrm{d}A \\ &= \int_\Gamma fgH\boldsymbol{\nu}_i\mathrm{d}A + \int_{\partial\Gamma} fg\boldsymbol{\mu}_i\mathrm{d}A - \int_\Gamma f\underline{D}_i(g)\mathrm{d}A\end{aligned}$$

□

Lemme 2.10

Let Γ be a $\mathcal{C}^2$-hypersurface. Then, for all $u \in H^2(\Gamma)$ and $v \in H^1(\Gamma)$, we have

$$\int_\Gamma \mathrm{div}_\Gamma(\mathbb{A}\boldsymbol{\nabla}_\Gamma u)v\mathrm{d}A = -\int_\Gamma \langle \mathbb{A}\boldsymbol{\nabla}_\Gamma u, \boldsymbol{\nabla}_\Gamma v\rangle \mathrm{d}A + \int_{\partial\Gamma} \langle \mathbb{A}\boldsymbol{\nabla}_\Gamma u, \boldsymbol{\mu}\rangle \mathrm{d}A. \tag{2.18}$$

Proof. Using (2.17) with $f = A_{i,j}\underline{D}_j(u)$ and $g = v$ we obtain

$$\begin{aligned}\int_\Gamma \underline{D}_i(A_{i,j}\underline{D}_j(u))v\mathrm{d}A &= \int_\Gamma A_{i,j}\underline{D}_j(u)vH\boldsymbol{\nu}_i\mathrm{d}A - \int_\Gamma A_{i,j}\underline{D}_j(u)\underline{D}_i(v)\mathrm{d}A \\ &+ \int_{\partial\Gamma} A_{i,j}\underline{D}_j(u)v\boldsymbol{\mu}_i\mathrm{d}A.\end{aligned}$$

Summation in i and j give

$$\begin{aligned}\int_\Gamma \mathrm{div}_\Gamma(\mathbb{A}\boldsymbol{\nabla}_\Gamma u)v\mathrm{d}A &= \int_\Gamma \langle \mathbb{A}\boldsymbol{\nabla}_\Gamma u, \boldsymbol{\nu}\rangle Hv\mathrm{d}A - \int_\Gamma \langle \mathbb{A}\boldsymbol{\nabla}_\Gamma u, \boldsymbol{\nabla}_\Gamma v\rangle \mathrm{d}A \\ &+ \int_{\partial\Gamma} \langle \mathbb{A}\boldsymbol{\nabla}_\Gamma u, \boldsymbol{\mu}\rangle \mathrm{d}A.\end{aligned}$$

By hypothesis on $\mathbb{A}$, $\mathbb{A}(x)\boldsymbol{\nabla}_\Gamma u(x) \in T_x\Gamma$ and so $\langle \mathbb{A}(x)\boldsymbol{\nabla}_\Gamma u(x), \boldsymbol{\nu}(x)\rangle = 0$. □

Lemme 2.11

Let Γ be a $\mathcal{C}^2$-hypersurface. Then, for all $u, v \in H^1(\Gamma)$, we have

$$\int_\Gamma \operatorname{div}_\Gamma(\boldsymbol{b}u) v \mathrm{d}A = -\int_\Gamma \langle \boldsymbol{b}u, \boldsymbol{\nabla}_\Gamma v \rangle \mathrm{d}A + \int_{\partial\Gamma} \langle \boldsymbol{b}u, \boldsymbol{\mu} \rangle v \mathrm{d}A. \tag{2.19}$$

Proof. Using (2.17) with $f = b_i u$ and $g = v$ we obtain

$$\int_\Gamma \underline{D}_i(b_i u) v \mathrm{d}A = \int_\Gamma b_i u v H \boldsymbol{\nu}_i \mathrm{d}A - \int_\Gamma b_i u \underline{D}_i(v) \mathrm{d}A + \int_{\partial\Gamma} b_i u v \boldsymbol{\mu}_i \mathrm{d}A.$$

Summation in i give

$$\begin{aligned} \int_\Gamma \operatorname{div}_\Gamma(\boldsymbol{b}u) v \mathrm{d}A &= \int_\Gamma \langle \boldsymbol{b}u, \boldsymbol{\nu} \rangle H v \mathrm{d}A - \int_\Gamma \langle \boldsymbol{b}u, \boldsymbol{\nabla}_\Gamma v \rangle \mathrm{d}A \\ &+ \int_{\partial\Gamma} \langle \boldsymbol{b}u, \boldsymbol{\mu} \rangle v \mathrm{d}A. \end{aligned}$$

By hypothesis, for all $x \in \Gamma$, $\boldsymbol{b}(x) \in T_x\Gamma$ and so $\langle \boldsymbol{b}(x)u(x), \boldsymbol{\nu}(x) \rangle = u(x) \langle \boldsymbol{b}(x), \boldsymbol{\nu}(x) \rangle = 0$. □

So we have prooved the following theorem

Theorem 2.12

Let $\Gamma \subset \mathbb{R}^{n+1}$ be a $\mathcal{C}^2$-hypersurface. Let a_0, b_i, c_i, $A_{i,j} \in L^\infty(\Gamma)$, $\forall (i,j) = [\![1, n+1]\!]^2$. We also assume that for all $x \in \Gamma$, $\boldsymbol{b}(x) \in T_x\Gamma$ and the matrix $\mathbb{A}(x) = (A_{i,j}(x))_{i,j=1,\dots,n+1}$ maps the tangent space $T_x\Gamma$ into itself (i.e. $\mathbb{A}(x) : T_x\Gamma \longrightarrow T_x\Gamma$). We denote by $\mathcal{L}$ the second order linear differential operator acting on scalar fields defined by

$$\mathcal{L}(u) \stackrel{\mathsf{def}}{=} -\operatorname{div}_\Gamma(\mathbb{A} \boldsymbol{\nabla}_\Gamma u) + \operatorname{div}_\Gamma(\boldsymbol{b}u) + \langle \boldsymbol{\nabla}_\Gamma u, \boldsymbol{c} \rangle + a_0 u, \quad \forall u \in H^2(\Gamma).$$

Then, for all $u \in H^2(\Gamma)$ and $v \in H^1(\Gamma)$, we have

$$\begin{aligned} \int_\Gamma \mathcal{L}(u) v \mathrm{d}A &= \int_\Gamma \langle \mathbb{A} \boldsymbol{\nabla}_\Gamma u, \boldsymbol{\nabla}_\Gamma v \rangle \mathrm{d}A + \int_\Gamma a_0 u v \mathrm{d}A \\ &- \left(\int_\Gamma u \langle \boldsymbol{\nabla}_\Gamma v, \boldsymbol{b} \rangle \mathrm{d}A - \int_\Gamma \langle \boldsymbol{\nabla}_\Gamma u, \boldsymbol{c} \rangle v \mathrm{d}A \right) \\ &- \int_{\partial\Gamma} \langle \mathbb{A} \boldsymbol{\nabla}_\Gamma u, \boldsymbol{\mu} \rangle v \mathrm{d}A + \int_{\partial\Gamma} \langle \boldsymbol{b}u, \boldsymbol{\mu} \rangle v \mathrm{d}A. \end{aligned}$$

Let $\mathcal{D}_\mathcal{L} = \mathcal{D}_{\mathbb{A},\boldsymbol{b},\boldsymbol{c},a_0}$ be the first order bilinear differential operator acting on *scalar fields* associated to the $\mathcal{L}$ operator defined $\forall (u,v) \in (H^1(\Gamma))^2$ by

$$\mathcal{D}_\mathcal{L}(u,v) = \langle \mathbb{A} \boldsymbol{\nabla}_\Gamma u, \boldsymbol{\nabla}_\Gamma v \rangle - (\langle \boldsymbol{\nabla}_\Gamma v, \boldsymbol{b} \rangle u - \langle \boldsymbol{\nabla}_\Gamma u, \boldsymbol{c} \rangle v) + a_0 u v. \tag{2.20}$$

The variational formulation associated to the scalar surface BVP (1.2)-(1.4) is given by

Variational formulation of the scalar surface BVP (1.2)-(1.4)

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

$$\mathcal{A}_\mathcal{L}(u,v) = \mathcal{F}(v) \quad \forall v \in H^1_{0,\partial\Gamma^D}(\Gamma) \tag{2.21}$$

where

$$\mathcal{A}_\mathcal{L}(u,v) = \int_\Gamma \mathcal{D}_\mathcal{L}(u,v) \mathrm{d}A + \int_{\partial\Gamma^R} a^R u v \mathrm{d}A \tag{2.22}$$

$$\mathcal{F}(v) = \int_\Gamma f v \mathrm{d}A + \int_{\partial\Gamma^R} g^R v \mathrm{d}A \tag{2.23}$$

Chapter 3

scalar surface BVP

This next section looks first at the construction of surface BVPs where analytic solutions are known by using Sage (a formal calculation software). Thereafter, some surface BVPs are numerically solved by using the FC-VFEM$\mathbb{P}_1$ Python package and solutions compared to exact solutions when we know them.

3.1 Analytical solutions of surface BVPs

3.1.1 Sage computation of Laplace-Beltrami operator on hypersurfaces

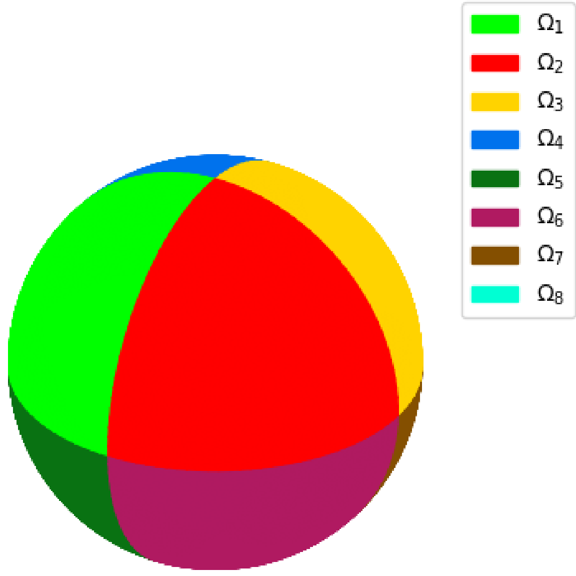
Let the hypersurface Γ given by a level set function $\Phi : \mathbb{R}^3 \longrightarrow \mathbb{R}$:

$$\Gamma = \left\{ \boldsymbol{x} \in \mathbb{R}^3 \mid \Phi(\boldsymbol{x}) = 0. \right\}$$

For the unit sphere, one can use

$$\Phi : \boldsymbol{x} \mapsto x_1^2 + x_2^2 + x_3^3 - 1.$$

With the FC-OOGMSH and FC-SIMESH packages, one can easily generated this surface from a sphere mesh made by `gmsh` and `sphere4surf.geo` :




```
geofile='sphere4surf'
(geodir,geofile)=get_geo(3,2,geofile)
meshfile=gmsh.buildmesh3ds(geodir+os.sep+geofile+'.geo',20,
    force=False,verbose=0)
Th=siMesh(meshfile)
```

Listing 3.1: Mesh of a sphere with 8 subdomains, $\Gamma = \Omega_1 \cup \cdots \cup \Omega_8$.

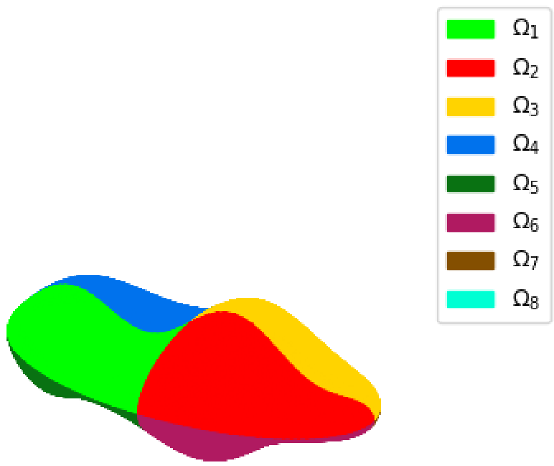
More complicated surfaces can be obtained by mapping the unit sphere S^2 onto Γ. we note $F : S^2 \longrightarrow \Gamma$ the map function. From example 4.8 in [7], we take

$$F(y_1, y_2, y_3) = \left(2y_1, y_2, \frac{1}{2}y_3\left(1 + \frac{1}{2}\sin(2\pi y_1)\right)\right) \in \Gamma, \quad \forall \boldsymbol{y} = (y_1, y_2, y_3) \in S^2 \tag{3.1}$$

The representation of $\Gamma = \boldsymbol{F}(S^2)$ as a hypersurface $\{\boldsymbol{x} \in \mathbb{R}^3 \mid \phi(\boldsymbol{x}) = 0\}$ follows from $y_1^2 + y_2^2 + y_3^3 = 1$ with the level set function

$$\phi(\boldsymbol{x}) = \frac{1}{4}x_1^2 + x_2^2 + \frac{4x_3^2}{(1 + \frac{1}{2}\sin(\pi x_1))^2} - 1. \tag{3.2}$$

With the FC-OOGMSH and FC-SIMESH packages, one can easily generated this surface from a mesh of the sphere made by `gmsh` and `sphere4surf.geo` :




```
geofile='sphere4surf'
(geodir,geofile)=get_geo(3,2,geofile)
meshfile=gmsh.buildmesh3ds(geodir+os.sep+geofile+'.geo',20,
    force=False,verbose=0)
from numpy import sin,pi
map=lambda x,y,z : [2*x,y,0.5*z*(1+0.5*sin(2*pi*x))]
Th=siMesh(meshfile,mapping=map)
```

Listing 3.2: Mesh of a sphere with 8 subdomains transform with mapping function F, $\Gamma = \Omega_1 \cup \cdots \cup \Omega_8$.

To compute the Laplace-Beltrami operator apply on a given function over a given surface we use the Sage software (see [6]) by implementing :

- the surface gradient operator $\boldsymbol{\nabla}_\Gamma$, formula (2.4),
- the surface divergence operator $\operatorname{div}_\Gamma$, formula (2.6),
- the Laplace-Beltrami operator Δ_Γ, formula (2.7).

The source code is given in Listing 3.3.

Listing 3.3: **Implementation of Laplace-Beltrami operator, Sage code**

```
# gradient of function Phi
def Gradient(Phi):
  return Phi.diff()
# normal to the surface function Phi
def normal(Phi):
  G=Gradient(Phi)
  return G/G.norm()
# Projection matrix on tangent space
def Pmat(normal):
  P=[];dim=len(normal)
  for i in range(dim):
    Pr=[];
    for j in range(dim):
      if i==j:
        Pr.append(1-normal[i]*normal[j])
      else:
        Pr.append(-normal[i]*normal[j])
    P.append(Pr)
  return matrix(P)
# Surface gradient of u over surface function Phi
def SurfGrad(u,Phi):
  P=Pmat(normal(Phi))
  gradu=Gradient(u)
  return P*gradu
# Surface divergence of u over surface function Phi
def SurfDiv(V,Phi):
  P=Pmat(normal(Phi))
  f=0
  for i in range(len(V)):
    g=P*Gradient(V[i])
    f=f+g[i]
  return f.simplify_full()
# Laplace-Beltrami operator on u over surface function Phi
def LaplaceBeltrami(u,Phi):
  return SurfDiv(SurfGrad(u,Phi),Phi).simplify_full()
```

Under Sage, one can now compute some exact solutions on hypersurfaces :

- sample 1 : $\Gamma = S^2$

```
load('../sage/LaplaceBeltrami.sage')
var('x,y,z',domain=RR)
Phi(x,y,z)=x**2+y**2+z**2 -1
u(x,y,z)=x*y
f(x,y,z)=-LaplaceBeltrami(u,Phi)
```

  So with $\Phi : (x, y, z) \mapsto x^2 + y^2 + z^2 - 1$ and $u : (x, y, z) \mapsto xy$ we have

$$f = -\Delta_\Gamma u : (x, y, z) \mapsto \frac{6\,xy}{x^2 + y^2 + z^2}$$

- sample 2 : $\Gamma = S^2$

```
load('../sage/LaplaceBeltrami.sage')
var('x,y,z',domain=RR)
Phi(x,y,z)=x**2+y**2+z**2 -1
u(x,y,z)=x**2*y**2
f(x,y,z)=-LaplaceBeltrami(u,Phi)
```

  So with $\Phi : (x, y, z) \mapsto x^2 + y^2 + z^2 - 1$ and $u : (x, y, z) \mapsto x^2y^2$ we have

$$f = -\Delta_\Gamma u : (x, y, z) \mapsto -\frac{2\left(x^4 - 8\,x^2y^2 + y^4 + \left(x^2 + y^2\right)z^2\right)}{x^2 + y^2 + z^2}$$

- sample 3 : $\Gamma = F(S^2)$ with F given by (3.1) and the level set function Φ defined in (3.2)

```
load('../sage/LaplaceBeltrami.sage')
var('x,y,z',domain=RR)
u(x,y,z)=x*y
Phi(x,y,z)=x**2/4+y**2+(4*z**2)/(1+sin(pi*x)/2)**2 -1
f(x,y,z)=-LaplaceBeltrami(u,Phi)
```

In this case, the function $f = -\Delta_\Gamma u$ obtained is to long to be written.

3.2 Numerical samples on closed hypersurfaces

3.2.1 $-\Delta_\Gamma u + a_0 u = f$ with exact solution on S^2

We want to solve the surface PDE on a closed hypersurface Γ :

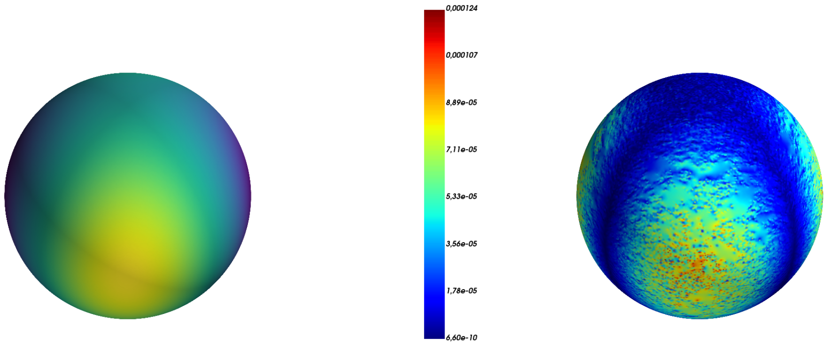
$$-\Delta_\Gamma u + a_0 u = f, \quad \text{on } \Gamma \tag{3.3}$$

where $f \in L^2(\Gamma)$ and $a_0 \in L^\infty(\Gamma)$ are given with $a_0 > 0$. With this last assumption we have existence and unicity of a $u \in \mathrm{H}^2(\Gamma)$.

To solve and compare to exact solution, we choose $u(\boldsymbol{x}) = x_1 x_2$, $a_0(\boldsymbol{x}) = 1 + x_1^2$ and we calculate the right-hand side f as $f = -\Delta_\Gamma u + a_0 u$ using sage with $\Gamma = S^2$. The complete code using the FC-VFEM$\mathbb{P}_1$ Python package is given in Listing 3.4. In figure 3.1

One can also used the `run` function from the module `fc_vfemp1_surface.examples.sBVP_sphere` :

```
from fc_vfemp1_surface.examples import sBVPsamples
from fc_vfemp1_surface.examples.sBVP_sphere import run
u=sBVPsamples.uxyz01
f=sBVPsamples.fxyz01
Fmap=sBVPsamples.Fmapxyz01
a0=lambda x,y,z: 1+x**2
run(N=75,u=u, f=f,Fmap=Fmap, a0=a0)
```



```
print('***␣Building/reading␣mesh')
N=75;geofile='sphere4surf';
(geodir,geofile)=get_geo(3,2,geofile)
meshfile=gmsh.buildmesh3ds(geodir+os.sep+geofile+'.geo',N,
    force=False,verbose=0)
Th=siMesh(meshfile)
print('***␣Setting␣surface␣BVP')
u=lambda x,y,z: x*y
LPu=lambda x,y,z: 6*x*y/(x**2 + y**2 + z**2)
a0=lambda x,y,z: 1+x**2
f=lambda x,y,z: LPu(x,y,z)+a0(x,y,z)*u(x,y,z)
Lop=Loperator(dim=3,d=2,A=[[1,None,None],[None,1,None],[None,None,1]],a0=a0)
pde=PDE(Op=Lop,f=f)
bvp=BVP(Th,pde=pde)
print('***␣Solving␣surface␣BVP')
U=bvp.solve()
```

Listing 3.4: Solving $-\Delta_\Gamma u + a_0 u = f$ on unit sphere with $a_0(x, y, z) = 1 + x^2$, solution (left) and relative error (right)

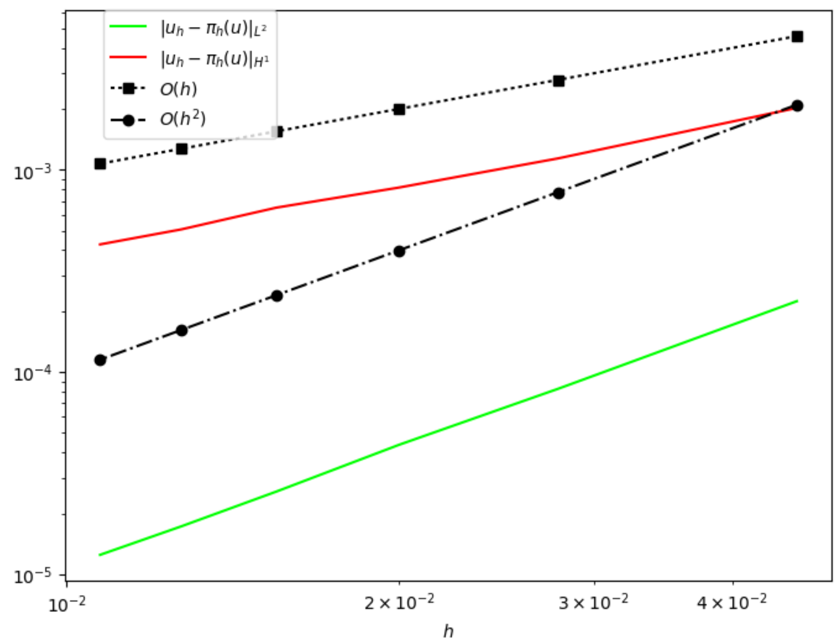
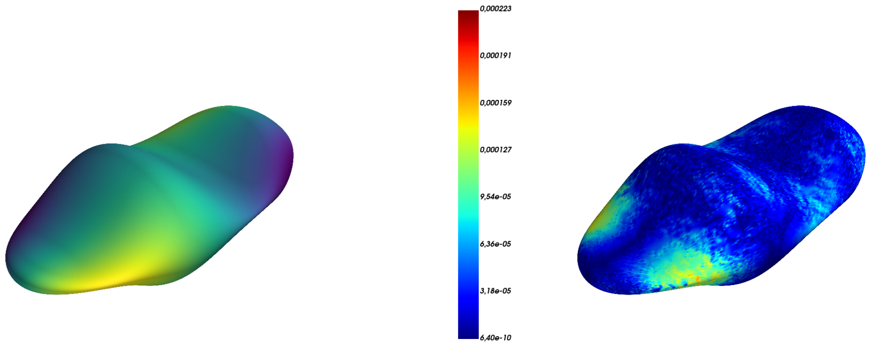


Figure 3.1: Order for surface BVP $-\Delta_\Gamma u + a_0 u = f$ on unit sphere with $a_0(x, y, z) = 1 + x^2$ and exact solution $u(x, y, z) = xy$.

3.2.2 $-\Delta_\Gamma u + a_0 u = f$ on $F(S^2)$ with exact solution

To solve and compare to exact solution, we choose $u(\boldsymbol{x}) = x_1 x_2$, $a_0(\boldsymbol{x}) = 1 + x_1^2$ and we calculate with Sage the right-hand side f as $f = -\Delta_\Gamma u + a_0 u$ where $\Gamma = F(S^2)$ and F given by (3.1).

The complete code using the FC-VFEM$\mathbb{P}_1$ Python package is given in Listing 3.5. In figure **??** the orders for the L^2 and H^1 norms are represented.




```
from fc_vfemp1_surface.examples import sBVPsamples
from fc_vfemp1_surface.examples.sBVP_sphere import run
u=sBVPsamples.uxyz03
f=sBVPsamples.fxyz03
Fmap=sBVPsamples.Fmapxyz03
a0=lambda x,y,z: 1+x**2
res=run(N=75,u=u,f=f,Fmap=Fmap,a0=a0)
```

Listing 3.5: Solving $-\Delta_\Gamma u + a_0 u = f$ on $F(S^2)$ with $a_0(x, y, z) = 1 + x^2$, solution (left) and relative error (right)

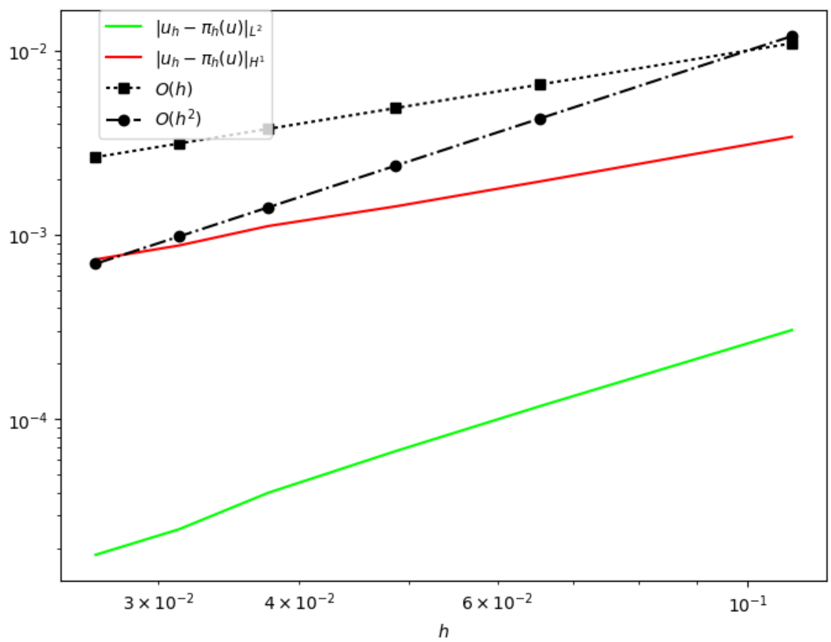


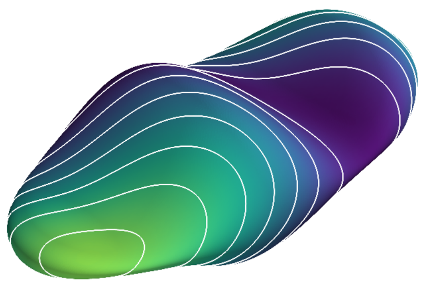
Figure 3.2: Order for surface BVP $-\Delta_\Gamma u + a_0 u = f$ on unit sphere mapping by function (3.1) : $u(x, y, z) = xy$.

3.2.3 $-\Delta_\Gamma u + a_0 u = f$ on $F(S^2)$ without explicit solution

We want to solve the surface PDE

$$-\Delta_\Gamma u + a_0 u = f, \quad \text{on } \Gamma = F(S^2) \tag{3.4}$$

with $a_0(x,y,z) = 1 + 0.9\cos(x+y+z)$, $f = 1$



```python
from numpy import cos,sin,pi
a0=lambda x,y,z: 1+0.9*cos(x+y+z)
f=1
Fmap=lambda x,y,z: [2*x, y,0.5*z*(0.5*sin(2*pi*x) + 1)]
print('***␣Building/reading␣mesh')
N=75;geofile='sphere4surf';
(geodir,geofile)=get_geo(3,2,geofile)
meshfile=gmsh.buildmesh3ds(geodir+os.sep+geofile+'.geo',N,
      force=False,verbose=0)
Th=siMesh(meshfile,mapping=Fmap)
print('***␣Setting␣surface␣BVP')
Lop=Loperator(dim=3,d=2,A=[[1,None,None],[None,1,None],[None,None,1]],a0=a0)
pde=PDE(Op=Lop,f=f)
bvp=BVP(Th,pde=pde)
print('***␣Solving␣surface␣BVP')
U=bvp.solve()
```

Listing 3.6: Solution of $-\Delta_\Gamma u + +a_0 u = f$ PDE on unit sphere mapping by function (3.1) with $a_0(x,y,z) = 1 + 0.9\cos(x+y+z)$ and $f = 1$. Python code

3.2.4 $-\Delta_\Gamma u + \langle \nabla_\Gamma u, \boldsymbol{c} \rangle + a_0 u = f$ on $F(S^2)$ with exact solution

We want to solve the surface PDE on a closed hypersurface Γ :

$$-\Delta_\Gamma u + \langle \nabla_\Gamma u, \boldsymbol{c} \rangle + a_0 u = f, \quad \text{on } \Gamma \tag{3.5}$$

where $f \in L^2(\Gamma)$, $\boldsymbol{c} \in (L^\infty(\Gamma))^3$ and $a_0 \in L^\infty(\Gamma)$ are given with $a_0 > 0$.

For this example we choose Γ as the surface obtained by mapping the unit sphere by the function defined in (3.1)

To solve and compare to exact solution, we choose

$$u(\boldsymbol{x}) = x_1 x_2, \quad a_0(\boldsymbol{x}) = 1 + x_1^2 \quad \text{and} \quad \boldsymbol{c}(\boldsymbol{x}) = (\cos(x_1), \sin(x_2), 2 + x_1 x_2 x_3)^t$$

With Sage, we compute $-\Delta_\Gamma u$ and $\nabla_\Gamma u$. The results are saved in module `fc_vfemp1_surface.examples.sBVPsamples` as functions respectively obtain by

```
from fc_vfemp1_surface.examples.sBVPsamples import fxyz04
from fc_vfemp1_surface.examples.sBVPsamples import gradSuxyz04
```

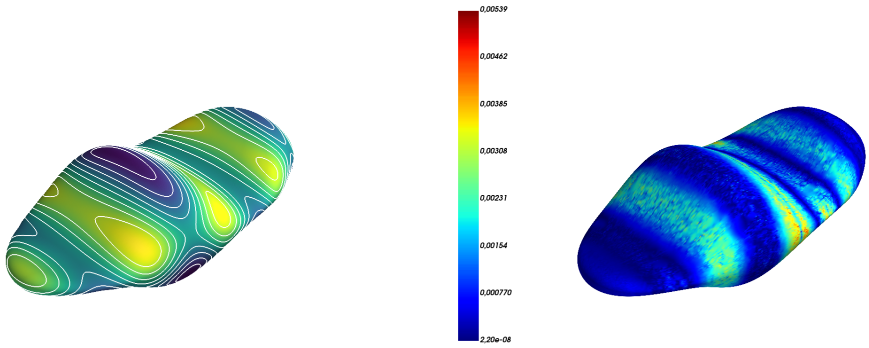
as well as corresponding u and map function (3.1) respectively given by

```
from fc_vfemp1_surface.examples.sBVPsamples import uxyz04
from fc_vfemp1_surface.examples.sBVPsamples import Fmapxyz04
```

We give in Listing **??** the complet Matlab code to solve 3.5 with

$$a_0 : \boldsymbol{x} \mapsto 1 + x_1^2 \quad \text{and} \quad \boldsymbol{c} : \boldsymbol{x} \mapsto \begin{pmatrix} \cos(x_1) \\ \sin(x_2) \\ 2 + x_1 x_2 x_3 \end{pmatrix}.$$

The order of the P_1-lagrange finite element method used in the FC-VFEM$\mathbb{P}_1$ toolbox is given in Figure 3.2.




```python
from numpy import cos, sin, pi
from fc_vfemp1_surface.examples import sBVPsamples
from fc_vfemp1_surface.examples.sBVP_sphere import run
u=sBVPsamples.uxyz04
f=sBVPsamples.fxyz04
Fmap=sBVPsamples.Fmapxyz04
gradSu=sBVPsamples.gradSuxyz04
a0=lambda x,y,z: 1+x**2
c=[lambda x1,x2,x3: cos(x1), lambda x1,x2,x3: sin(x2), lambda x1,x2,x3:
    2+x1*x2*x3]
cgradSu=lambda x,y,z: np.sum(np.array(gradSu(x,y,z))*np.array([c[0](x,y,z),
    c[1](x,y,z),c[2](x,y,z)]),axis=0)
# RHS function :
RHSfun=lambda x,y,z: f(x,y,z)+cgradSu(x,y,z)+a0(x,y,z)*u(x,y,z)
print('*** Building/reading mesh')
N=75;geofile='sphere4surf';
(geodir,geofile)=get_geo(3,2,geofile)
meshfile=gmsh.buildmesh3ds(geodir+os.sep+geofile+'.geo',N,
    force=False,verbose=0)
Th=siMesh(meshfile,mapping=Fmap)
print('*** Setting surface BVP')
Lop=Loperator(dim=3,d=2,A=[[1,None,None],[None,1,None],[None,None,1]],
    c=c,a0=a0)
pde=PDE(Op=Lop,f=RHSfun)
bvp=BVP(Th,pde=pde)
print('*** Solving surface BVP')
U=bvp.solve()
```

Listing 3.7: Solution of surface BVP $-\Delta_\Gamma u + \langle \nabla_\Gamma u, \boldsymbol{c}\rangle + a_0 u = f$ on unit sphere mapping by function (3.1). Python code

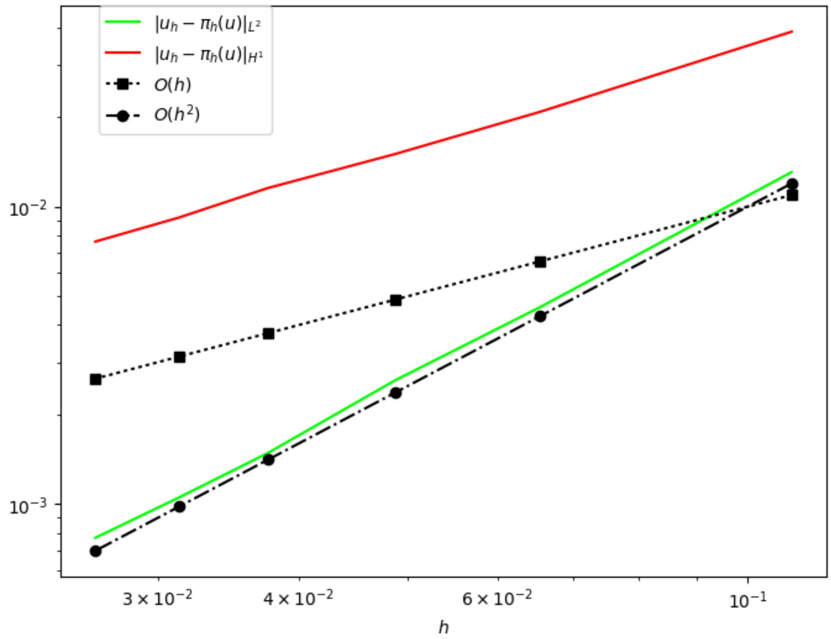
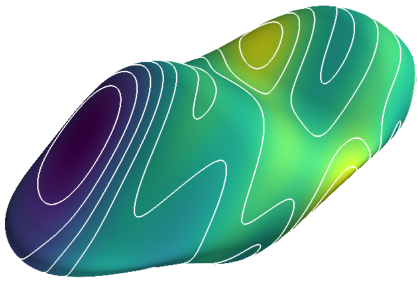


Figure 3.3: Order for the surface BVP $-\Delta_\Gamma u + \langle \nabla_\Gamma u, \boldsymbol{c}\rangle + a_0 u = f$ on unit sphere mapping by function (3.1).

3.2.5 $-\Delta_\Gamma u + \langle \nabla_\Gamma u, \boldsymbol{c}\rangle + a_0 u = f$ on $F(S^2)$ without exact solution

From previous sample, just differs the f function given by

$$f(\boldsymbol{x}) = 5\cos(5x_1 - 5x_2 + 7x_3)$$



```
from numpy import cos, sin, pi
a0=lambda x,y,z: 1+0.9*cos(x+y+z)
c=[lambda x1,x2,x3: cos(x1), lambda x1,x2,x3: sin(x2), lambda x1,x2,x3:
    2+x1*x2*x3]
f=lambda x1,x2,x3: 5*cos(5*x1-5*x2+7*x3)
Fmap=lambda x,y,z: [2*x, y,0.5*z*(0.5*sin(2*pi*x) + 1)]
print('***␣Building/reading␣mesh')
N=75;geofile='sphere4surf';
(geodir,geofile)=get_geo(3,2,geofile)
meshfile=gmsh.buildmesh3ds(geodir+os.sep+geofile+'.geo',N,
    force=False,verbose=0)
Th=siMesh(meshfile,mapping=Fmap)
print('***␣Setting␣surface␣BVP')
Lop=Loperator(dim=3,d=2,A=[[1,None,None],[None,1,None],[None,None,1]],
    c=c,a0=a0)
pde=PDE(Op=Lop,f=f)
bvp=BVP(Th,pde=pde)
print('***␣Solving␣surface␣BVP')
U=bvp.solve()
```

Listing 3.8: Solution of surface PDE $-\Delta_\Gamma u + \langle \boldsymbol{\nabla}_\Gamma u, \boldsymbol{c}\rangle + a_0 u = f$ on unit sphere mapping by function (3.1) without exact solution. Python code

3.2.6 3D surface Laplace-Beltrami BVP on closed hypersurface

On closed and regular hypersurface there is no boundary. The surface Laplace-Beltrami BVP is the following

Scalar surface BVP 2 : Laplace-Beltrami

Let Γ be a regular and closed hypersurface. Find $u \in \mathrm{H}^1(\Gamma)$ such that

$$-\Delta_\Gamma u = f \text{ in } \Gamma \tag{3.6}$$

$$\int_\Gamma u d\mathrm{q} = 0. \tag{3.7}$$

From [7], (Therorem 3.3, page 302 and Therorem 4.9, page 319) we obtain

Theorem 3.1

Suppose that $f \in L^2(\Gamma)$ with $\int_\Gamma f d\mathrm{q} = 0$. There exists a unique solution $u \in \mathrm{H}^2(\Gamma)$ of the Laplace-Beltrami surface BVP (3.6)-(3.7).

Let u_h be the P_1-Lagrange FEM solution of the discrete Laplace-Beltrami surface BVP then

$$\|u - u_h\|_{\mathrm{H}^1(\Gamma_h)} \leqslant C_1 h \quad \text{and} \quad \|u - u_h\|_{L^2(\Gamma_h)} \leqslant C_0 h^2. \tag{3.8}$$

The linear system $\mathbb{A}\boldsymbol{u}_h = \boldsymbol{b}$ obtain from (3.6) is singular. Numerically, to solve the problem (3.6)-(3.7), we done the following step:

1. Replace the first row of the system by $u_1 = 1$ (i.e. the first row of $\mathbb{A}$ is replaced by $(1, 0, \ldots, 0)$ and $\boldsymbol{b}_1$ is set to one).
2. Solve the new system $\mathbb{A}\boldsymbol{u}_h = \boldsymbol{b}$
3. Compute $I_h = \int_{\Gamma_h} u_h d\mathrm{q}$
4. Replace u_h by $u_h - I_h/|\Gamma_h|$ and then we have $\int_{\Gamma_h} u_h d\mathrm{q} = 0$.

By construction, we necessarily have $\int_\Gamma f \mathrm{d}A = 0$. Indeed for all $\varphi \in H^1(\Gamma)$ we have

$$-\int_\Gamma \Delta_\Gamma u \varphi \mathrm{d}A = \int_\Gamma f \varphi \mathrm{d}A$$

By using Green formula, we obtain

$$\int_\Gamma \langle \boldsymbol{\nabla}_\Gamma u, \boldsymbol{\nabla}_\Gamma \varphi \rangle \mathrm{d}A = \int_\Gamma f \varphi \mathrm{d}A$$

Taking $\varphi \equiv 1$ gives $\int_\Gamma f \mathrm{d}A = 0$.

On the unit sphere $\Gamma = S^2$, exact solution $u_{\mathrm{ex}}(x, y, z) = C + xy$

With Sage, one can compute $f = -\Delta_\Gamma u : (x, y, z) \mapsto \frac{6\,xy}{x^2+y^2+z^2}$ where $u : (x, y, z) \mapsto xy$. Furthermore, we have $\int_{S^2} u d\mathrm{q} = 0$ and so the constant C is equal to zero.

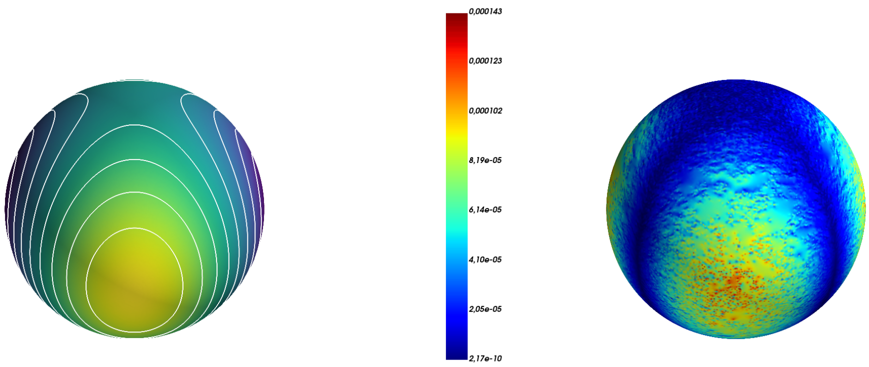


Figure 3.4: Numerical solution for 3D surface Laplace-Beltrami BVP on unit sphere (left) and error where exact solution is $u(x, y, z) = xy$ (right)

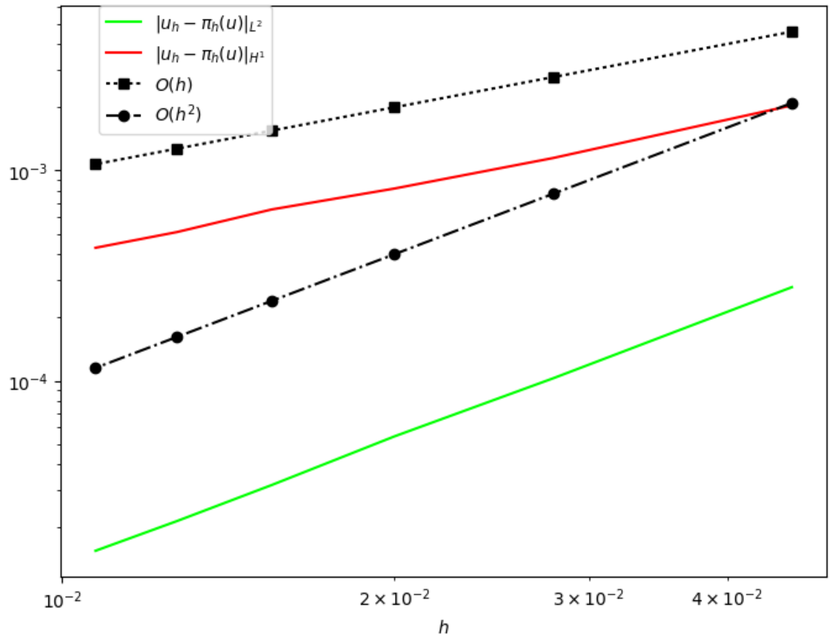


Figure 3.5: Order for 3D surface Laplace-Beltrami BVP on unit sphere : $u(x, y, z) = xy$.

On the unit sphere $\Gamma = S^2$**, exact solution** $u_{\text{ex}}(x, y, z) = C + x^2y^2$

With $u : (x, y, z) \mapsto x^2y^2$ we obtain from Sage

$$f = -\Delta_\Gamma u : (x, y, z) \mapsto -\frac{2\left(x^4 - 8\,x^2y^2 + y^4 + \left(x^2 + y^2\right)z^2\right)}{x^2 + y^2 + z^2}$$

Furthermore, we have $\int_{S^2} u d\text{q} \neq 0$ and so the constant C is not equal to zero.
To compute the contant C we use (**??**) and

$$\int_{S^2} u_{\text{ex}} d\text{q} = \int_{S^2} C dq + \int_{S^2} u d\text{q} = 4\pi C + \frac{4}{15}\pi = 0.$$

So we obtain

$$C = -\frac{1}{15}.$$

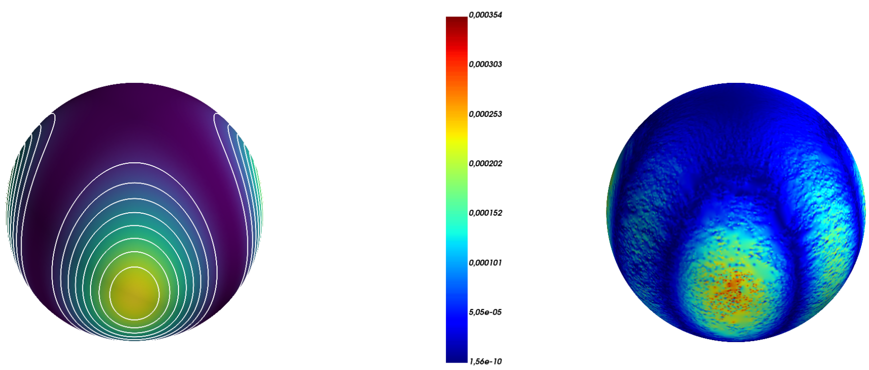


Figure 3.6: Numerical solution for 3D surface Laplace-Beltrami BVP on unit sphere (left) and error where exact solution is $u(x, y, z) = x^2y^2 + C$ (right)

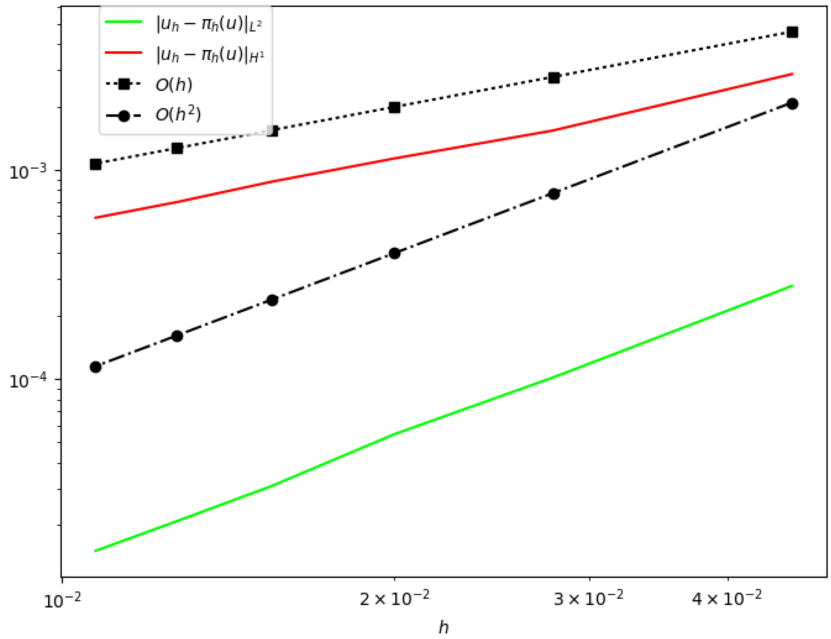


Figure 3.7: Order for 3D surface Laplace-Beltrami BVP on unit sphere : $u(x, y, z) = x^2y^2 + C$.

On the unit sphere, exact solution $u(x, y, z) = C + \cos(2\pi xy)\sin(2\pi z)$

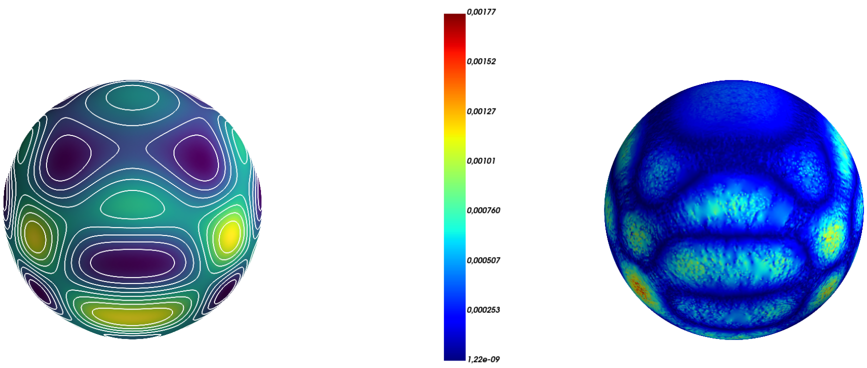


Figure 3.8: Numerical solution for 3D surface Laplace-Beltrami BVP on unit sphere (left) and error where exact solution is $u(x, y, z) = C + \cos(2\pi xy)\sin(2\pi z)$ (right)

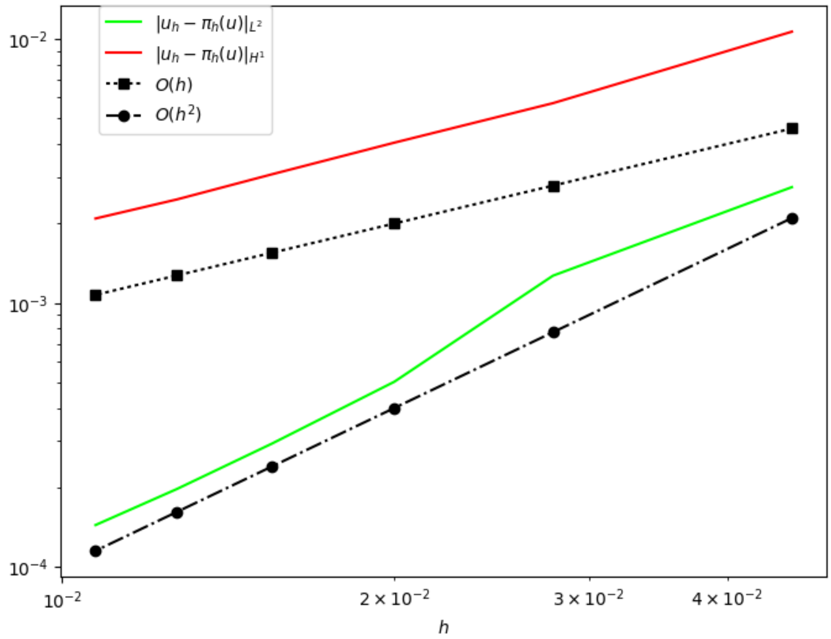


Figure 3.9: Order for 3D surface Laplace-Beltrami BVP on unit sphere : $u(x, y, z) = C + \cos(2\pi xy)\sin(2\pi z)$.

On the unit sphere mapping by a function F, exact solution $u(x, y, z) = C + xy$

The mapping function is given in (3.1).

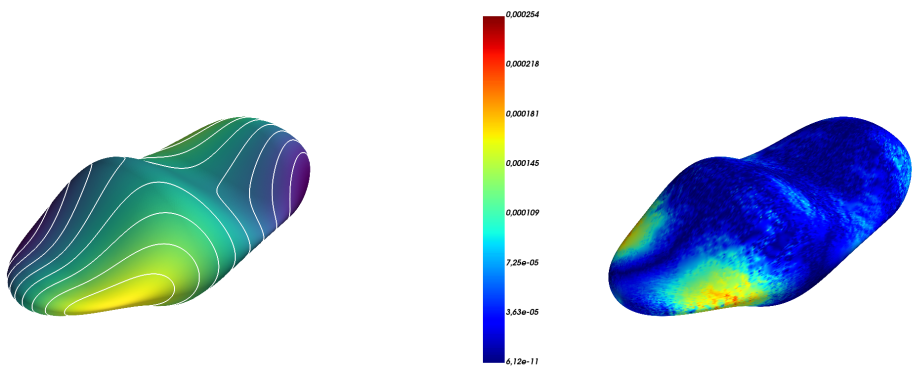


Figure 3.10: Numerical solution for 3D surface Laplace-Beltrami BVP on unit sphere mapping by function (3.1) (left) and error where exact solution is $u(x, y, z) = C + xy$ (right)

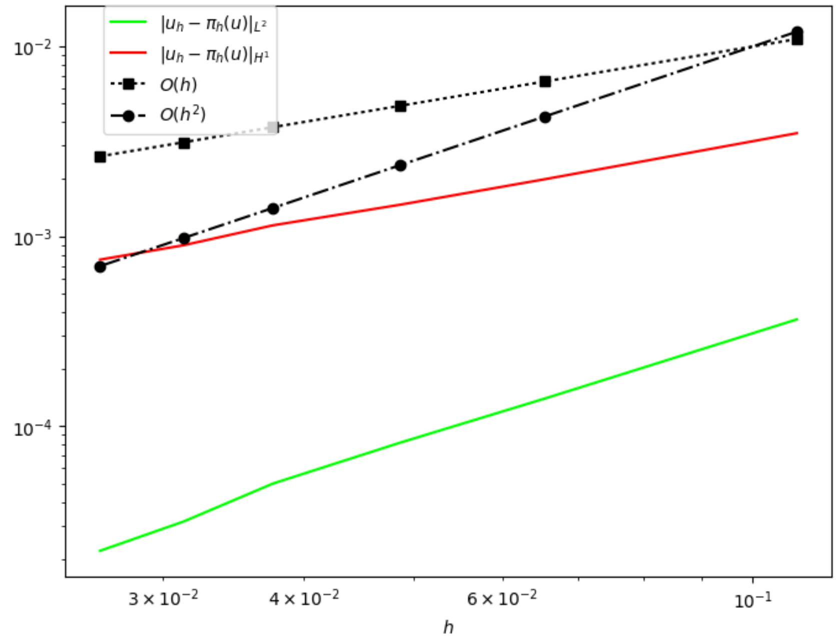


Figure 3.11: Order for 3D surface Laplace-Beltrami BVP on unit sphere mapping by function (3.1) : $u(x, y, z) = C + xy$.

On the unit sphere mapping by a function F, exact solution $u(x, y, z) = C + \cos(2\pi x) + \sin(2\pi z) + xyz$

The mapping function is given in (3.1).

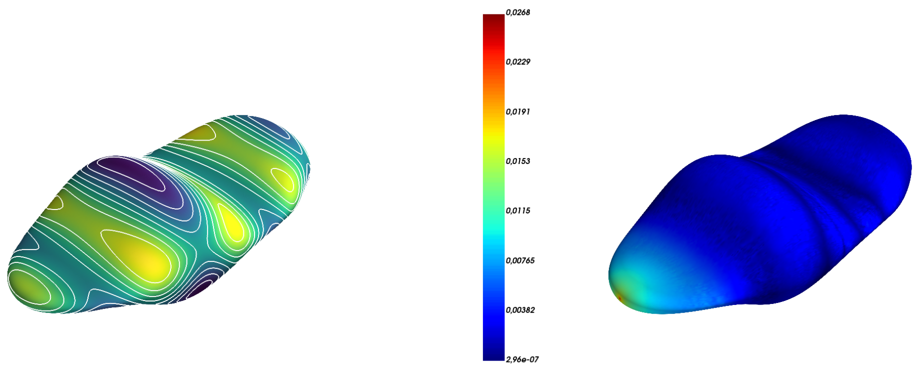


Figure 3.12: Numerical solution for 3D surface Laplace-Beltrami BVP on unit sphere mapping by function (3.1)(left) and error where exact solution is $u(x, y, z) = C + \cos(2\pi x) + \sin(2\pi z) + xyz$ (right)

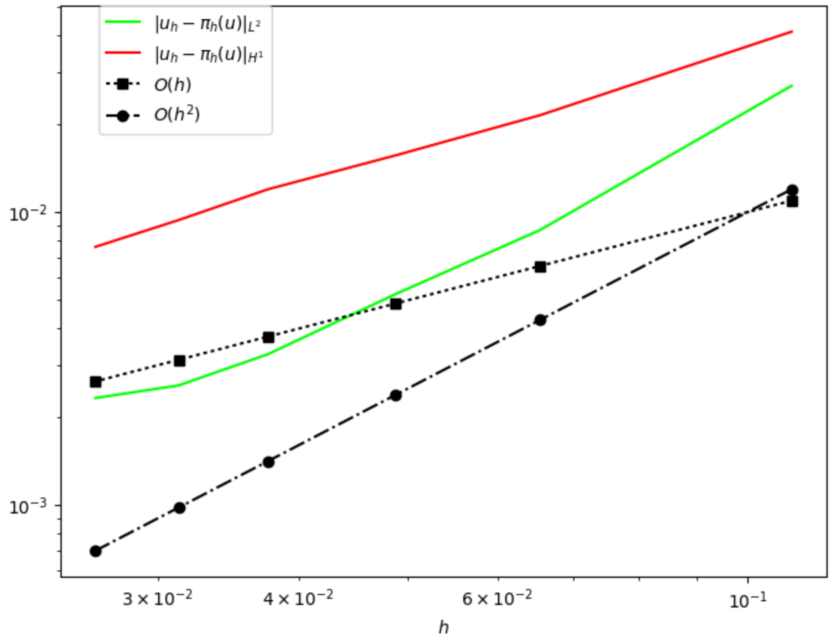


Figure 3.13: Order for 3D surface Laplace-Beltrami BVP on unit sphere mapping by function (3.1) : $u(x, y, z) = C + \cos(2\pi x) + \sin(2\pi z) + xyz$.

On the unit sphere mapping by a function F, exact solution $C + xy$

The hypersurface was constructed by mapping a discretization of the unit sphere S^2 onto the surface Γ by

$$\boldsymbol{F}(\boldsymbol{y}) = \left(y_1, y_2, y_3\sqrt{\frac{1}{20} + 2y_1^2 + \frac{1}{2}y_2^2} \right), \quad \boldsymbol{y} = (y_1, y_2, y_3) \in S^2. \tag{3.9}$$

The representation of $\Gamma = \boldsymbol{F}(S^2)$ as a hypersurface $\{\boldsymbol{x} \in \mathbb{R}^3 \mid \phi(\boldsymbol{x}) = 0\}$ follows from $y_1^2 + y_2^2 + y_3^3 = 1$ with the level set function

$$\phi(\boldsymbol{x}) = x_1^2 + x_2^2 + \frac{x_3^2}{\frac{1}{20} + 2y_1^2 + \frac{1}{2}y_2^2} - 1. \tag{3.10}$$

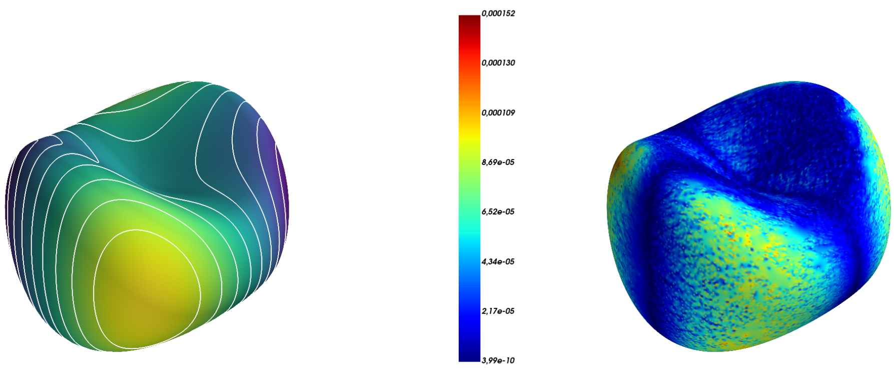


Figure 3.14: Numerical solution for 3D surface Laplace-Beltrami BVP on unit sphere mapping by function (3.9) (left) and error where exact solution is $u(x, y, z) = C + xy$ (right)

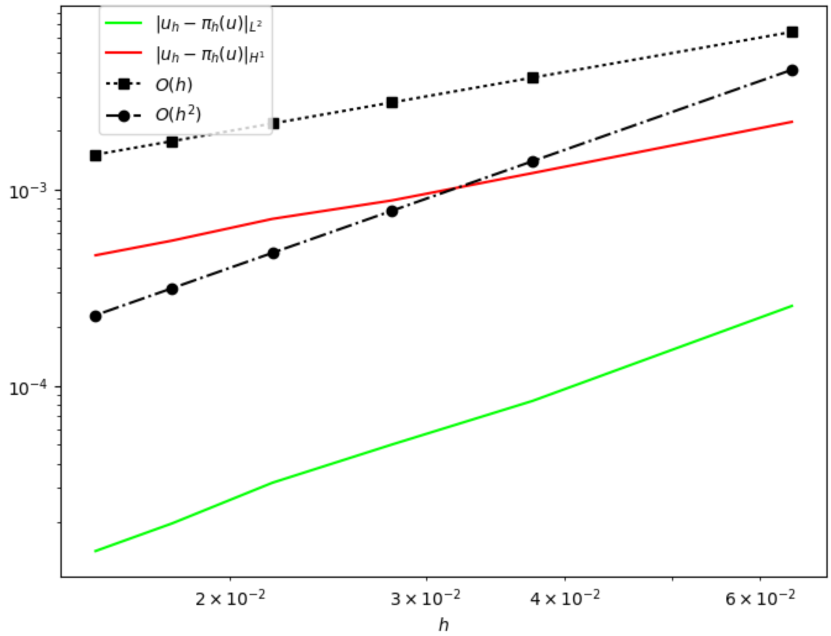


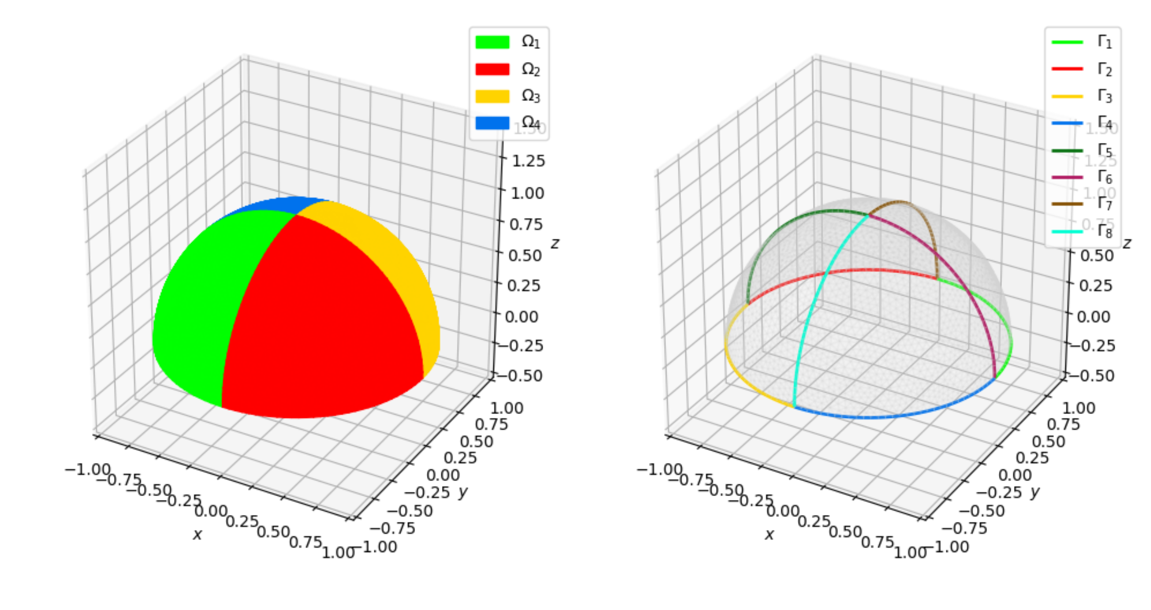
Figure 3.15: Order for 3D surface Laplace-Beltrami BVP on unit sphere mapping by function (3.9) : $u(x, y, z) = C + xy$.

3.3 Numerical samples on non-closed hypersurfaces

Let Γ be a non-closed hypersurfaces and $\partial\Gamma$ be its none empty boundary. We want to solve the scalar BVP (1.2)-(1.4)

3.3.1 $-\Delta_\Gamma u + \langle \boldsymbol{\nabla}_\Gamma u, \boldsymbol{c} \rangle + a_0 u = f$ with Dirichlet boundary conditions on an half-sphere

We represente in Listing 3.9, the unit demi-sphere $z \geqslant 0$. As we can remark, the *physical* boundaries are labeled from 1 to 4 : the label 5 to 8 only serves us for graphic representation purposes.




```
print('***␣Building/reading␣mesh')
N=20;geofile='halfsphere4surf';
(geodir,geofile)=get_geo(3,2,geofile)
meshfile=gmsh.buildmesh3ds(geodir+os.sep+geofile+'.geo',N,
    force=False,verbose=0)
Th=siMesh(meshfile)
```

Listing 3.9: Mesh of a half-sphere with 4 subdomains, $\Gamma = \Omega_1 \cup \cdots \cup \Omega_4$ (left) and boundaries $\partial\Gamma = \Gamma_1 \cup \cdots \cup \Gamma_4$ (right)

We want to solve the

Scalar surface BVP 3 : non-closed surface, sample 1

Find $u \in \mathrm{H}^1(\Gamma)$ such that

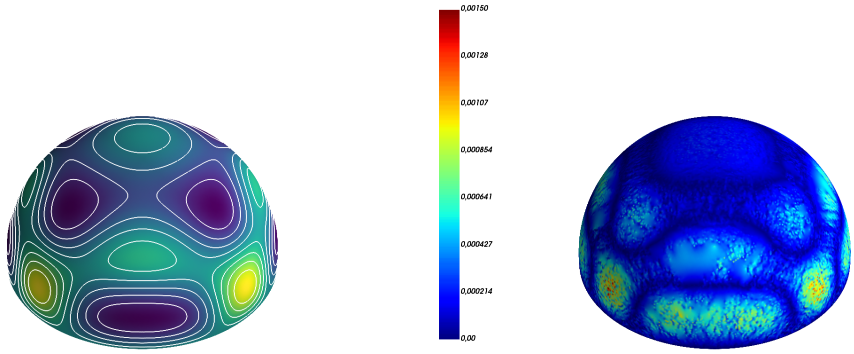
$$-\Delta_\Gamma u + \langle \nabla_\Gamma u, \boldsymbol{c}\rangle + a_0 u = f, \quad \text{in } \Gamma \tag{3.11}$$

$$u = g, \quad \text{on } \partial\Gamma. \tag{3.12}$$

With exact solution $u(x, y, z) = \cos(2\pi xy)\sin(2\pi z)$, we compute $-\Delta_\Gamma u$ and $\nabla_\Gamma u$ with Sage and store the results as Python functions, in the `fc_vfemp1_surface.examples.sBVPsamples` module, respectively named `fxyz02` and `gradSuxyz02` .

We give in Listing **??** the complet Python code to solve 3.5 with

$$a_0 : \boldsymbol{x} \mapsto 1 + x_1^2 \quad \text{and} \quad \boldsymbol{c} : \boldsymbol{x} \mapsto \begin{pmatrix} \cos(x_1) \\ \sin(x_2) \\ 2 + x_1 x_2 x_3 \end{pmatrix}.$$




```
from numpy import cos, sin, sqrt, pi
from fc_vfemp1_surface.examples import sBVPsamples
u=sBVPsamples.uxyz02
f=sBVPsamples.fxyz02
gradSu=sBVPsamples.gradSuxyz02
Fmap=sBVPsamples.Fmapxyz02
a0=lambda x,y,z: 1+x**2
c=[lambda x1,x2,x3: cos(x1), lambda x1,x2,x3: sin(x2), lambda x1,x2,x3:
    2+x1*x2*x3]
cgradSu=lambda x,y,z: np.sum(np.array(gradSu(x,y,z))*np.array([c[0](x,y,z),
    c[1](x,y,z),c[2](x,y,z)]),axis=0)
# RHS function :
RHSfun=lambda x,y,z: f(x,y,z)+cgradSu(x,y,z)+a0(x,y,z)*u(x,y,z)
print('*** Building/reading mesh')
N=75;
geofile='halfsphere4surf';
(geodir,geofile)=get_geo(3,2,geofile)
meshfile=gmsh.buildmesh3ds(geodir+os.sep+geofile+'.geo',N,
    force=False,verbose=0)
Th=siMesh(meshfile)
print('*** Setting surface BVP')

Lop=Loperator(dim=3,d=2,A=[[1,None,None],[None,1,None],[None,None,1]],c=c,a0=a0)
pde=PDE(Op=Lop,f=RHSfun)
bvp=BVP(Th,pde=pde)
for lab in [1,2,3,4]:
    bvp.setDirichlet(lab,u)

print('*** Solving surface BVP')
U=bvp.solve()
```

Listing 3.10: Solution of $-\Delta_\Gamma u + \langle \boldsymbol{\nabla}_\Gamma u, \boldsymbol{c} \rangle + a_0 u = f$ with Dirichlet boundary condition

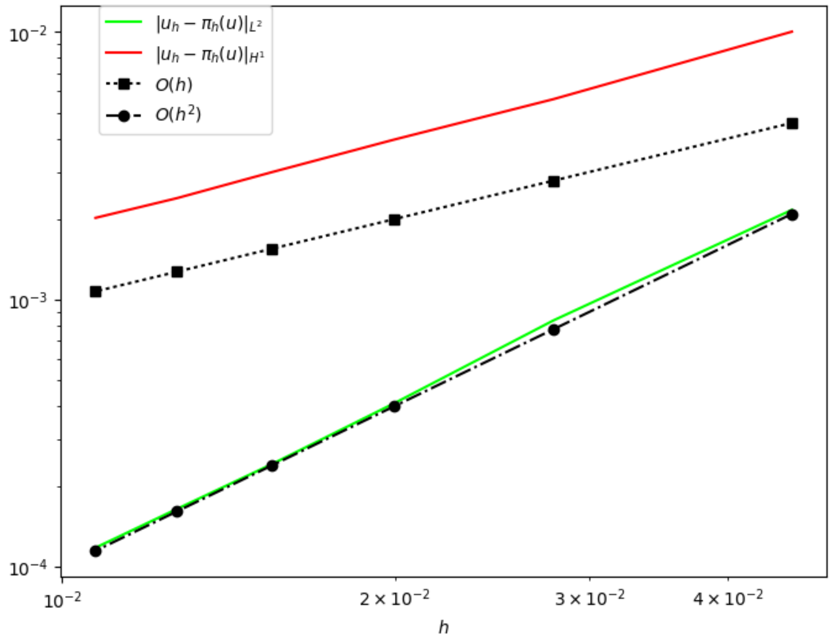


Figure 3.16: Order for 3Ds BVP $-\Delta_\Gamma u + \langle \nabla_\Gamma u, \boldsymbol{c}\rangle + a_0 u = f$ with Dirichlet boundary condition on unit half-sphere

3.3.2 $-\Delta_\Gamma u + \langle \nabla_\Gamma u, \boldsymbol{c}\rangle + a_0 u = f$ with mixed boundary conditions on an half-sphere

We want to solve the

Scalar surface BVP 4 : non-closed surface, sample 2

Find $u \in \mathrm{H}^1(\Gamma)$ such that

$$\begin{aligned} -\Delta_\Gamma u + \langle \nabla_\Gamma u, \boldsymbol{c}\rangle + a_0 u &= f, \quad \text{in } \Gamma & (3.13)\\ u &= g_D, \quad \text{on } \partial\Gamma_D. & (3.14)\\ \langle \nabla_\Gamma g, \boldsymbol{\mu}\rangle &= g_N, \quad \text{on } \partial\Gamma_N. & (3.15)\\ \langle \nabla_\Gamma g, \boldsymbol{\mu}\rangle + \alpha u &= g_R, \quad \text{on } \partial\Gamma_R. & (3.16) \end{aligned}$$

On the half-sphere ($z \geqslant 0$), we have $\boldsymbol{\mu} = (0, 0, -1)^{\mathsf{t}}$ on $\partial\Gamma$.

We also have to set g_N and g_R functions. We obtain them with the following Python code

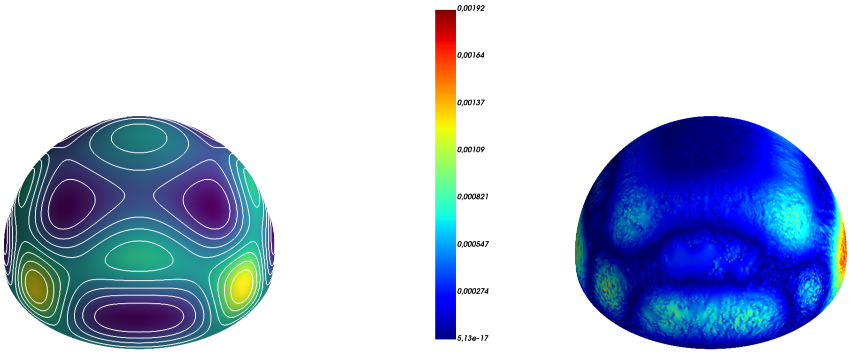
```
from fc_vfemp1_surface.examples import sBVPsamples
u=sBVPsamples.uxyz02
gradSu=sBVPsamples.gradSuxyz02
gN=lambda x,y,z: -gradSu(x,y,z)[2]
gR=lambda x,y,z: -gradSu(x,y,z)[2]+alpha(x,y,z)*u(x,y,z)
```

We give in Listing **??** the complet Python code to solve (3.13)-(3.16) with

$$\partial\Gamma_D = \Gamma_1 \cup \Gamma_3, \quad \partial\Gamma_N = \Gamma_2, \quad \partial\Gamma_R = \Gamma_4$$

and

$$a_0 : \boldsymbol{x} \mapsto 1 + x_1^2, \quad \boldsymbol{c} : \boldsymbol{x} \mapsto \begin{pmatrix} \cos(x_1) \\ \sin(x_2) \\ 2 + x_1 x_2 x_3 \end{pmatrix} \text{ and } \alpha : \boldsymbol{x} \mapsto 1 + x_3^2$$



```
from numpy import cos, sin, sqrt, pi
from fc_vfemp1_surface.examples import sBVPsamples
u=sBVPsamples.uxyz02
f=sBVPsamples.fxyz02
gradSu=sBVPsamples.gradSuxyz02
Fmap=sBVPsamples.Fmapxyz02
a0=lambda x,y,z: 1+x**2
c=[lambda x1,x2,x3: cos(x1), lambda x1,x2,x3: sin(x2), lambda x1,x2,x3:
    2+x1*x2*x3]
cgradSu=lambda x,y,z: np.sum(np.array(gradSu(x,y,z))*np.array([c[0](x,y,z),
    c[1](x,y,z),c[2](x,y,z)]),axis=0)
# RHS function :
RHSfun=lambda x,y,z: f(x,y,z)+cgradSu(x,y,z)+a0(x,y,z)*u(x,y,z)
alpha=lambda x,y,z: 1+z**2
print('*** Building/reading mesh')
N=75;
geofile='halfsphere4surf';
(geodir,geofile)=get_geo(3,2,geofile)
meshfile=gmsh.buildmesh3ds(geodir+os.sep+geofile+'.geo',N,
    force=False,verbose=0)
Th=siMesh(meshfile)
print('*** Setting surface BVP')

Lop=Loperator(dim=3,d=2,A=[[1,None,None],[None,1,None],[None,None,1]],c=c,a0=a0)
pde=PDE(Op=Lop,f=RHSfun)
bvp=BVP(Th,pde=pde)
for lab in [1,3]:
  bvp.setDirichlet(lab,u)
gN=lambda x,y,z: -gradSu(x,y,z)[2]
gR=lambda x,y,z: -gradSu(x,y,z)[2]+alpha(x,y,z)*u(x,y,z)
bvp.setRobin(2,gN)
bvp.setRobin(4,gR,ar=alpha)
print('*** Solving surface BVP')
U=bvp.solve()
```

Listing 3.11: Solution of $-\Delta_\Gamma u + \langle \nabla_\Gamma u, \boldsymbol{c}\rangle + a_0 u = f$ with mixed boundary condition

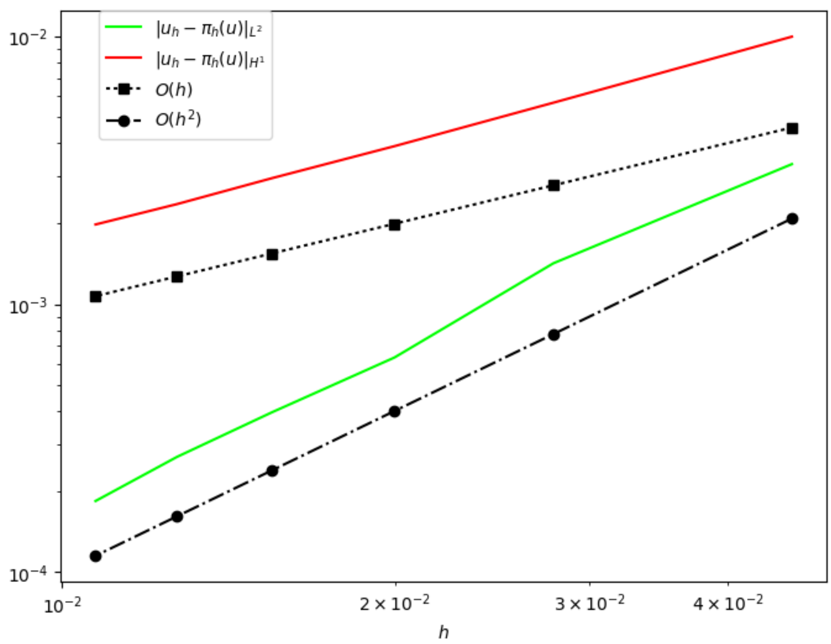
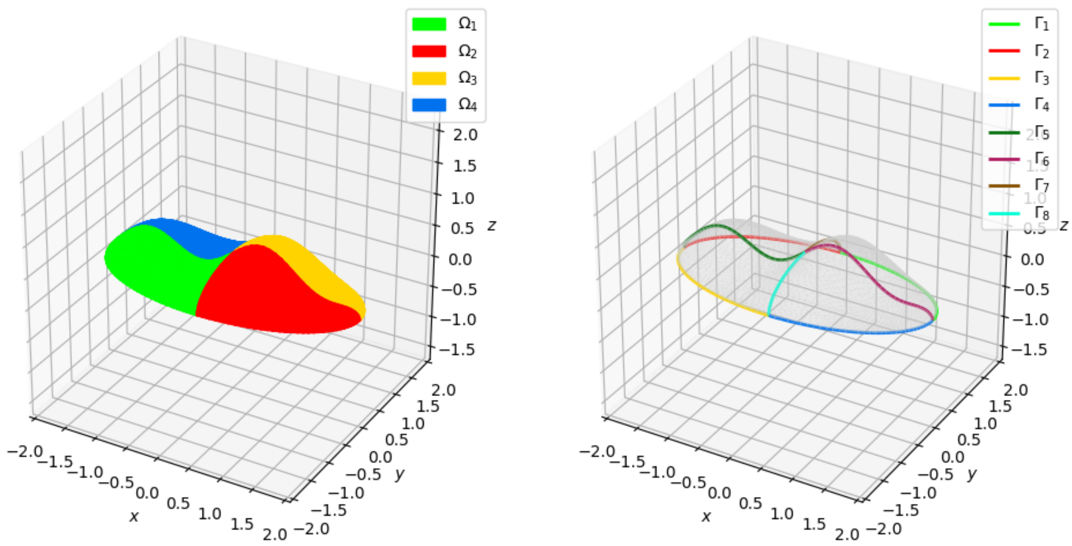


Figure 3.17: Order for 3Ds BVP $-\Delta_\Gamma u + \langle \nabla_\Gamma u, \boldsymbol{c} \rangle + a_0 u = f$ with mixed boundary condition on unit half-sphere

3.3.3 Mixed boundary conditions on an half-sphere mapping by a function

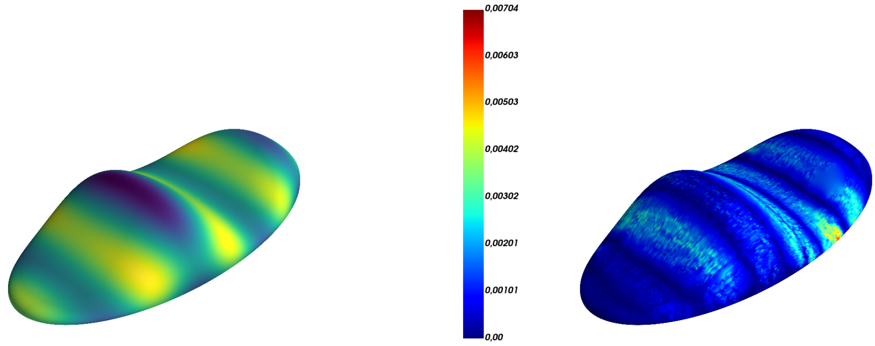
We represente in Listing 3.9, the unit demi-sphere $z \geqslant 0$. As we can remark, the *physical* boundaries are labeled from 1 to 4 : the label 5 to 8 only serves us for graphic representation purposes.




```
from numpy import sin, pi
Fmap=lambda x,y,z: [2*x,y,0.5*z*(1+0.5*sin(2*pi*x))]
N=20;geofile='halfsphere4surf';
(geodir,geofile)=get_geo(3,2,geofile)
meshfile=gmsh.buildmesh3ds(geodir+os.sep+geofile+'.geo',N,
     force=False,verbose=0)
Th=siMesh(meshfile,mapping=Fmap)
```

Listing 3.12: Mesh of a half-sphere with 4 subdomains, $\Gamma = \Omega_1 \cup \cdots \cup \Omega_4$ (left) and boundaries $\partial\Gamma = \Gamma_1 \cup \cdots \cup \Gamma_4$ (right)

On this mesh, we also have $\boldsymbol{\mu} = (0, 0, -1)^t$ on $\partial\Gamma$.




```
from numpy import cos, sin, pi
from fc_vfemp1_surface.examples.sBVP02_halfsphere_Mixed import run
from fc_vfemp1_surface.examples import sBVPsamples
u=sBVPsamples.uxyz04
f=sBVPsamples.fxyz04
Fmap=sBVPsamples.Fmapxyz04
gradSu=sBVPsamples.gradSuxyz04
a0=lambda x,y,z: 1+x**2
alpha=lambda x,y,z: 1+z**2
c=[lambda x1,x2,x3: cos(x1), lambda x1,x2,x3: sin(x2), lambda x1,x2,x3:
    2+x1*x2*x3]
run(N=70,u=u,f=f,Fmap=Fmap,gradSu=gradSu,a0=a0,c=c,alpha=alpha)
```

Listing 3.13: Solution of the surface BVP $-\Delta_\Gamma u + \langle \boldsymbol{\nabla}_\Gamma u, \boldsymbol{c}\rangle + a_0 u = f$ on unit half-sphere mapping by function (3.1) with mixed boundary conditions

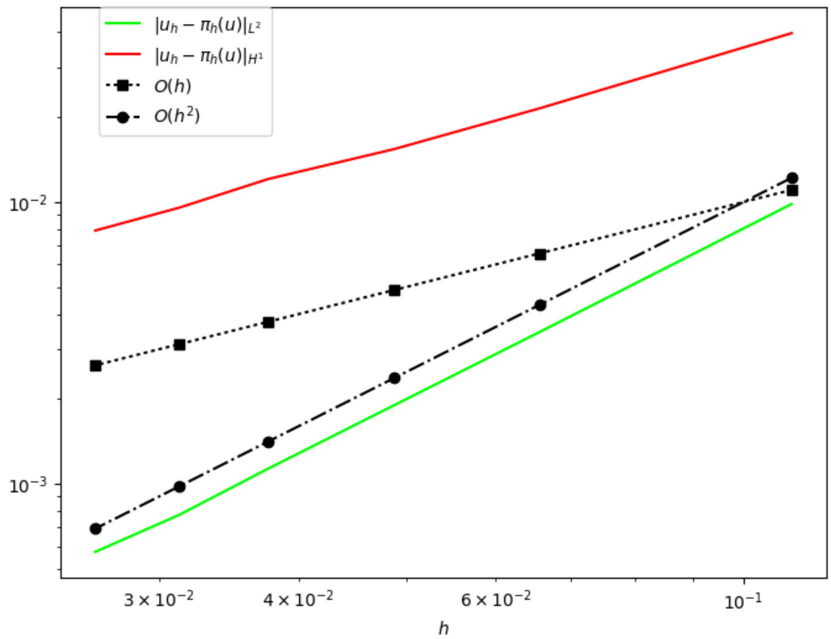


Figure 3.18: Order for the surface BVP $-\Delta_\Gamma u + \langle \boldsymbol{\nabla}_\Gamma u, \boldsymbol{c}\rangle + a_0 u = f$ with mixed boundary condition on unit half-sphere mapping by function (3.1).

Bibliography

[1] F. Cuvelier. fc_simesh: a object-oriented Python package for using simplices meshes generated from gmsh (in dimension 2 or 3) or an hypercube triangulation (in any dimension). http://www.math.univ-paris13.fr/~cuvelier/software/, 2017. User's Guide.

[2] F. Cuvelier. fc_simesh_matplotlib: an add-on to the fc_simesh Python package for displaying simplices meshes or datas on simplices meshes by using matplotlib python package. http://www.math.univ-paris13.fr/~cuvelier/software/, 2017. User's Guide.

[3] F. Cuvelier. fc_simesh_mayavi: an add-on to the fc_simesh Python package for displaying simplices meshes or datas on simplices meshes by using mayavi python package. http://www.math.univ-paris13.fr/~cuvelier/software/, 2017. User's Guide.

[4] F. Cuvelier and G. Scarella. A generic way to solve partial differential equations by the $\mathbb{P}_1$-Lagrange finite element method in vector languages. https://www.math.univ-paris13.fr/~cuvelier/software/docs/Recherch/VecFEM/distrib/0.1b1/vecFEMP1_report-0.1b1.pdf, 2015.

[5] François Cuvelier, Caroline Japhet, and Gilles Scarella. An efficient way to assemble finite element matrices in vector languages. *BIT Numerical Mathematics*, 56(3):833–864, dec 2015.

[6] The Sage Developers. *Sage Mathematics Software (Version 6.8)*, 2015. http://www.sagemath.org.

[7] Gerhard Dziuk and Charles M. Elliott. Finite element methods for surface pdes. *Acta Numerica*, 22:289–396, 5 2013.

[8] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*, volume 23 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 1994.

GIT : commit bfad42e9f9594171310a29a95eaa9e7aaed6a796 Date: Fri Jun 23 14:10:03 2017 +0200