



# FC-VFEMP<sub>1</sub>-EIGS Python package, User's Guide <sup>1</sup>

François Cuvelier<sup>2</sup>

2017/06/19

<sup>1</sup>Compiled with Python 3.6.0

<sup>2</sup>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 $\mathbb{P}_1$ -EIGS is an **experimental** object-oriented Python package dedicated to solve scalar or vector **eigenvalue** boundary value problems (eBVP) 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.

In the first chapter, the FC-VFEM $\mathbb{P}_1$  package is quickly presented with some examples. Thereafter, in the second chapter a generalized eigenvalue problem coming from *scalar* BVP is decribed. Some examples are proposed and numericaly solved by using the FC-VFEM $\mathbb{P}_1$ -EIGS package. Finally in the last chapter a generalized eigenvalue problem coming from *vector* BVP is given. Some biharmonic eigenvalue problems are numericaly solved.

# Contents

<b>1</b>	<b>Generic Boundary Value Problems</b>	<b>2</b>
1.1	Scalar boundary value problem . . . . .	2
1.2	Vector boundary value problem . . . . .	4
<b>2</b>	<b>Generalized Eigenvalue scalar BVP</b>	<b>7</b>
2.1	eBVPsolve function . . . . .	7
2.2	2D samples . . . . .	8
2.2.1	2D Laplace eigenvalues problem with Dirichlet boundary condition . . . . .	8
2.2.2	2D Laplace eigenvalues problem with mixed boundary conditions . . . . .	20
2.2.3	Other 2D eigenvalues problems with Dirichlet boundary condition . . . . .	24
2.3	3D examples . . . . .	26
2.3.1	Eigenvalues of the laplacian on the unit sphere with Dirichlet boundary condition . . . . .	26
<b>3</b>	<b>Generalized Eigenvalue vector BVP</b>	<b>29</b>
3.1	Biharmonic eigenvalue BVP for plate vibration . . . . .	29
3.1.1	Simply supported plate . . . . .	30
3.1.2	Mixed boundary conditions . . . . .	31
3.2	Linear elasticity . . . . .	36
3.2.1	Elasticity problem . . . . .	36
3.2.2	2D example . . . . .	39
3.2.3	3D example . . . . .	42
<b>A</b>	<b>Biharmonic BVP</b>	<b>45</b>
A.1	Link with $\mathcal{H}$ -operator . . . . .	45
A.2	Some boundary conditions . . . . .	48
A.2.1	Clamped Plate boundary condition . . . . .	48
A.2.2	Simply Supported Plate boundary condition . . . . .	49
A.2.3	Cahn-Hilliard boundary condition . . . . .	49

# Chapter 1

## Generic Boundary Value Problems

The notations of [10] are employed in this section and extended to the vector case.

### 1.1 Scalar boundary value problem

Let  $\Omega$  be a bounded open subset of  $\mathbb{R}^d$ ,  $d \geq 1$ . The boundary of  $\Omega$  is denoted by  $\Gamma$ .

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

$$\mathcal{L}_{\mathbb{A}, \mathbf{b}, \mathbf{c}, a_0}(u) \stackrel{\text{def}}{=} -\operatorname{div}(\mathbb{A} \nabla u) + \operatorname{div}(\mathbf{b}u) + \langle \nabla u, \mathbf{c} \rangle + a_0 u \quad (1.1)$$

where  $\mathbb{A} \in (L^\infty(\Omega))^{d \times d}$ ,  $\mathbf{b} \in (L^\infty(\Omega))^d$ ,  $\mathbf{c} \in (L^\infty(\Omega))^d$  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 [10] and we note that we can omit either  $\operatorname{div}(\mathbf{b}u)$  or  $\langle \nabla u, \mathbf{c} \rangle$  if  $\mathbf{b}$  and  $\mathbf{c}$  are sufficiently regular functions. We keep both terms with  $\mathbf{b}$  and  $\mathbf{c}$  to deal with more boundary conditions. It should be also noted that it is important to preserve the two terms  $\mathbf{b}$  and  $\mathbf{c}$  in the generic formulation to enable a greater flexibility in the choice of the boundary conditions.

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

A *scalar* boundary value problem is given by

#### **Scalar BVP 1 : generic problem**

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

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

$$u = g^D \quad \text{on } \Gamma^D, \quad (1.3)$$

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

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

$$\frac{\partial u}{\partial n_{\mathcal{L}}} \stackrel{\text{def}}{=} \langle \mathbb{A} \nabla u, \mathbf{n} \rangle - \langle \mathbf{b}u, \mathbf{n} \rangle \quad (1.5)$$

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  $u \in H^2(\Omega)$  such that

$$-\Delta u = 0 \text{ in } \Omega \subset \mathbb{R}^2, \quad (1.6)$$

$$u = 0 \text{ on } \Gamma_1, \quad (1.7)$$

$$u = -12 \text{ on } \Gamma_{98}, \quad (1.8)$$

$$u = 12 \text{ on } \Gamma_{99}, \quad (1.9)$$

where  $\Omega$  and its boundaries are given in Figure 1.1.

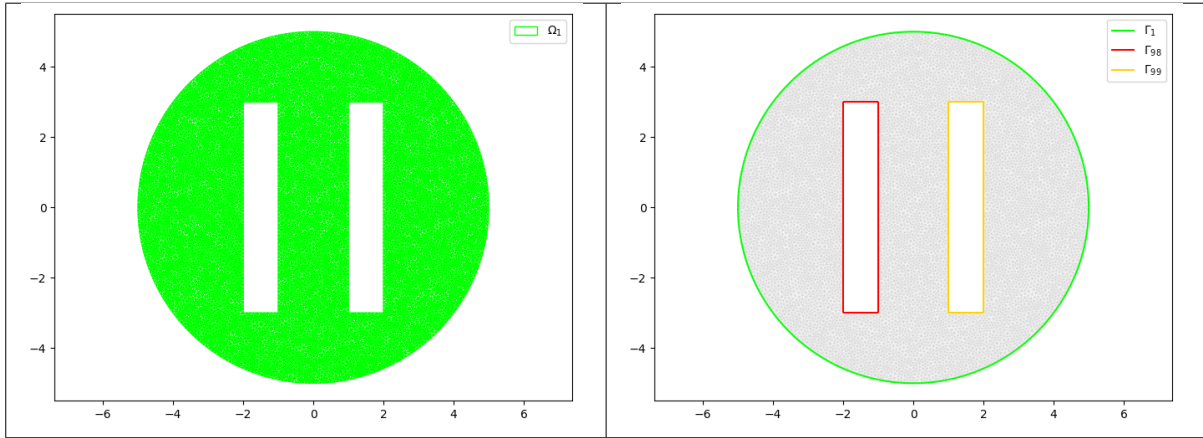


Figure 1.1: 2D condenser mesh and boundaries (left) and numerical solution (right)

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  $u \in H^2(\Omega)$  such that

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

where  $\mathcal{L} := \mathcal{L}_{1,0,0,0}$ ,  $f \equiv 0$ , and

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

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

```
meshfile=gmsb.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)
splt.plot(Th,u)
plt.colorbar(label='u')
set_axes_equal()
plt.figure(4)
splt.plotiso(Th,u,contours=15)
plt.colorbar(label='u')
splt.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

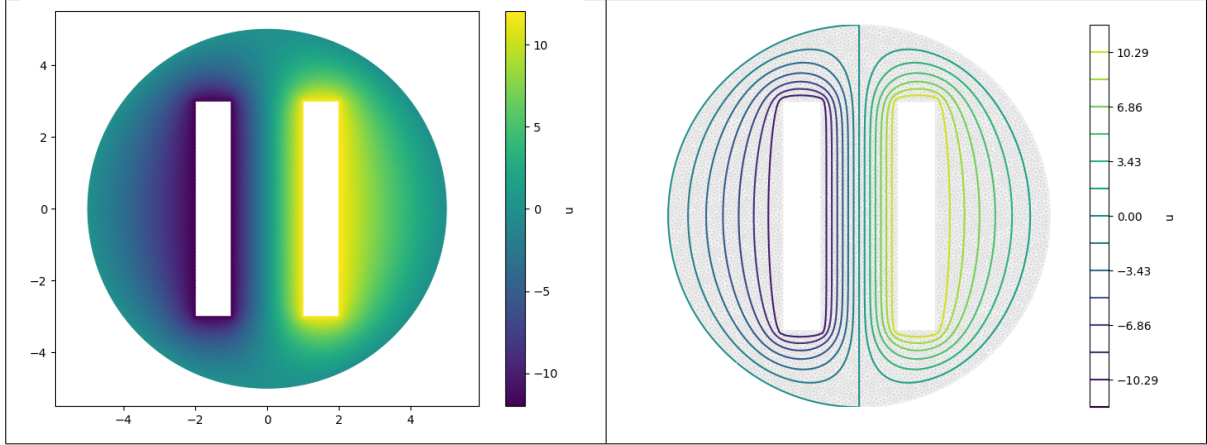


Figure 1.2: 2D condenser numerical solution

## 1.2 Vector boundary value problem

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

$$\begin{cases} \mathcal{H} : (H^2(\Omega))^m & \longrightarrow (L^2(\Omega))^m \\ \mathbf{u} = (\mathbf{u}_1, \dots, \mathbf{u}_m) & \longmapsto \mathbf{f} = (\mathbf{f}_1, \dots, \mathbf{f}_m) \stackrel{\text{def}}{=} \mathcal{H}(\mathbf{u}) \end{cases} \quad (1.10)$$

where

$$\mathbf{f}_\alpha = \sum_{\beta=1}^m \mathcal{H}_{\alpha,\beta}(\mathbf{u}_\beta), \quad \forall \alpha \in \llbracket 1, m \rrbracket, \quad (1.11)$$

with, for all  $(\alpha, \beta) \in \llbracket 1, m \rrbracket^2$ ,

$$\mathcal{H}_{\alpha,\beta} \stackrel{\text{def}}{=} \mathcal{L}_{\mathbb{A}^{\alpha,\beta}, \mathbf{b}^{\alpha,\beta}, \mathbf{c}^{\alpha,\beta}, a_0^{\alpha,\beta}} \quad (1.12)$$

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

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

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

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

A *vector* boundary value problem is given by

### **Vector BVP 1 : generic problem**

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

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

$$\mathbf{u}_\alpha = g_\alpha^D \quad \text{on } \Gamma_\alpha^D, \quad \forall \alpha \in \llbracket 1, m \rrbracket, \quad (1.15)$$

$$\frac{\partial \mathbf{u}}{\partial n_{\mathcal{H}_\alpha}} + a_\alpha^R \mathbf{u}_\alpha = g_\alpha^R \quad \text{on } \Gamma_\alpha^R, \quad \forall \alpha \in \llbracket 1, m \rrbracket, \quad (1.16)$$

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

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

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  $\frac{\partial \mathbf{u}}{\partial n_{\mathcal{H}_1}} + a_1^R \mathbf{u}_1 = g_1^R$  and a Dirichlet one with  $\mathbf{u}_2 = g_2^D$ .

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

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

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

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

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

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

where  $\Omega$  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  $\mathbf{u} = (u_1, u_2) \in (\mathbf{H}^2(\Omega))^2$  such that

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

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

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

where

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

$$f \equiv 0,$$

and

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

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

```
meshfile=gmsb.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

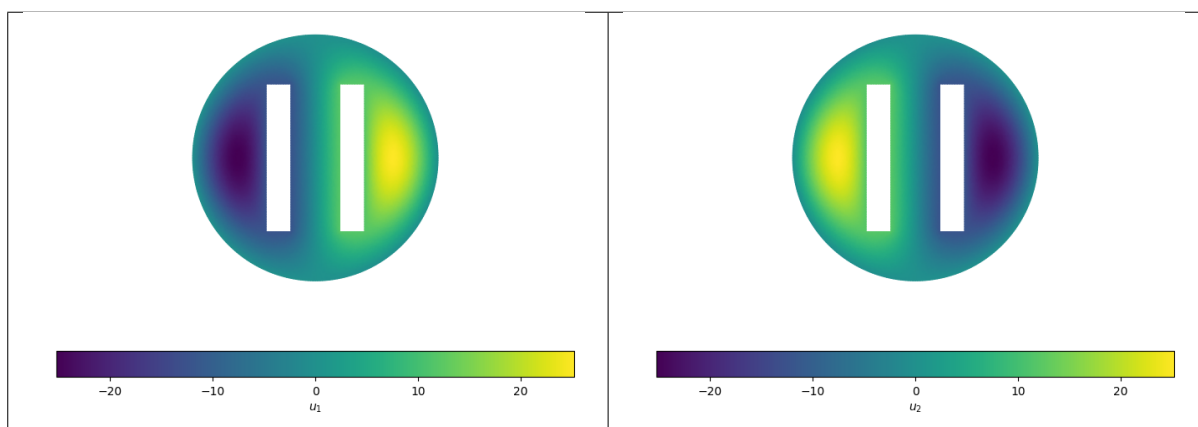


Figure 1.3: Funny vector BVP,  $u_1$  numerical solution (left) and  $u_2$  numerical solution (right)

# Chapter 2

## Generalized Eigenvalue scalar BVP

We want to solve generalized eigenvalue problems coming from scalar BVP's.

The **generalized eigenvalue problem** associated with *scalar* BVP (1.2)-(1.4) can be written as



### Scalar EBVP 1 : generic problem

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

$$\mathcal{L}(u) = \lambda \mathcal{B}(u) \quad \text{in } \Omega, \quad (2.1)$$

$$u = 0 \quad \text{on } \Gamma^D, \quad (2.2)$$

$$\frac{\partial u}{\partial n_{\mathcal{L}}} + a^R u = 0 \quad \text{on } \Gamma^R. \quad (2.3)$$

where  $\mathcal{B} = \mathcal{L}_{\mathbb{O}_{d \times d}, \mathbf{0}_d, \tilde{\mathbf{c}}, \tilde{a}_0}$ .

In the next section, we briefly describe the main function that will be used to solve eigenvalue boundary value problems. Let `bvp` be a `BVP` object. Thereafter some eigenvalue problems are given and solved by using the FC-VFEM $\mathbb{P}_1$ -EIGS package.

### 2.1 eBVPsolve function

The function `fc_vfemp1_eigs.lib.eBVPsolve` returns a few eigenvalues and eigenvectors obtained by solving a generalized eigenvalue scalar BVP with  $\mathbb{P}_1$ -Lagrange finite elements.

```
eigValues, eigVectors = fc_vfemp1_eigs.lib.eBVPsolve(bvp)
eigValues, eigVectors = fc_vfemp1_eigs.lib.eBVPsolve(bvp, **kwargs)
eigValues, eigVectors = fc_vfemp1_eigs.lib.eBVPsolve(bvp, RHSop=Bop, **kwargs)
```

**Description** The inputs are :

- `bvp` a `BVP` object which described the *scalar* BVP (1.2)-(1.4) with all right-hand sides equal to zeros, i.e.  $f := 0$ ,  $g^D := 0$  and  $g^R := 0$ .
- `Bop` a `Loperator` object corresponding to operator  $\mathcal{B}$ . By default `Bop` is the operator  $\mathcal{L}_{\mathbb{O}_{d \times d}, \mathbf{0}_d, \mathbf{0}_d, 1}$  for scalar BVP.

- `**kwargs` are the optional parameters of the `scipy.sparse.linalg.eigs` Python function.

The outputs are those given by the `scipy.sparse.linalg.eigs` Python function where the eigenvalues are stored in the complex numpy array `eigValues` which is sorted with `numpy.argsort`. The eigenvectors are stored in the complex numpy array `eigVectors` such that `eigValues.shape[0]==eigVectors.shape[1]` and `eigVectors[:, i]` is the eigenvector associated with the eigenvalue `eigValues[i]`.

## 2.2 2D samples

### 2.2.1 2D Laplace eigenvalues problem with Dirichlet boundary condition

We want to solve the eigenvalue problem given by (2.4)-(2.5).



#### Usual EBVP 1 : 2D Laplace with Dirichlet boundary condition

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

$$-\Delta u = \lambda u \quad \text{in } \Omega, \quad (2.4)$$

$$u = 0 \quad \text{on } \Gamma, \quad (2.5)$$

The problem (2.4)-(2.5) can be equivalently write as the *Scalar* EBVP 1:



#### Scalar EBVP 2 : 2D Laplace with Dirichlet boundary condition

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

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

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

where  $\mathcal{L} = \mathcal{L}_{1,0,0,0}$ ,  $\mathcal{B} = \mathcal{L}_{0,0,0,1}$ .

**Applications on the rectangle**  $\Omega = [0, L] \times [0, H]$ .

The eigenvalues and the associated eigenfunctions are given by

$$\lambda_{k,l} = \left( \frac{k\pi}{L} \right)^2 + \left( \frac{l\pi}{H} \right)^2, \quad u_{k,l}(x, y) = \sin\left(\frac{k\pi}{L}x\right) \sin\left(\frac{l\pi}{H}y\right), \quad \forall (k, l) \in \mathbb{N}^* \times \mathbb{N}^*. \quad (2.6)$$

In Table 2.1, the first eigenvalues are given for  $(k, l) \in \llbracket 1, 5 \rrbracket$ .

$\begin{matrix} k \\ l \end{matrix}$	1	2	3	4	5
1	$\frac{13}{36}\pi^2 \approx 3.56402$	$\frac{10}{9}\pi^2 \approx 10.9662$	$\frac{85}{36}\pi^2 \approx 23.3032$	$\frac{37}{9}\pi^2 \approx 40.5750$	$\frac{229}{36}\pi^2 \approx 62.7817$
2	$\frac{25}{36}\pi^2 \approx 6.85389$	$\frac{13}{9}\pi^2 \approx 14.2561$	$\frac{97}{36}\pi^2 \approx 26.5931$	$\frac{40}{9}\pi^2 \approx 43.8649$	$\frac{241}{36}\pi^2 \approx 66.0715$
3	$\frac{5}{4}\pi^2 \approx 12.3370$	$2\pi^2 \approx 19.7392$	$\frac{13}{4}\pi^2 \approx 32.0762$	$5\pi^2 \approx 49.3480$	$\frac{29}{4}\pi^2 \approx 71.5546$
4	$\frac{73}{36}\pi^2 \approx 20.0134$	$\frac{25}{9}\pi^2 \approx 27.4156$	$\frac{145}{36}\pi^2 \approx 39.7526$	$\frac{52}{9}\pi^2 \approx 57.0244$	$\frac{325}{36}\pi^2 \approx 79.2310$
5	$\frac{109}{36}\pi^2 \approx 29.8830$	$\frac{34}{9}\pi^2 \approx 37.2852$	$\frac{181}{36}\pi^2 \approx 49.6222$	$\frac{61}{9}\pi^2 \approx 66.8940$	$\frac{325}{36}\pi^2 \approx 89.1006$

Table 2.1: Eigenvalues  $\lambda_{k,l}$  for  $(k, l) \in \llbracket 1, 5 \rrbracket$  with  $L = 3$ ,  $H = 2$

In file `laplacian_DDDD_rectangle_ex.py`<sup>1</sup>, the `run` function solve the Dirichlet eigenvalue problem for the laplacian on  $\Omega = [0, L] \times [0, H]$ . In Listing 3.3, part of the `run` function is given.

```

return
    print( '***_Setting_the_mesh_using_HyperCube_function_[fc_simesh]' )
    Th=HyperCube(2,N, mapping=lambda q:np.array([L*q[0],H*q[1]]))
    print( '***_Setting_the_mesh_using_gmsh_and_%s.geo_file'%(geoFile) )
    meshfile=gmsh.buildmesh2d(geoFile,N, force=True, verbose=0, options='-smooth_4_
    -setnumber_H_%g_-setnumber_L_%g'%(H,L) )

```

<sup>1</sup>directory: `fc_vfemp1_eigs/examples`

```

print('***_Setting_the_eBVP')
Lop=Loperator(dim=2,A=[[1,None],[None,1]])
pde=PDE(Op=Lop)
bvp=BVP(Th,pde=pde)
for lab in Th.sThlab[Th.find(d=1)]:
    print('***_Solving_the_eBVP')
    print('***_Computing_exact_solutions_to_the_eBVP')

```

Listing 2.1: eigenvalue problem  $\Omega = [0, L] \times [0, H]$ .

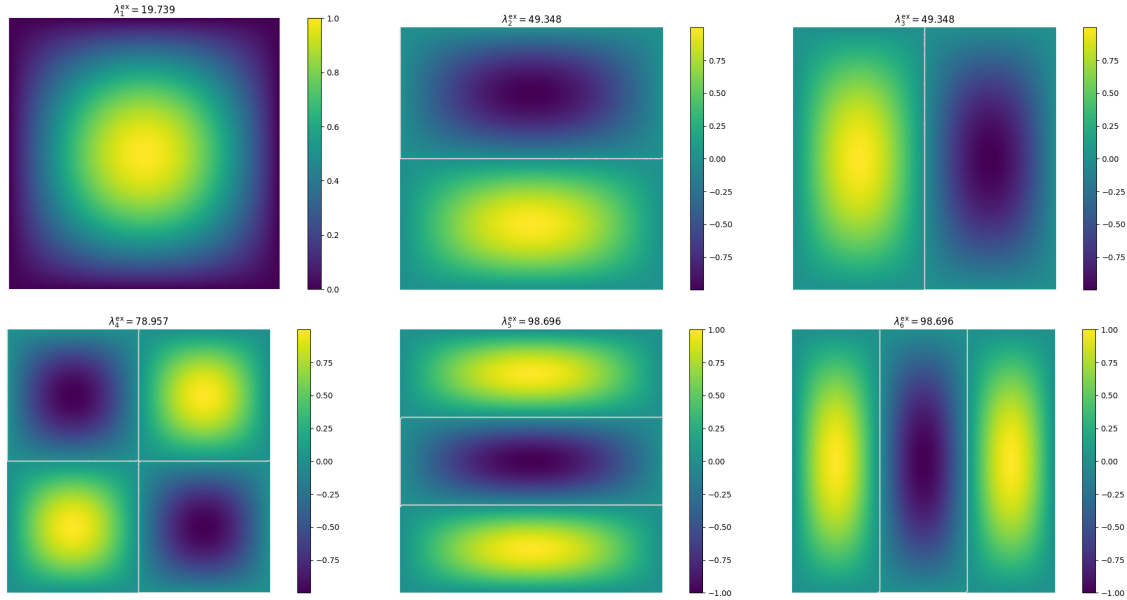


Figure 2.1: Dirichlet eigenvalue problem for the laplacian : exact eigenvectors of the smallest magnitude eigenvalues

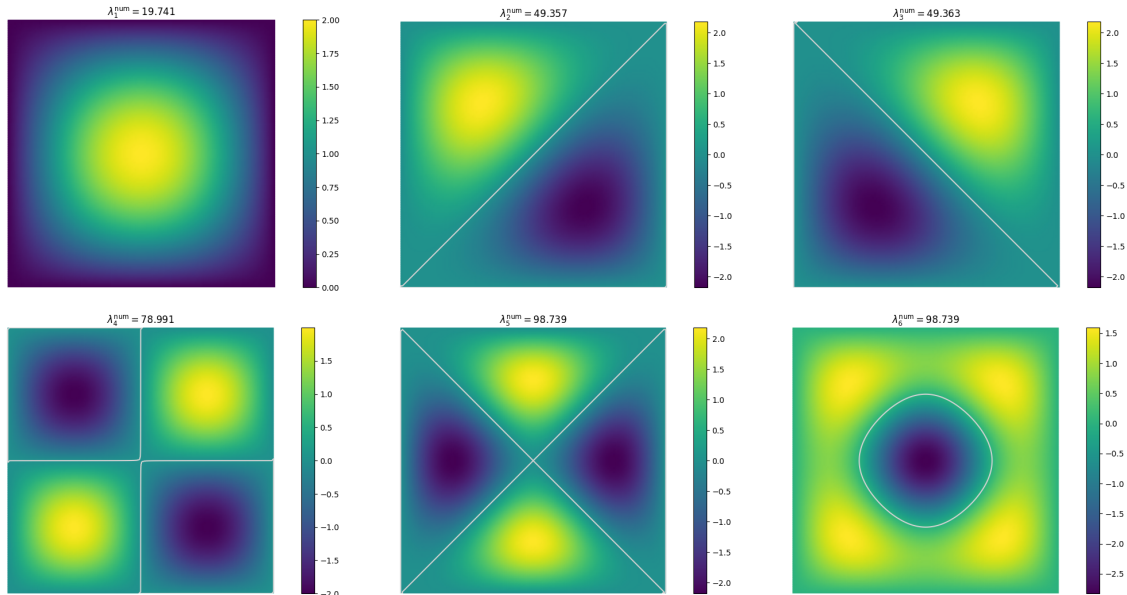


Figure 2.2: Dirichlet eigenvalue problem for the laplacian : numerical eigenvectors of the smallest magnitude eigenvalues

The Figure 2.1 and Figure 2.2 are obtained by running the following python code:

```
from fc_vfemp1_eigs.examples.laplacian_DDDD_rectangle_ex import run
res=run(L=1,H=1,trans=False , gerror=False , colorbar=True)
```

Listing 2.2: Dirichlet eigenvalues problem for the laplacian on  $\Omega = [0, 1] \times [0, 1]$  without transformation

As we can see by comparing the exact eigenfunctions given in Figure 2.1 with the numerical eigenfunctions given in Figure 2.2, their graphic representations differ. With an eigenvalue's geometric multiplicity  $n \geq 1$ , we have a set of  $n$  exact eigenfunctions  $\{u_1, \dots, u_n\}$  (eigenspace) to compare with the set of  $n$  numerical eigenfunctions  $\{v_{h,1}, \dots, v_{h,n}\}$ . So we transform the  $n$  numerical eigenfunctions by the following linear combination

$$\hat{v}_{h,i} = \sum_{j=1}^n \frac{\langle u_{h,i}, v_{h,j} \rangle}{\langle v_{h,j}, v_{h,j} \rangle} v_{h,j}$$

where  $\langle u, v \rangle = \int_{\Omega} \bar{u}(q)v(q)dq$  and  $u_{h,i} = \pi_h(u_i)$  is the  $\mathbb{P}^1$ -Lagrange interpolate of  $u_i$ . This transformation is applied on each eigenspace by the function `L2_complete_trans`

```
if trans:
```

In Figure 2.3 we represent the transformed eigenfunctions and in Figure ?? the relative errors. These results are obtained by running the following python code:

```
from fc_vfemp1_eigs.examples.laplacian_DDDD_rectangle_ex import run
res=run(L=1,H=1)
```

Listing 2.3: Dirichlet eigenvalues problem for the laplacian on  $\Omega = [0, 1] \times [0, 1]$  with transformation

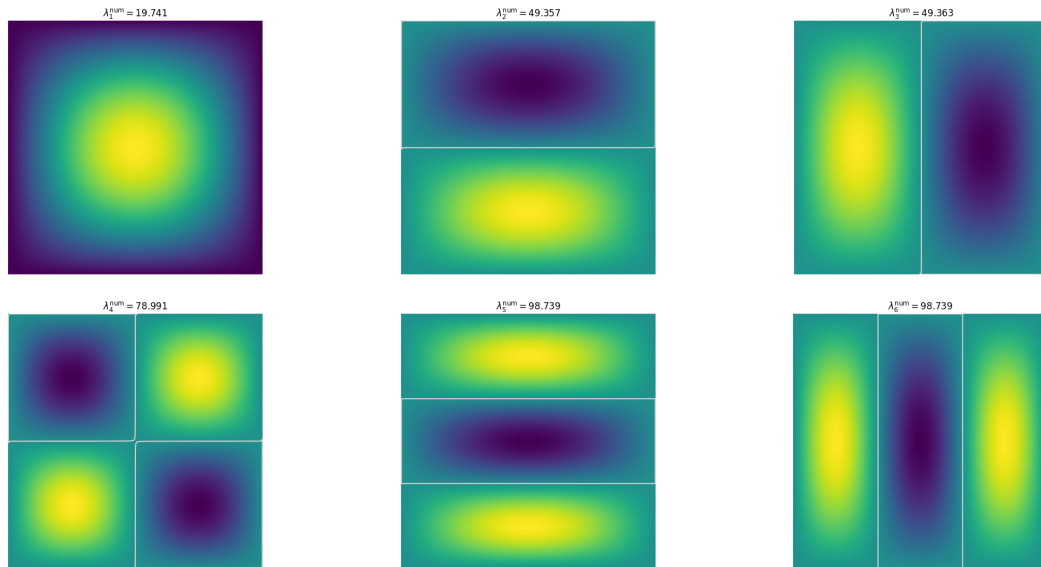


Figure 2.3: Dirichlet eigenvalue problem for the laplacian : numerical eigenvectors of the smallest magnitude eigenvalues after linear transformation

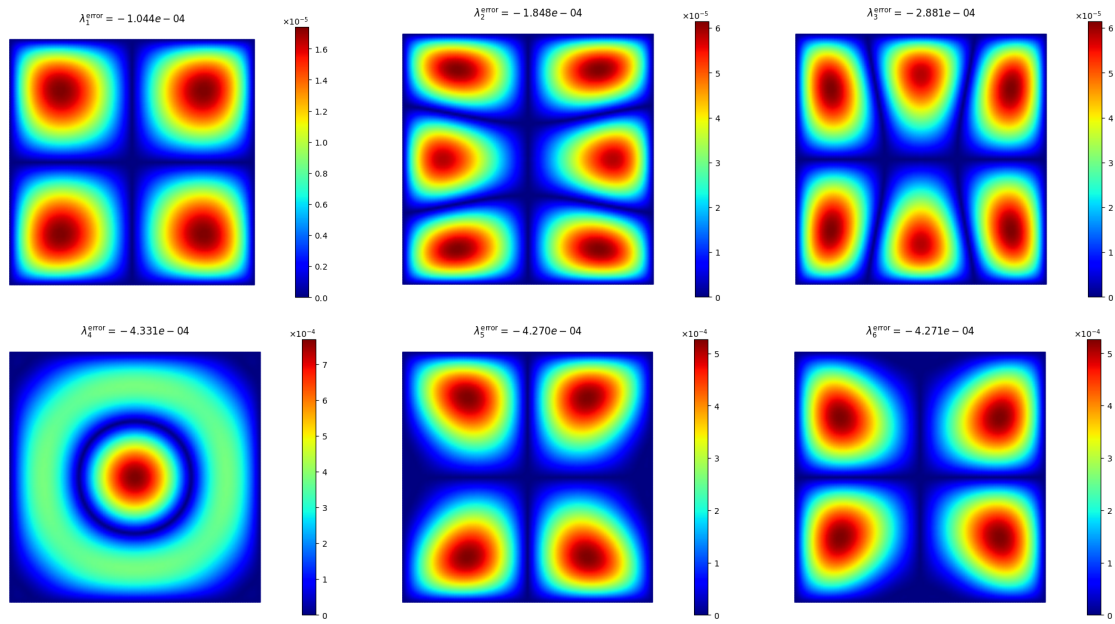


Figure 2.4: Dirichlet eigenvalue problem for the laplacian on unit square: relative errors

Now we test the `run` function on the rectangle  $[0, 3] \times [0, 2]$ . The python command used to compute the twelve first eigenvalues is the following:

```
from fc_vfemp1_eigs.examples.laplacian_DDDD_rectangle_ex
import run
res=run(k=12,L=3,H=2,N=250)
```

The exact eigenfunctions, the computed eigenfunctions and the relative errors are respectively represented in Figure 2.5, Figure 2.6 and Figure 2.7.

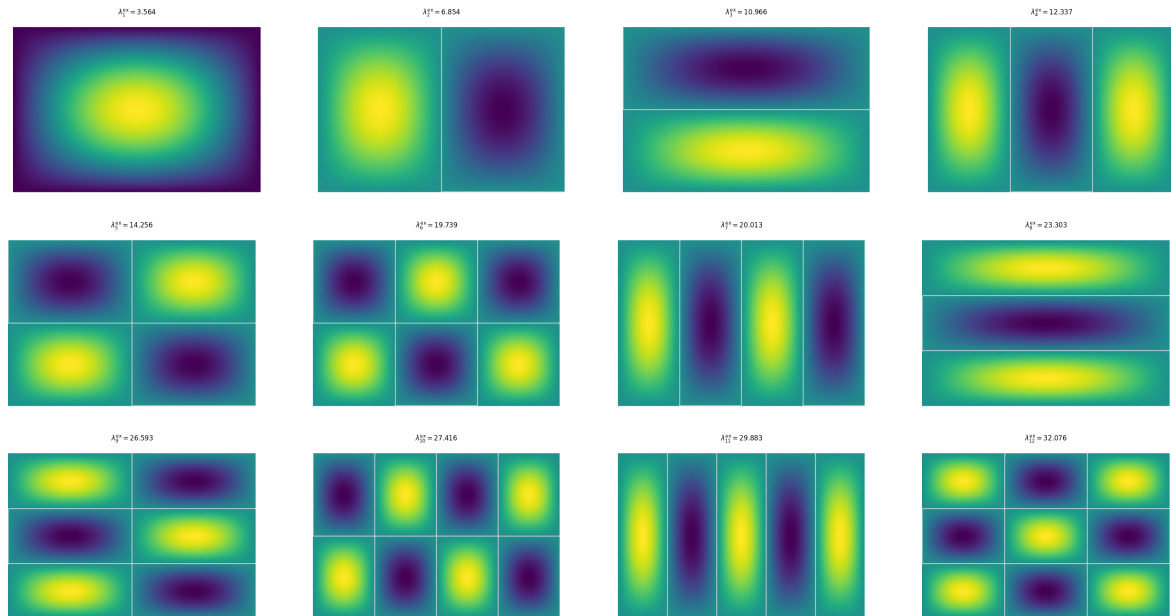


Figure 2.5: Dirichlet eigenvalue problem for the laplacian on  $[0, 3] \times [0, 2]$  : exact eigenfunctions of the smallest magnitude eigenvalues

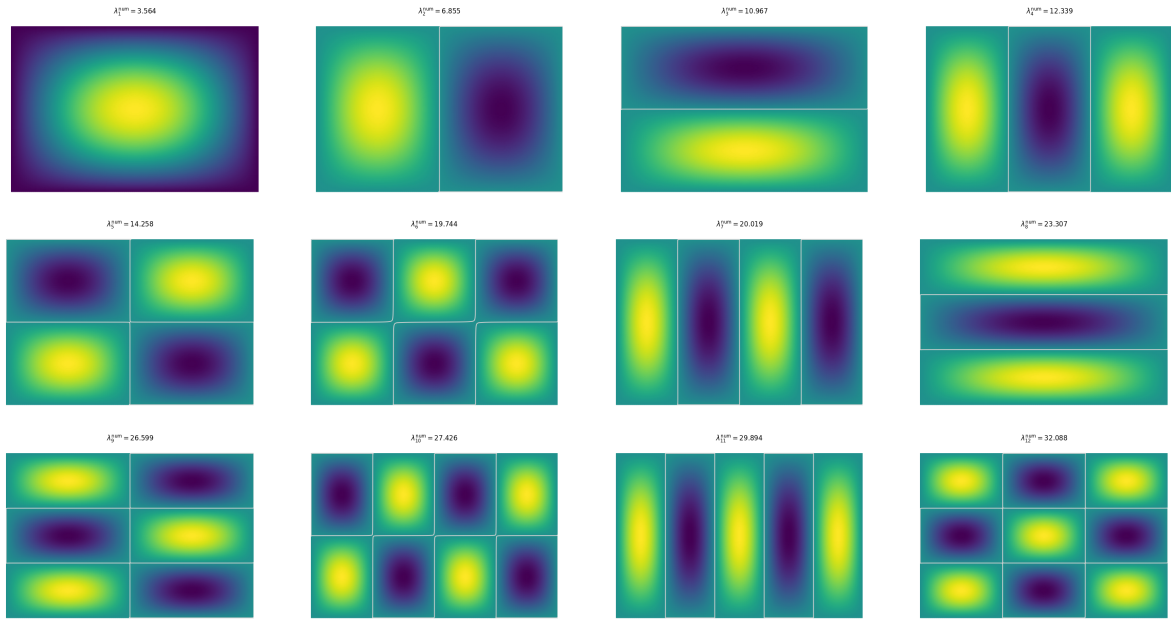


Figure 2.6: Dirichlet eigenvalue problem for the laplacian on  $[0, 3] \times [0, 2]$  : numerical eigenfunctions of the smallest magnitude eigenvalues

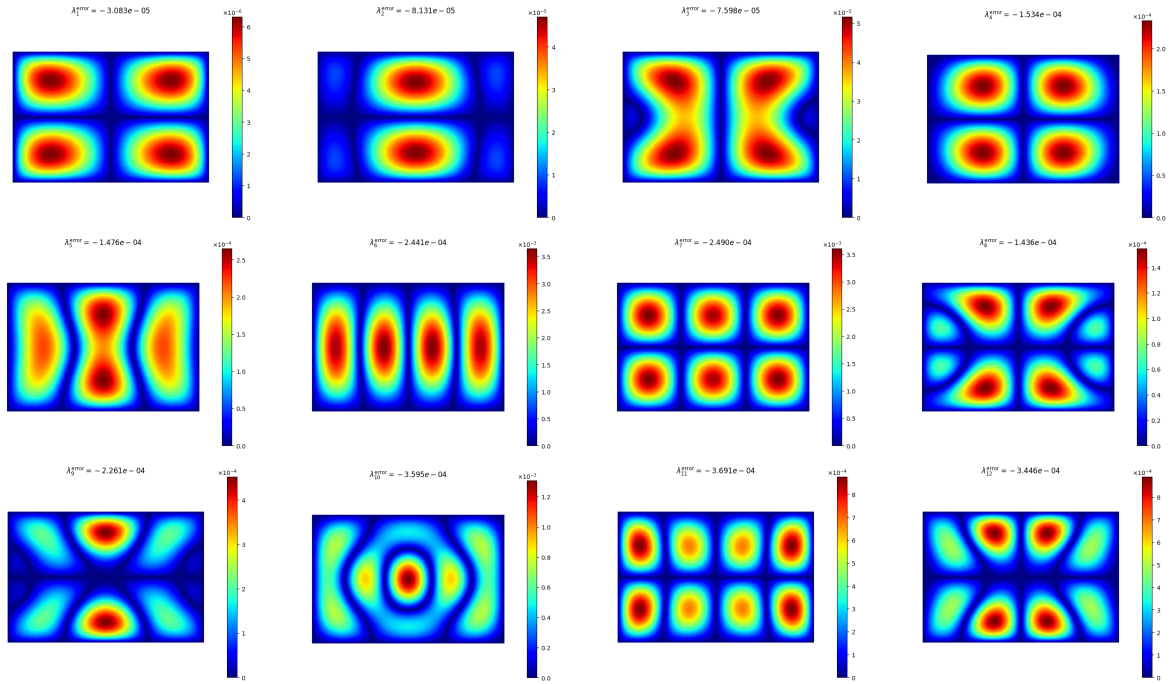


Figure 2.7: Dirichlet eigenvalue problem for the laplacian on  $[0, 3] \times [0, 2]$  : relative error of the eigenfunctions of the smallest magnitude eigenvalues

We represent in Figure 2.8 the order of convergence of the first ten eigenvalues.

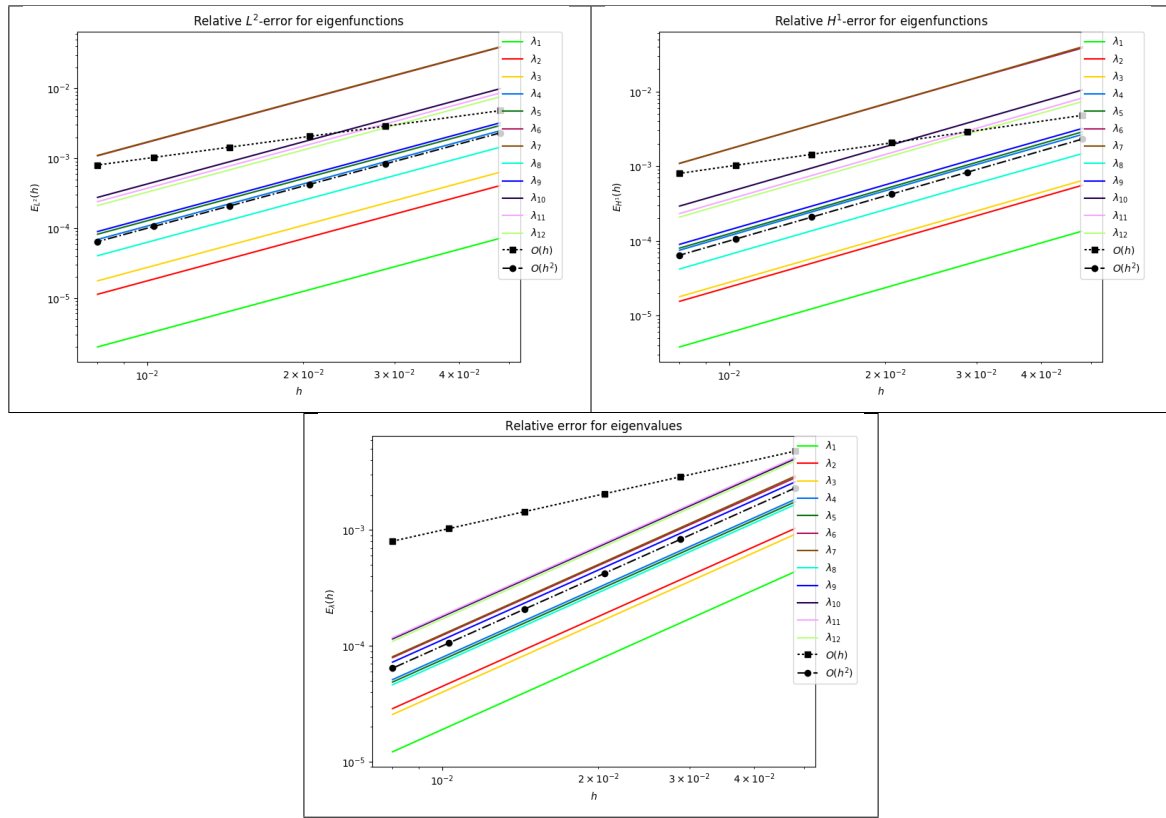


Figure 2.8: eigenvalues and eigenfunctions : order computation for Dirichlet eigens problem for the laplacian on rectangle  $[0, 3] \times [0, 2]$ (regular meshes) . Relative errors of eigenfunctions in  $L^2$ -norm (upper left) and  $H^1$ -norm (upper right). Relative errors of eigenvalues (bottom).

One can see that a superconvergence phenomena occurs due to regularity of the hypercube mesh. Indeed, for the  $H^1$ -norm an order 1 is expected. To highlight it, gmsh is now used to generate all the meshes of  $\Omega$  : results are given in Figure 2.9.

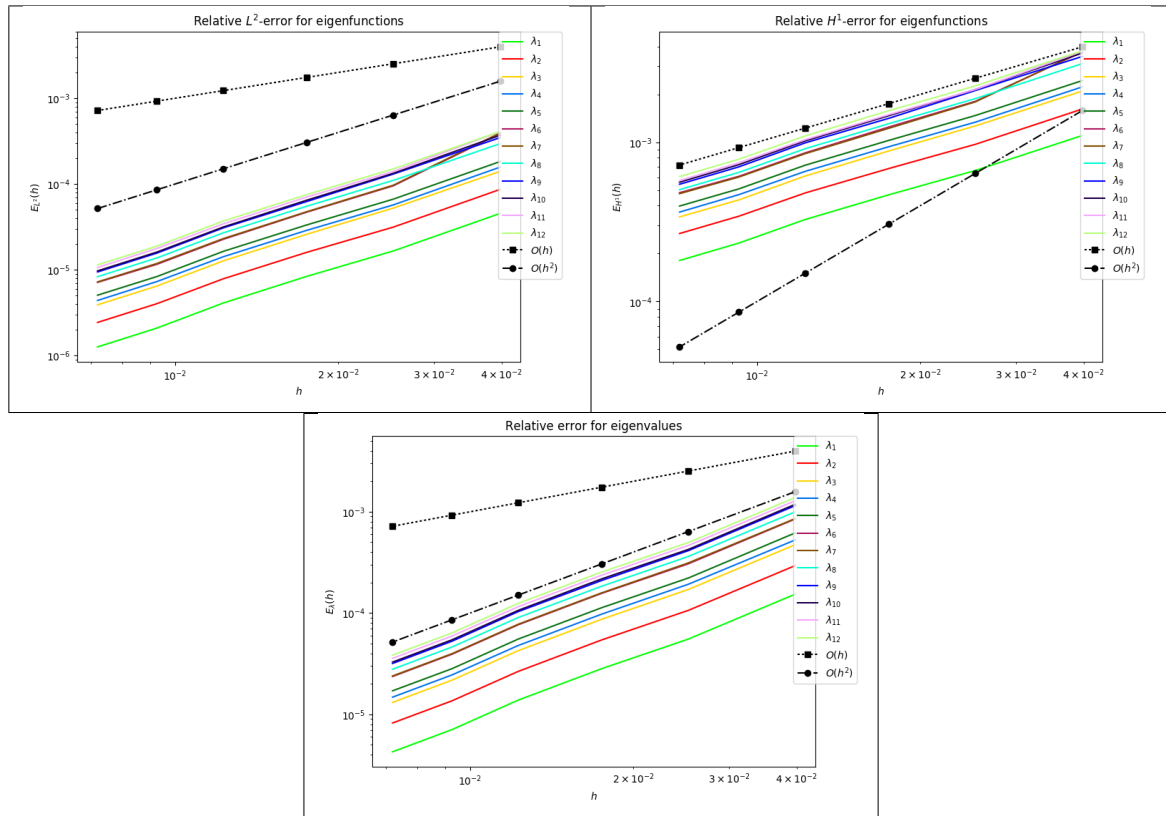


Figure 2.9: eigenvalues and eigenfunctions : order computation for Dirichlet eigens problem for the laplacian on rectangle  $[0, 3] \times [0, 2]$  (gmsh meshes) . Relative errors of eigenfunctions in  $L^2$ -norm (upper left) and  $H^1$ -norm (upper right). Relative errors of eigenvalues (bottom).

Just for fun, we now compute the nine eigenvalues near 200 and the corresponding eigenfunctions. In Figure 2.10 we represent the numerical solutions given by the following command:

```
from fc_vfemp1_eigs.examples.laplacian_DDDD_rectangle_ex
import run
res=run(k=9,sigma=200,L=3,H=2,N=350,
        gerror=False,gsolox=False,trans=False)
```

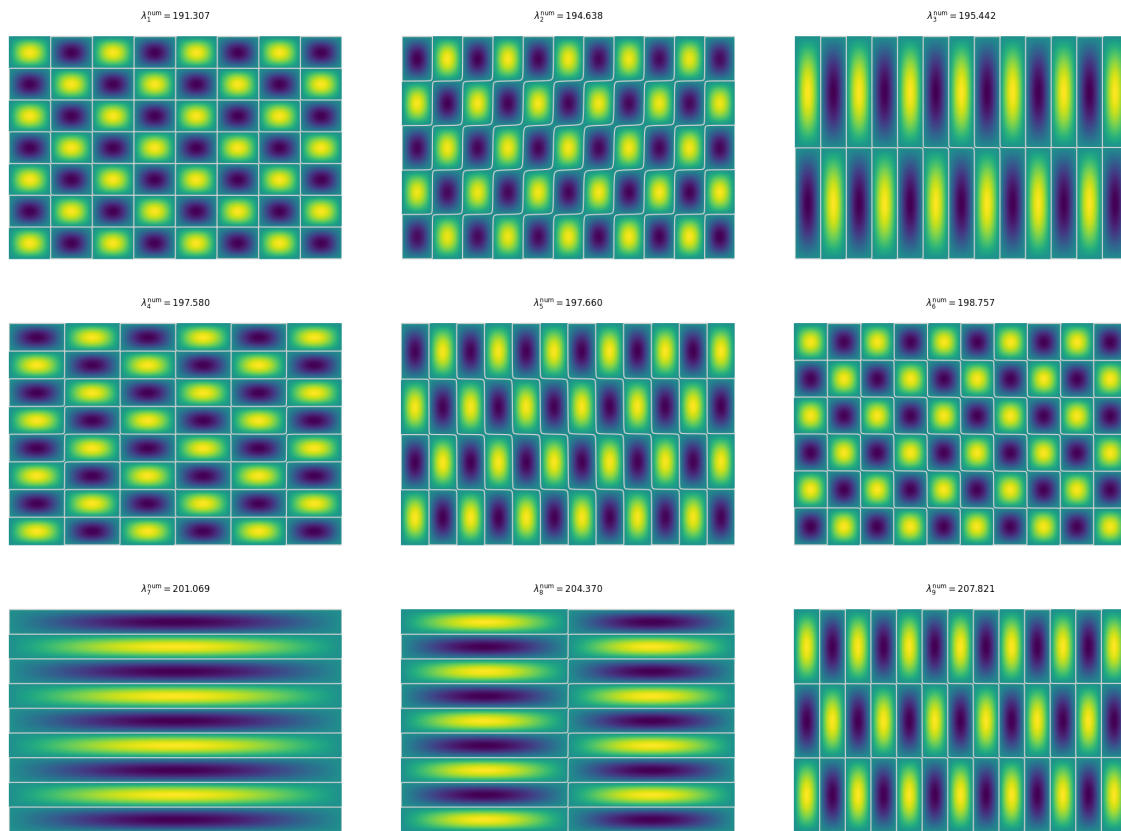


Figure 2.10: Dirichlet eigenvalue problem for the laplacian on  $[0, 3] \times [0, 2]$  : the nine eigenvalues near 200 and the corresponding eigenfunctions

### Application on the unit disk.

Let  $\Omega \subset \mathbb{R}^2$  be the unit disk meshed by gmsh and given in Figure 2.11.

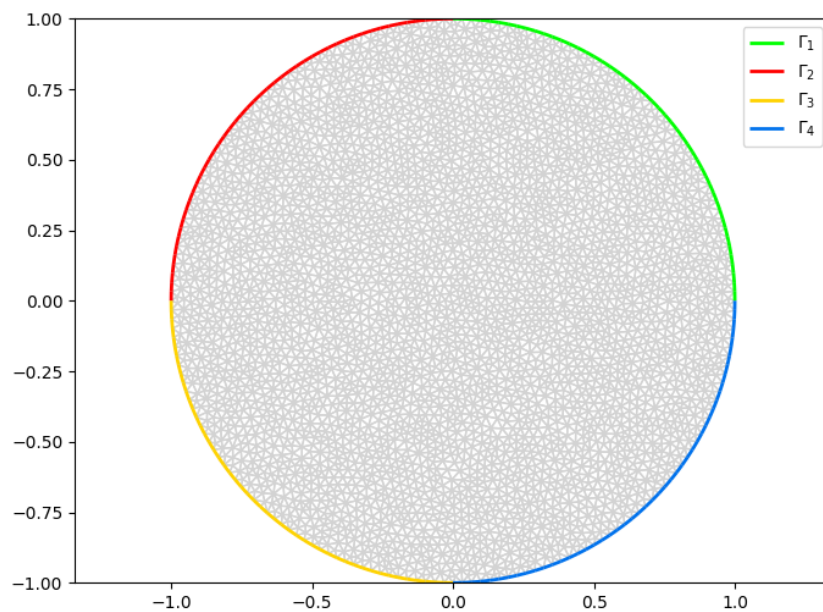


Figure 2.11: Unit disk with four boundaries

Let  $\alpha_{nl}$  be the  $l$ -th zero of the Bessel function of the first kind  $J_n$ . The eigenvalues are given by

$$\lambda_{n,l} = \alpha_{nl}^2 \quad \forall (n,l) \in \mathbb{N} \times \mathbb{N}^*$$

In Table 2.1, the values of  $\alpha_{nl}$  are given for  $(n,l) \in \llbracket 0, 4 \rrbracket \llbracket 1, 5 \rrbracket$ .

The eigenvalues are simple for  $n = 0$  and twice degenerate for  $n > 0$ .

$l$	$J_0$	$J_1$	$J_2$	$J_3$	$J_4$	$J_5$	$J_6$
1	2.4048256	3.8317060	5.1356223	6.3801619	7.5883424	8.7714838	9.9361095
2	5.5200781	7.0155867	8.4172441	9.7610231	11.064709	12.338604	13.589290
3	8.6537279	10.173468	11.619841	13.015201	14.372537	15.700174	17.003820
4	11.791534	13.323692	14.795952	16.223466	17.615966	18.980134	20.320789
5	14.930918	16.470630	17.959819	19.409415	20.826933	22.217800	23.586084
6	18.071064	19.615859	21.116997	22.582730	24.019020	25.430341	26.820152

Table 2.2: Zeros of the Bessel function of the first kind  $J_n$

$\lambda_1$	$\lambda_2$	$\lambda_3$	$\lambda_4$	$\lambda_5$	$\lambda_6$	$\lambda_7$	$\lambda_8$
5.7831860	14.681971	14.681971	26.374616	26.374616	30.471262	40.706466	40.706466
$\lambda_9$	$\lambda_{10}$	$\lambda_{11}$	$\lambda_{12}$	$\lambda_{13}$	$\lambda_{14}$	$\lambda_{15}$	$\lambda_{16}$
49.218456	49.218456	57.582941	57.582941	70.849999	70.849999	74.887007	76.938928
$\lambda_{17}$	$\lambda_{18}$	$\lambda_{19}$	$\lambda_{20}$	$\lambda_{21}$	$\lambda_{22}$	$\lambda_{23}$	$\lambda_{24}$
76.938928	95.277573	95.277573	98.726272	98.726272	103.49945	103.49945	122.42780

Table 2.3: twenty four first eigenvalues

We represent in Figure 2.12 eigenvectors associated to the first twenty-four smallest magnitude eigenvalues.

We represent in Figure 2.13 eigenvectors associated to the twenty-four eigenvalues near 1000.

**Application on the L-shape domain.** The *Lshape* domain  $\Omega$  meshed by gmsh is given in Figure 2.14.

Part of the source code (file `fc_vfemp1_eigs/examples/Laplacian_Dir_Lshape_01.m`) is given in Listing 2.4

```
print('***_Setting_the_mesh_using_gmsh_and_%s_file'%geofile)
meshfile=gmsh.buildmesh2d(geodir+os.sep+geofile+'.geo',N, force=True, verbose=0)
print('***_Setting_the_eBVP')
Lop=Loperator(dim=2,A=[[1,None],[None,1]])
pde=PDE(Op=Lop)
bvp=BVP(Th,pde=pde)
for lab in Th.sThlab[Th.find(d=1)]:
print('***_Solving_the_eBVP')
```

Listing 2.4: 2D Laplace eigenvalues problem with Dirichlet boundary condition on *Lshape* domain

Results can be found in [11], Figure 1 page 4 and [7]. From [8] section 6.52 page 122 or [12] Table 1 page 1088, we have the bounds to the first ten eigenvalues of the L-shaped Laplacian problem is given Table 2.4. We also give the computed values from a L-shaped mesh with  $n_q = 89780$ ,  $n_{me} = 178358$  and  $h \approx 0.010459932939$ .

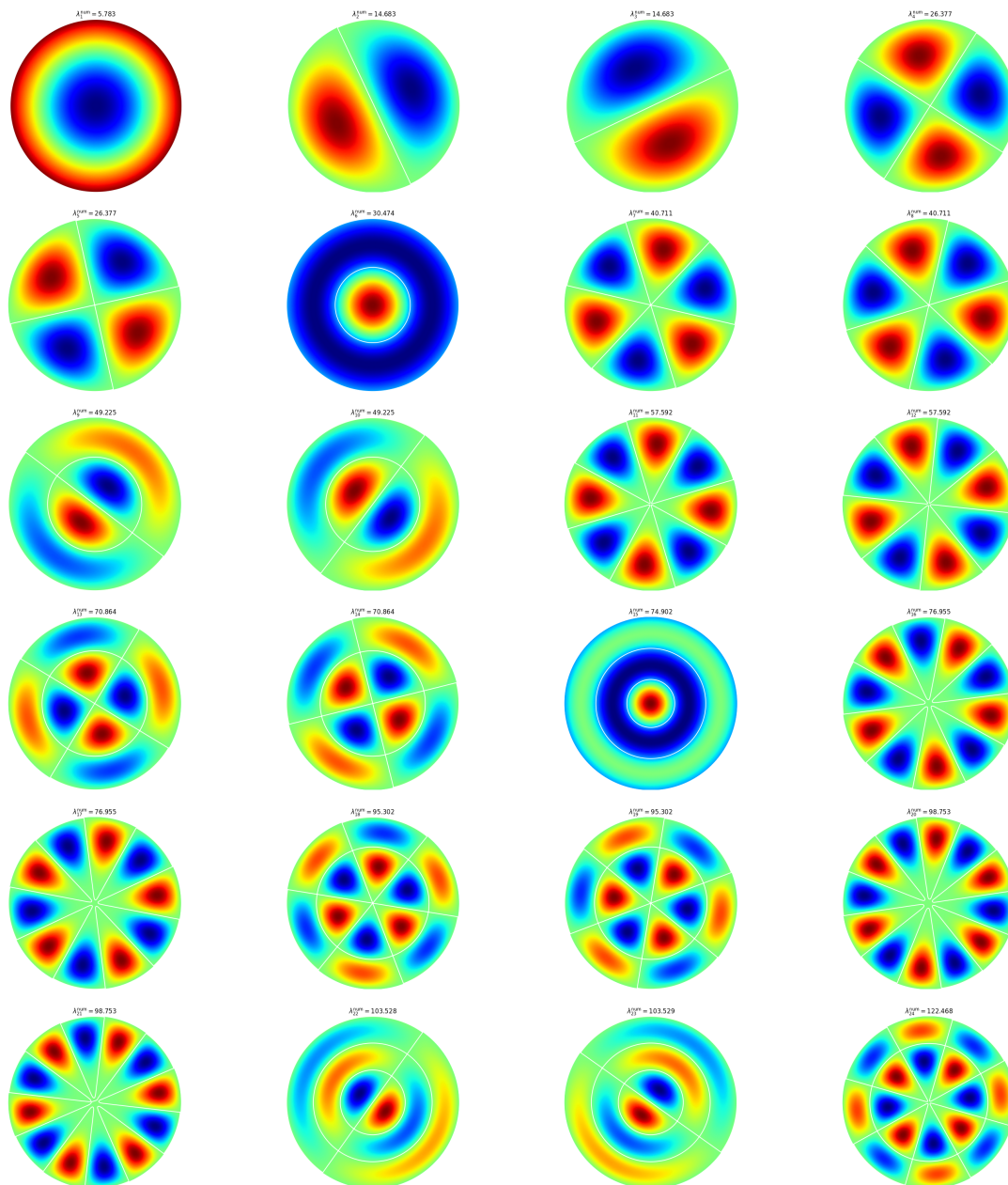


Figure 2.12: Dirichlet eigenvalue problem on the unit disk : eigenvectors of the smallest magnitude eigenvalues

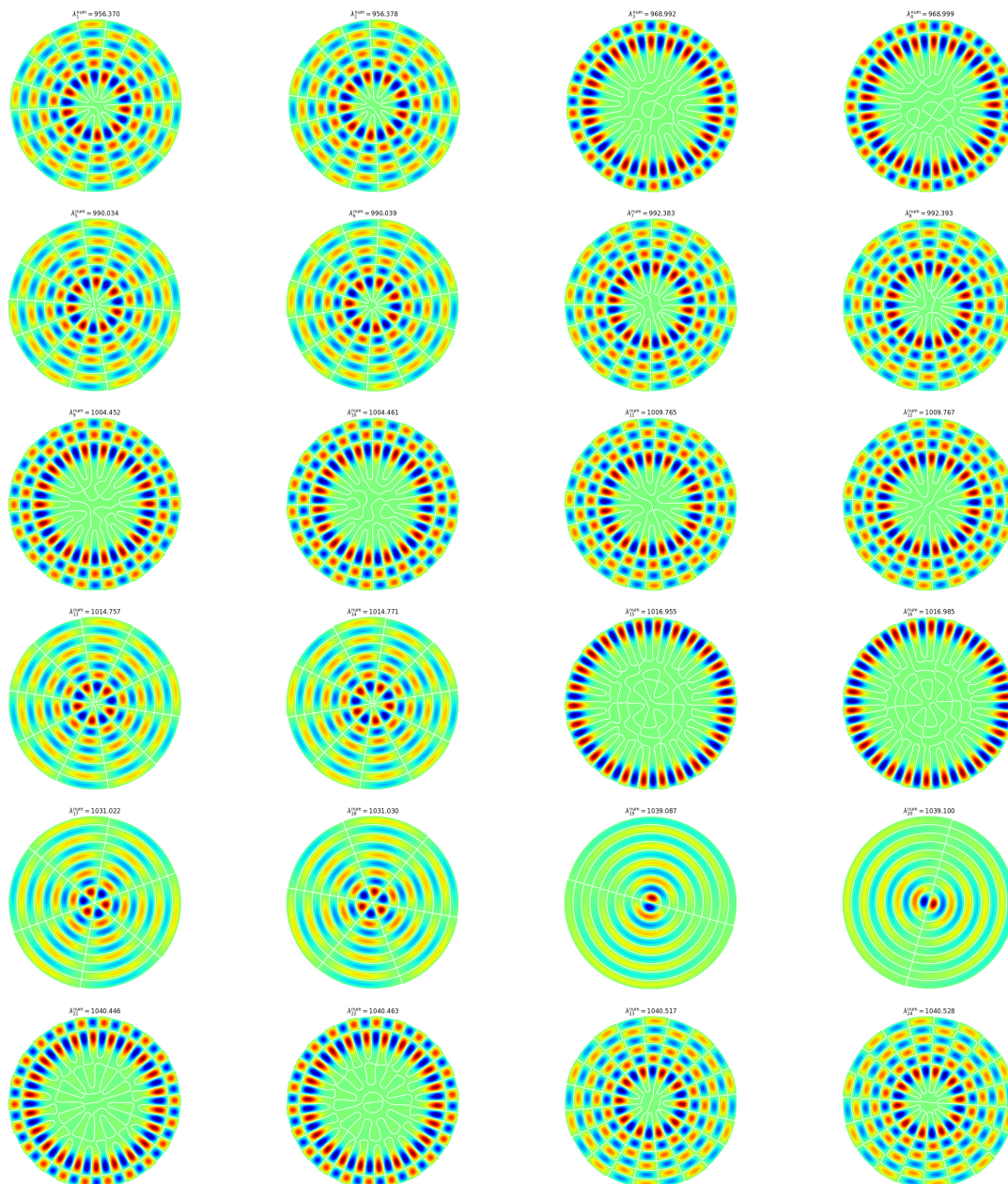
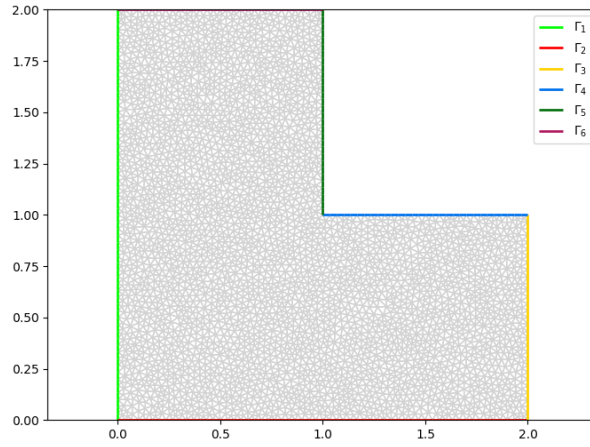


Figure 2.13: Dirichlet eigenvalue problem on the unit disk : eigenvectors of the eigenvalues near 1000

Figure 2.14: *Lshape* domain with four boundaries

$n$	bounds of $\lambda_n$ from [12]	computed here
1	$9.63972384_{04}^{44}$	9.64209719683
2	$15.19725192_{59}^{66}$	15.1978811701
3	$2\pi^2 = 19.739208802178$	19.740276353
4	$29.52148111_{38}^{42}$	29.523864337
5	$31.9126359_{37}^{59}$	31.9205911428
6	$41.4745098_{66}^{92}$	41.4831105006
7	$44.9484877_{77}^{82}$	44.9540109819
8	$5\pi^2 = 49.34802200544$	49.3546645871
9	$5\pi^2 = 49.34802200544$	49.3547009123
10	$56.7096098_{18}^{90}$	56.7227438731

Table 2.4: Bounds to the first ten eigenvalues of the L-shaped Laplacian problem

We represent in Figure 2.15 eigenvectors associated to the first twenty-four smallest magnitude eigenvalues.

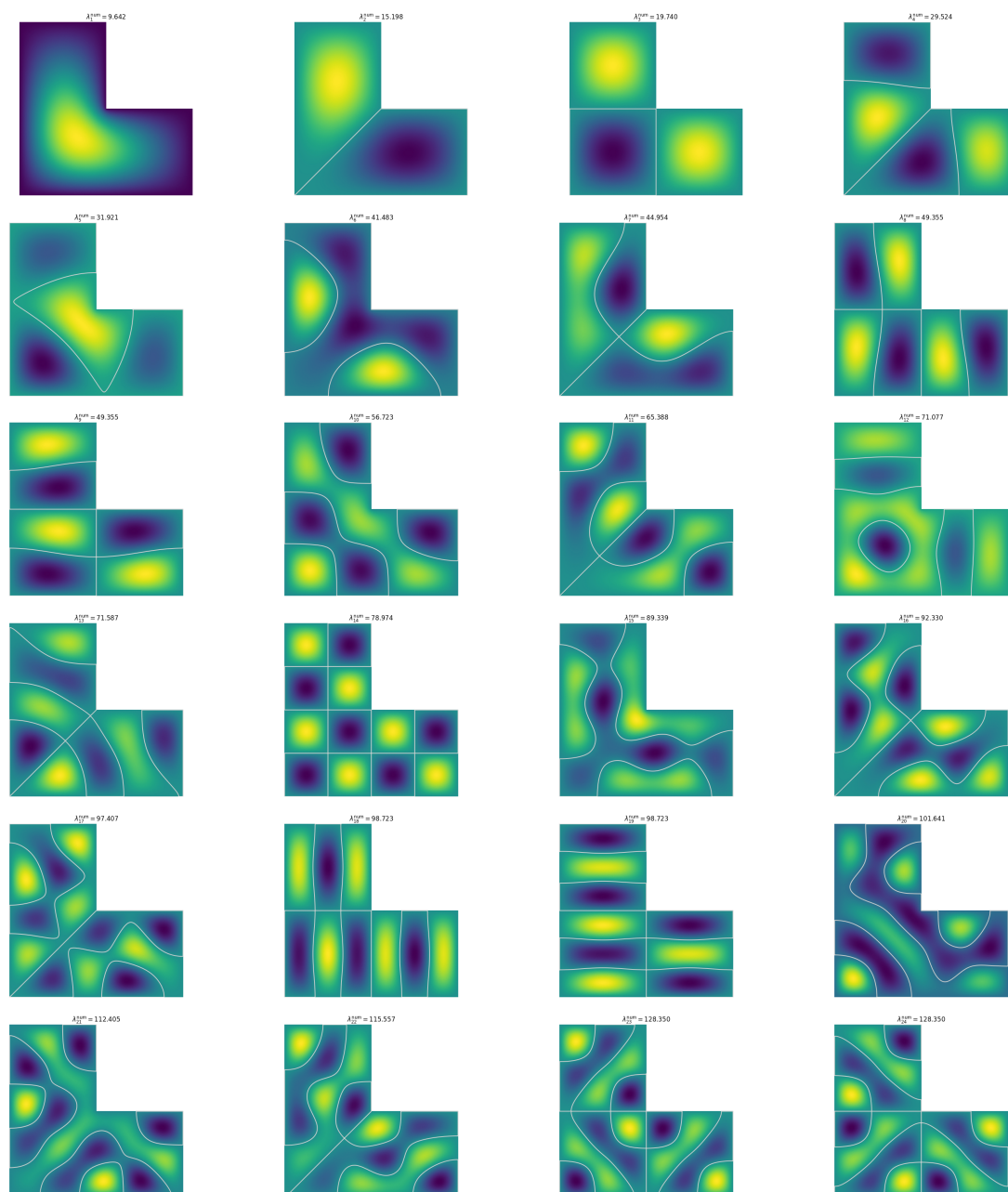


Figure 2.15: 2D Laplace in L-shaped domain with Dirichlet boundary conditions : eigenvectors of the smallest magnitude eigenvalues

In Figure 2.16 the eigenvectors associated with the four eigenvalues nearest 250 (multiplicity 1) and 493 (multiplicity 3) are represented. This is done by setting *sigma* option to 250 for the first case and to 493 for the second one.

## 2.2.2 2D Laplace eigenvalues problem with mixed boundary conditions

We want to solve the eigenvalue problem given by (2.7)-(2.10)).

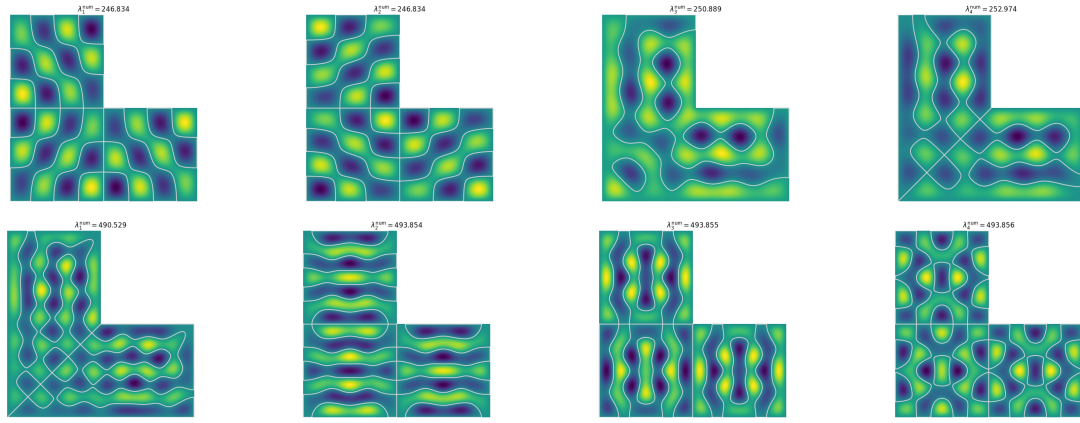


Figure 2.16: 2D Laplace with Dirichlet boundary conditions : eigenvectors of the eigenvalues near  $\lambda_{50} = 250.78548$  (multiplicity 1) and  $\lambda_{104} = 493.48022$  (multiplicity 3)



### Usual EBVP 2 : 2D Laplace with mixed boundary condition



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

$$-\Delta u = \lambda u \quad \text{in } \Omega, \quad (2.7)$$

$$\frac{\partial u}{\partial n} + \alpha u = 0 \quad \text{on } \Gamma^a, \quad (2.8)$$

$$\frac{\partial u}{\partial n} = 0 \quad \text{on } \Gamma^b, \quad (2.9)$$

$$u = 0 \quad \text{on } \Gamma^c, \quad (2.10)$$

The problem (2.7)-(2.10) can be equivalently written as the *Scalar* EBVP 1:



### Scalar EBVP 3 : 2D Laplace with mixed boundary condition



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

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

$$\frac{\partial u}{\partial n_{\mathcal{L}}} + a^R u = 0 \quad \text{on } \Gamma^R = \Gamma^a \cup \Gamma^b,$$

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

where  $\mathcal{L} = \mathcal{L}_{1,0,0,0}$  (and then  $\frac{\partial u}{\partial n_{\mathcal{L}}} = \frac{\partial u}{\partial n}$ ),  $\mathcal{B} = \mathcal{L}_{0,0,0,1}$ ,  $a^R = \alpha \delta_{\Gamma^b}$

**Application on the disk with 5 holes domain.** Let  $\Gamma_1$  be the unit disk,  $\Gamma_{10}$  be the disk with center point  $(0,0)$  and radius 0.3. Let  $\Gamma_{20}$ ,  $\Gamma_{21}$ ,  $\Gamma_{22}$  and  $\Gamma_{23}$  be the disks with radius 0.1 and respectively with center point  $(0, -0.7)$ ,  $(0, 0.7)$ ,  $(-0.7, 0)$  and  $(0.7, 0)$ . The domain  $\Omega \subset \mathbb{R}^2$  is defined as the inner of  $\Gamma_1$  and the outer of all other disks (see Figure 2.17).

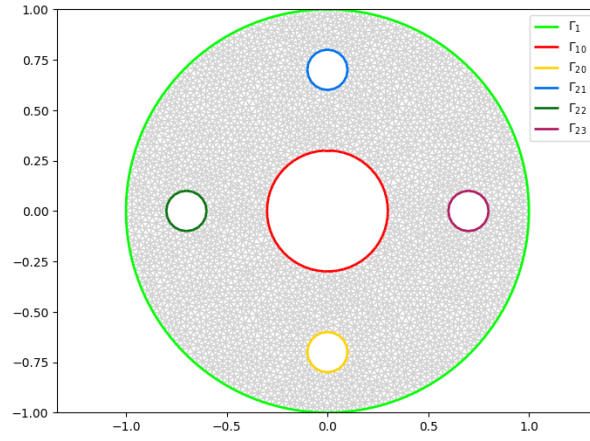


Figure 2.17: Domain and boundaries

We want to solve the eigenvalue problem given by (2.11)-(2.14).

#### **Scalar EBVP 4 : 2D Laplace eigenvalues problem with mixed boundary conditions**

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

$$-\Delta u = \lambda u \quad \text{in } \Omega, \quad (2.11)$$

$$\frac{\partial u}{\partial n} + 10u = 0 \quad \text{on } \Gamma_{22} \cup \Gamma_{23}. \quad (2.12)$$

$$\frac{\partial u}{\partial n} = 0 \quad \text{on } \Gamma_{20} \cup \Gamma_{21}, \quad (2.13)$$

$$u = 0 \quad \text{on } \Gamma_1 \cup \Gamma_{10}. \quad (2.14)$$

So we have,  $\Gamma^D = \Gamma_1 \cup \Gamma_{10}$ ,  $\Gamma^R = \bigcup_{i=20}^{23} \Gamma_i$ , and  $a^R = 10\delta_{\Gamma_{22} \cup \Gamma_{23}}$ .

We give in Listing 2.5 the corresponding Matlab code.

Listing 2.5: 2D Laplacian eigenvalues problem with mixed boundary conditions on a domain with 5 holes

```
print('Mesh sizes : nq=%6d, nme=%7d, \n
      h=%3e'%(Th.nq,Th.get_nme(),Th.get_h()))
pde=PDE(Op=Lop)
bvp=BVP(Th,pde=pde)
bvp.setDirichlet(1,0.)
bvp.setDirichlet(10,0.)
bvp.setRobin(20,0.)
bvp.setRobin(21,0.)
bvp.setRobin(22,0.,ar=10)
bvp.setRobin(23,0.,ar=10)
```

We represent in Figure 2.18 the twenty four first eigenvectors.

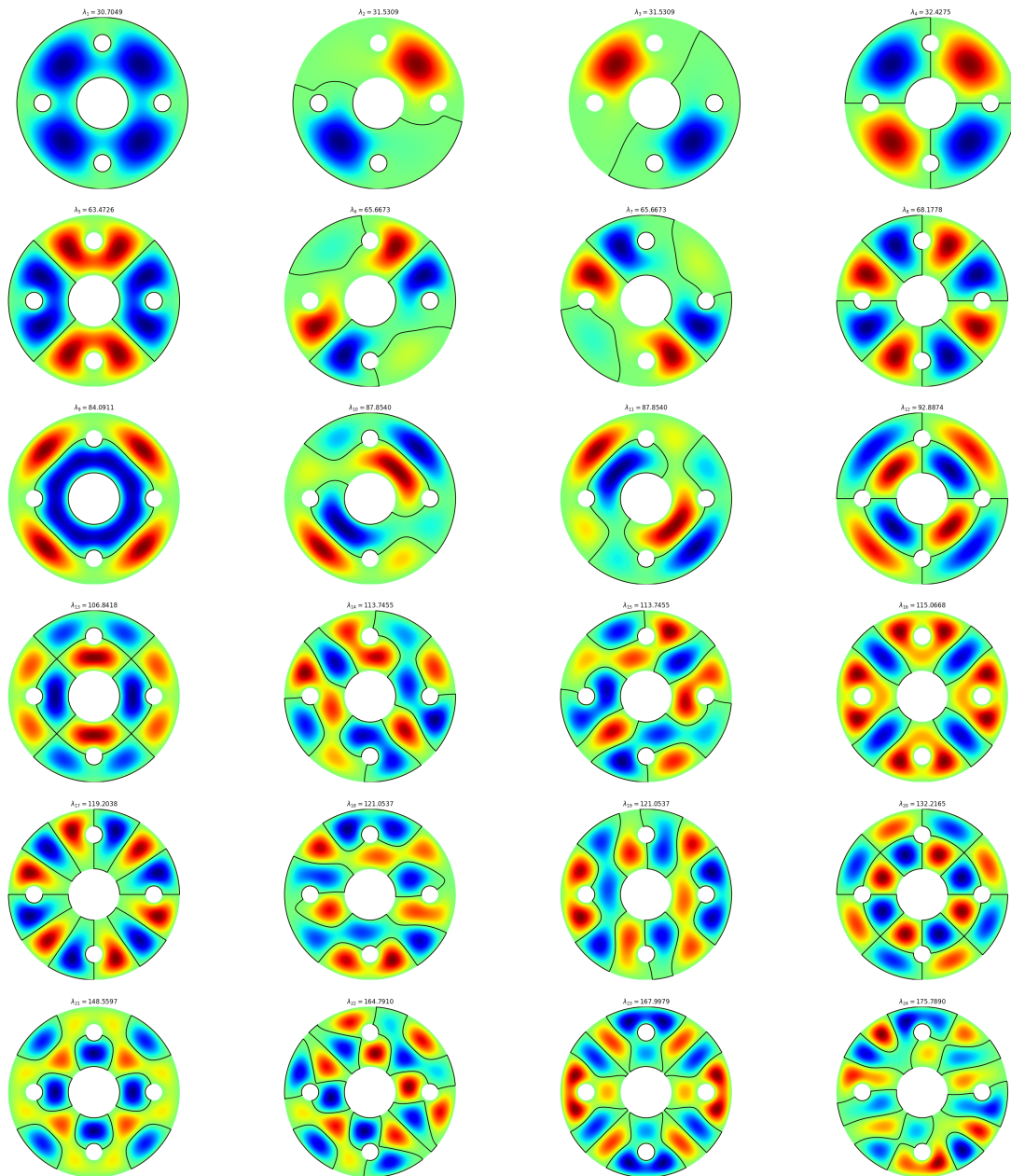



Figure 2.18: 2D Laplace with mixed boundary conditions : eigenvectors of the smallest magnitude eigenvalues

### 2.2.3 Other 2D eigenvalues problems with Dirichlet boundary condition

#### Convection-Diffusion on the L-shaped domain.

We want to solve the eigenvalue problem given by

 **Usual EBVP 3 : 2D Convection-Diffusion eigenvalues problem with Dirichlet boundary condition**

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

$$\begin{aligned} -\Delta u + \beta \cdot \nabla u &= \lambda u && \text{in } \Omega, \\ u &= 0 && \text{on } \Gamma. \end{aligned}$$

with constant convection parameter  $\beta \in \mathbb{R}^2$ .

From [8] section 6.52 page 122 the eigenvalues of Usual EBVP 3 are  $\lambda_i^\beta = |\beta|/4 + \lambda_i$  where  $\lambda_i$  are the eigenvalues of the L-shaped Laplacian problem with Dirichlet boundary condition (the ten first are given in Table 2.4). We have for example

$$\begin{aligned} \lambda_1^\beta &\approx |\beta|/4 + 9.63972, & \lambda_3^\beta &= |\beta|/4 + 2\pi^2 \\ \lambda_5^\beta &\approx |\beta|/4 + 31.912636, & \lambda_8^\beta &= \lambda_9^\beta = |\beta|/4 + 5\pi^2 \\ \lambda_{20}^\beta &\approx |\beta|/4 + 101.60529, & \lambda_{50}^\beta &\approx |\beta|/4 + 250.78548. \end{aligned}$$

We give in Listing 2.6 the corresponding Python code.

Listing 2.6: 2D L-shaped Convection-Diffusion problem with  $\beta = (3, 0)$  : Python code

```
print('Mesh sizes : nq=%6d, nme=%7d, \n
      h=%.3e'%(Th.nq,Th.get_nme(),Th.get_h()))
pde=PDE(Op=Lop)
bvp=BVP(Th,pde=pde)
for lab in Th.sThlab[Th.find(d=1)]:
    bvp.setDirichlet(lab,0.)
```

We give the computed values from a L-shaped mesh with  $n_q = 89780$ ,  $n_{me} = 178358$  and  $h \approx 0.0105$ . We represent in Figure 2.18 the twenty four first eigenvectors with  $\beta = (3, 0)$ .

$n$	bounds of $\lambda_n^\beta$ from [12]	computed here
1	$ \beta /4 + 9.63972384_{04}^{44} \approx 11.88972384$	11.891874
2	$ \beta /4 + 15.19725192_{59}^{66} \approx 17.44725192$	17.447521
3	$ \beta /4 + 2\pi^2 = 21.989208802178716$	21.989805
4	$ \beta /4 + 29.52148111_{38}^{42} \approx 31.77148111$	31.773153
5	$ \beta /4 + 31.9126359_{37}^{59} \approx 34.1626359$	34.169821
6	$ \beta /4 + 41.4745098_{66}^{92} \approx 43.7245099$	43.732105
7	$ \beta /4 + 44.9484877_{77}^{82} \approx 47.19848777$	47.202920
8	$ \beta /4 + 5\pi^2 = 51.598022005446794$	51.603105
9	$ \beta /4 + 5\pi^2 = 51.598022005446794$	51.603864
10	$ \beta /4 + 56.7096098_{18}^{90} \approx 58.9596098$	58.971362
20	$ \beta /4 + 101.60529 \approx 103.85529$	103.88829
50	$ \beta /4 + 250.78548 \approx 253.03548$	253.21186

Table 2.5: Eigenvalues of the L-shaped Convection-Diffusion problem with  $\beta = (3, 0)$ .

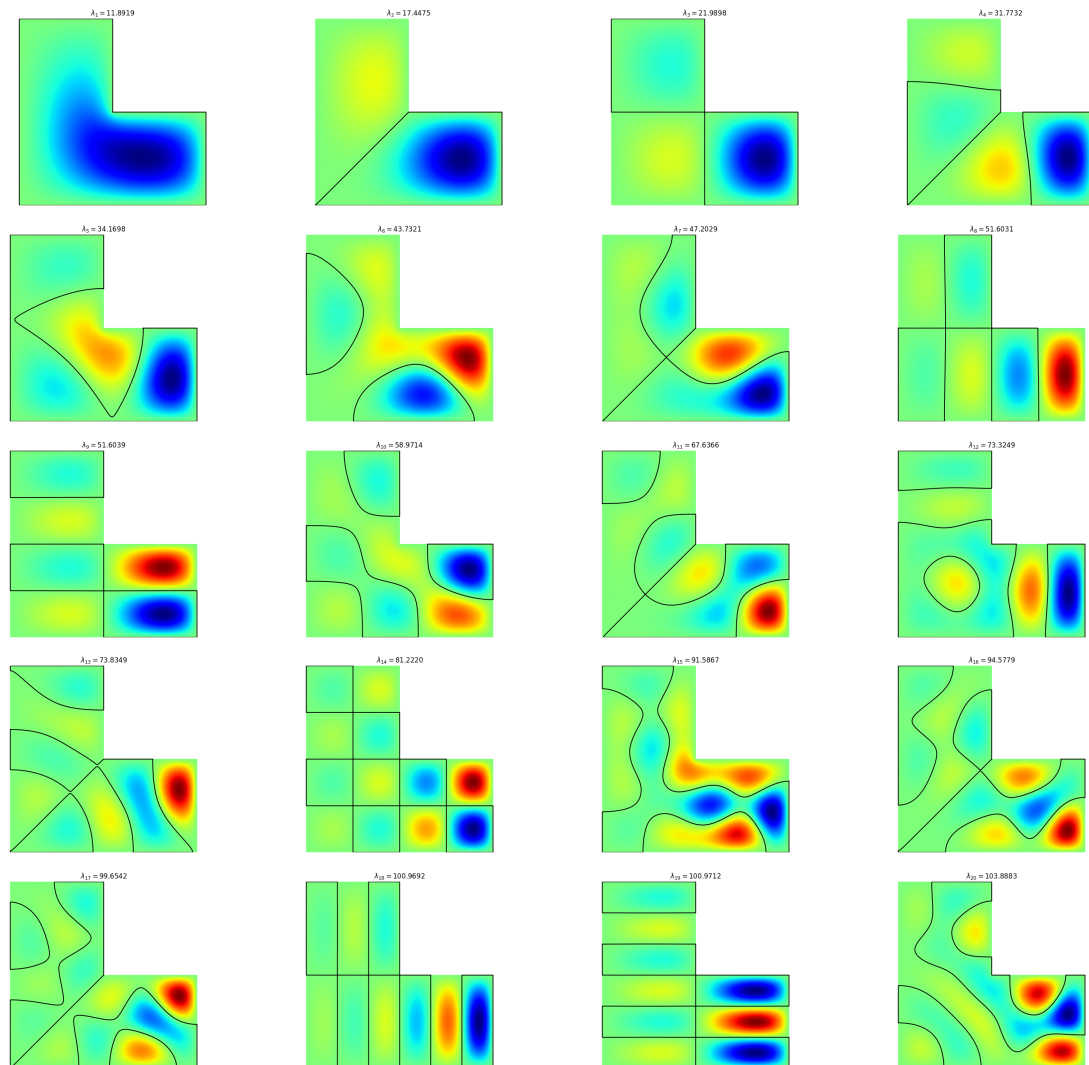


Figure 2.19: 2D L-shaped Convection-Diffusion problem with  $\beta = (3, 0)$ . Eigenvectors of the smallest magnitude eigenvalues ( $n_q = 89780$ )

## 2.3 3D examples

### 2.3.1 Eigenvalues of the laplacian on the unit sphere with Dirichlet boundary condition

**Application on the unit sphere.**

Let  $\Omega \subset \mathbb{R}^3$  be the unit sphere meshed by gmsh and given in Figure 2.20.

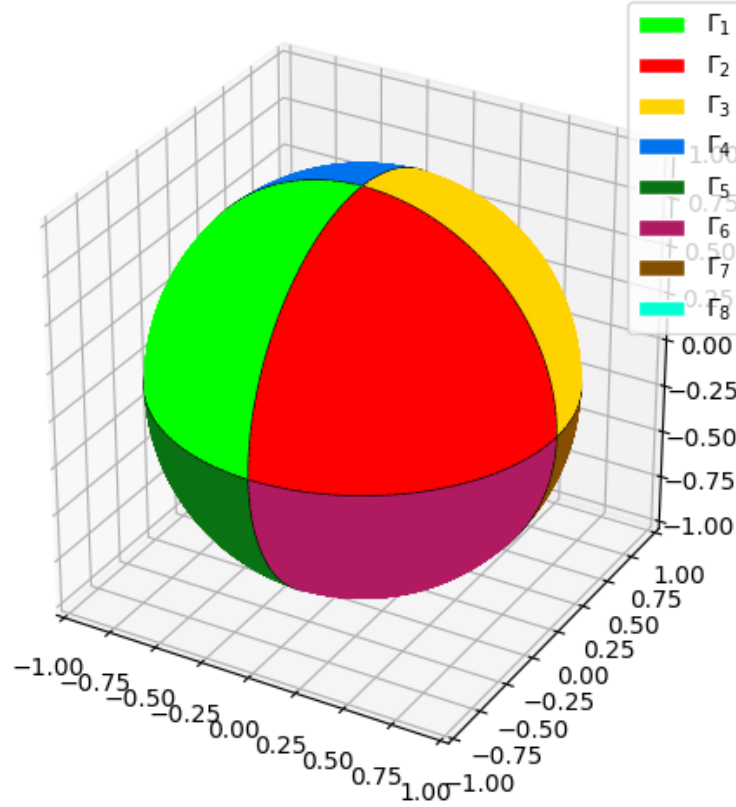


Figure 2.20: Unit sphere with eight boundaries

From [9], page 609, we get the eigenvalues of the laplacian on the sphere with Dirichlet boundary conditions. Let  $\alpha_{nk}$  be the  $k$ -th zero of the Bessel function of the first kind  $J_{n+1/2}$ . The eigenvalues are given by

$$\lambda_{n,k} = \alpha_{nk}^2 \quad \forall n \in \mathbb{N}, \quad \forall k \in \mathbb{N}^*$$

In Table 2.6, the values of  $\alpha_{nk}$  are given for  $(n, k) \in \llbracket 0, 4 \rrbracket \llbracket 1, 5 \rrbracket$ . The eigenvalues have the degeneracy  $2n + 1$ .

$l$	$J_{1/2}$	$J_{1+1/2}$	$J_{2+1/2}$	$J_{3+1/2}$	$J_{4+1/2}$	$J_{5+1/2}$	$J_{6+1/2}$
1	3.1415927	4.4934095	5.7634592	6.9879320	8.1825615	9.3558121	10.512835
2	6.2831853	7.7252518	9.0950113	10.417119	11.704907	12.966530	14.207392
3	9.4247780	10.904122	12.322941	13.698023	15.039665	16.354710	17.647975
4	12.566371	14.066194	15.514603	16.923621	18.301256	19.653152	20.983463
5	15.707963	17.220755	18.689036	20.121806	21.525418	22.904551	24.262768
6	18.849556	20.371303	21.853874	23.304247	24.727566	26.127750	27.507868

Table 2.6: Zeros of the Bessel function of the first kind  $J_{n+1/2}$

	<b>exact</b>	<b>numerical</b>	<b>rel. error</b>
$\lambda_1$	9.8696044	9.8795733	0.0010100591
$\lambda_2$	20.190729	20.241845	0.0025316829
$\lambda_3$	20.190729	20.241972	0.0025379680
$\lambda_4$	20.190729	20.242068	0.0025427072
$\lambda_5$	33.217462	33.366667	0.0044917671
$\lambda_6$	33.217462	33.367653	0.0045214645
$\lambda_7$	33.217462	33.367833	0.0045268671
$\lambda_8$	33.217462	33.368352	0.0045424835
$\lambda_9$	33.217462	33.369297	0.0045709448
$\lambda_{10}$	39.478418	39.683576	0.0051967347
$\lambda_{11}$	48.831194	49.167078	0.0068784726
$\lambda_{12}$	48.831194	49.168061	0.0068986160
$\lambda_{13}$	48.831194	49.169862	0.0069354936
$\lambda_{14}$	48.831194	49.170425	0.0069470166
$\lambda_{15}$	48.831194	49.171088	0.0069606076
$\lambda_{16}$	48.831194	49.173549	0.0070110001
$\lambda_{17}$	48.831194	49.173957	0.0070193439
$\lambda_{18}$	59.679516	60.162479	0.0080926014
$\lambda_{19}$	59.679516	60.169330	0.0082074079
$\lambda_{20}$	59.679516	60.170223	0.0082223667
$\lambda_{21}$	66.954312	67.603193	0.0096913974
$\lambda_{22}$	66.954312	67.605029	0.0097188145
$\lambda_{23}$	66.954312	67.605519	0.0097261446
$\lambda_{24}$	66.954312	67.608407	0.0097692707

Table 2.7: Comparison between the twenty four first exact eigenvalues and numerical computation on mesh with  $n_q = 81324$  and  $n_{me} = 462596$

We represent in Figure 2.21 eigenvectors associated to the first twenty-four smallest magnitude eigenvalues.

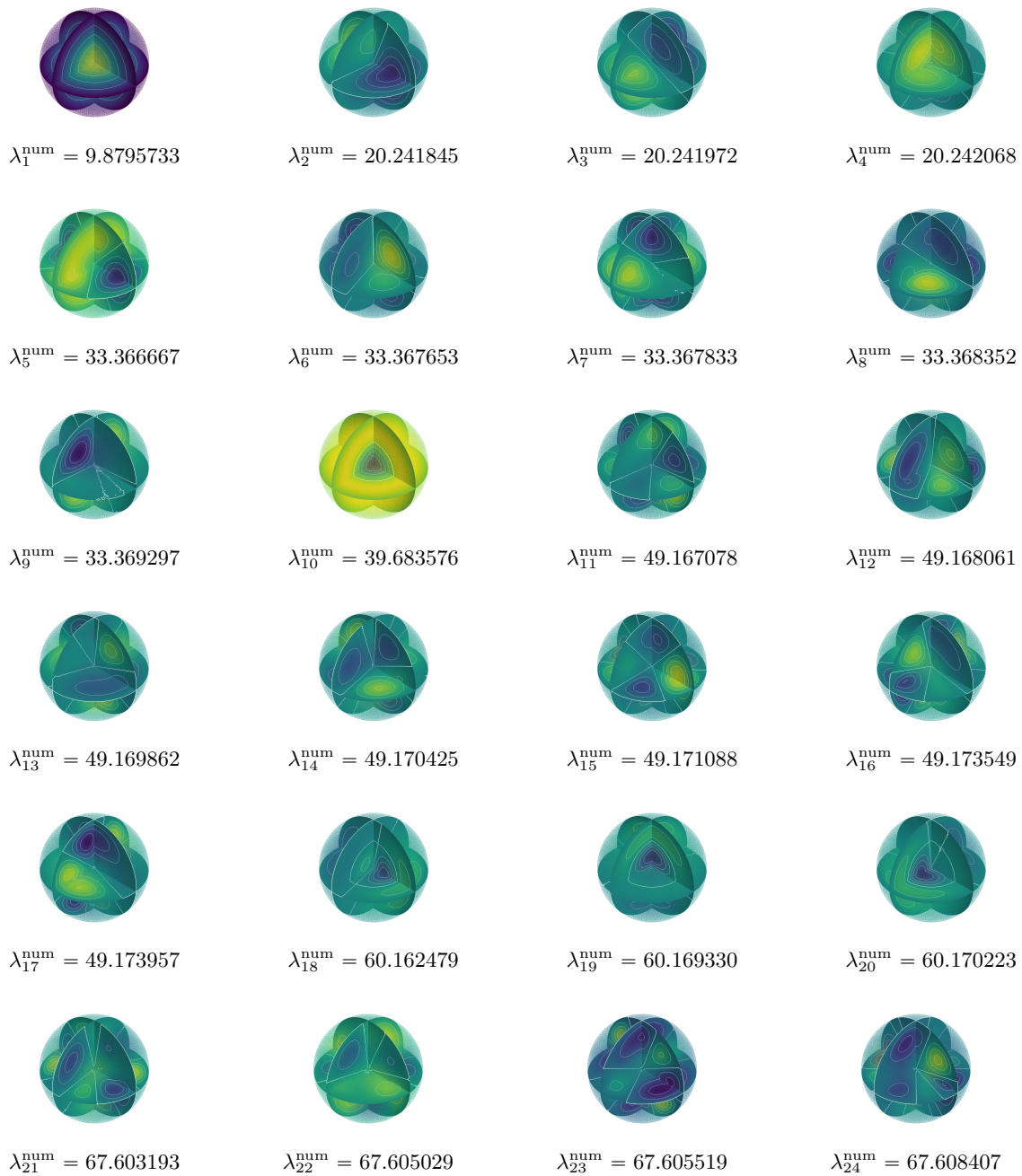


Figure 2.21: Dirichlet eigenvalue problem on the unit sphere : eigenvectors of the smallest magnitude eigenvalues

# Chapter 3

## Generalized Eigenvalue vector BVP

The eigenvalue problems associated with *vector* BVP (1.14)-(1.16) can be written as



### Vector EBVP 1 : generic problem



Find  $\lambda \in \mathbb{K}$  and  $\mathbf{u} = (\mathbf{u}_1, \dots, \mathbf{u}_m) \in (H^2(\Omega))^m$  such that

$$\mathcal{H}(\mathbf{u}) = \lambda \mathcal{B}(\mathbf{u}) \quad \text{in } \Omega, \quad (3.1)$$

$$\mathbf{u}_\alpha = 0 \quad \text{on } \Gamma_\alpha^D, \quad \forall \alpha \in \llbracket 1, m \rrbracket, \quad (3.2)$$

$$\frac{\partial \mathbf{u}}{\partial n_{\mathcal{H}_\alpha}} + a_\alpha^R \mathbf{u}_\alpha = 0 \quad \text{on } \Gamma_\alpha^R, \quad \forall \alpha \in \llbracket 1, m \rrbracket, \quad (3.3)$$

where  $\mathcal{B}$  is a given  $\mathcal{H}$ -operator.

In most cases  $\mathcal{B}$  is the identity operator ( $\mathcal{B}$  is a diagonal  $\mathcal{H}$ -operator with  $\mathcal{B}_{\alpha,\alpha} = \mathcal{L}_{0_{d \times d}, \mathbf{0}_d, \mathbf{0}_d, 1}$ ,  $\forall \alpha \in \llbracket 1, m \rrbracket$ ).

### 3.1 Biharmonic eigenvalue BVP for plate vibration

The biharmonic eigenvalue problem for plate vibration is to find  $u \neq 0$  and  $\lambda \in \mathbb{R}$  such that

$$\Delta^2 u = \lambda u, \quad \text{in } \Omega \quad (3.4)$$

where boundary conditions on  $\Gamma$  can be

- *Clamped Plate* (CP) or *pure Dirichlet* type:

$$u = \frac{\partial u}{\partial \mathbf{n}} = 0 \quad (3.5)$$

- *Simply Supported Plate* (SSP) or *Navier* type :

$$u = \Delta u = 0 \quad (3.6)$$

- *Cahn-Hilliard* (CH) type

$$\frac{\partial u}{\partial n} = \frac{\partial \Delta u}{\partial n} = 0 \quad (3.7)$$

Classically the fourth-order PDE (3.4) is converted to the two second-order PDE

$$-\Delta v = \lambda u \quad (3.8)$$

$$-\Delta u = v \quad (3.9)$$

So with the the  $\mathcal{H}$ -operator  $\mathcal{G}$  defined in (A.9) these two PDE can be written as

$$\mathcal{G} \begin{pmatrix} u \\ v \end{pmatrix} \stackrel{\text{def}}{=} \begin{pmatrix} 0 & \mathcal{L}_{\mathbb{I}, \mathbf{0}, \mathbf{0}, 0} \\ \mathcal{L}_{\mathbb{I}, \mathbf{0}, \mathbf{0}, 0} & \mathcal{L}_{\mathbf{0}, \mathbf{0}, \mathbf{0}, -1} \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \lambda u \\ 0 \end{pmatrix} \quad (3.10)$$

If we define the  $\mathcal{B}$  operator as

$$\mathcal{B} \stackrel{\text{def}}{=} \begin{pmatrix} \mathcal{L}_{\mathbf{0}, \mathbf{0}, \mathbf{0}, 1} & 0 \\ 0 & 0 \end{pmatrix} \quad (3.11)$$

the two PDEs (3.8) can be equivalently written as

$$\mathcal{G} \begin{pmatrix} u \\ v \end{pmatrix} = \lambda \mathcal{B} \begin{pmatrix} u \\ v \end{pmatrix} \quad (3.12)$$

The code to build the two operators  $\mathcal{G}$  and  $\mathcal{B}$  is given in Listing 3.1. One can also used the FC-VFEMP<sub>1</sub>-BIHARMONIC package for a more concise code given in Listing 3.2

```
from fc_vfemp1.operators import Loperator, Hoperator
Lop=Loperator(dim=2,A=[[1,None],[None,1]])
Gop=Hoperator(dim=2,m=2)
Gop.H[0][1]=Gop.H[1][0]=Lop
Gop.H[1][1]=Loperator(dim=2,a0=-1)
Bop=Hoperator(dim=2,m=2)
Bop.H[0][0]=Loperator(dim=2,a0=1)
```

Listing 3.1: Setting the  $\mathcal{G}$  and  $\mathcal{B}$  operator with the FC-VFEMP<sub>1</sub> package in 2D

```
from fc_vfemp1.operators import Loperator, Hoperator
import fc_vfemp1_biharmonic.lib as blib
bvp=blib.BiharmonicOperator(2)
Bop=Hoperator(dim=2,m=2)
Bop.H[0][0]=Loperator(dim=2,a0=1)
```

Listing 3.2: Setting the  $\mathcal{G}$  and  $\mathcal{B}$  operator with the FC-VFEMP<sub>1</sub> and the FC-VFEMP<sub>1</sub>-BIHARMONIC packages in 2D

### 3.1.1 Simply supported plate



#### EBVP 1 : biharmonic eigenvalue problem for plate vibration with simply supported plate boundary conditions

Find  $\lambda \in \mathbb{K}$  and  $u \neq 0$  such that

$$\begin{aligned} \Delta^2 u &= \lambda u, \quad \text{in } \Omega \\ u &= 0 \quad \text{and} \quad \Delta u = 0, \quad \text{on } \Gamma \end{aligned}$$

This problem can be rewritten with operators  $\mathcal{G}$  and  $\mathcal{B}$  defined respectively in (3.10) and (3.11). Let  $v = -\Delta u$  and  $\mathbf{w} = (u, v)$ , previous problem becomes



#### Vector EBVP 2 : biharmonic eigenvalue problem for plate vibration with simply supported plate boundary conditions with $\mathcal{H}$ -operators

Find  $\lambda \in \mathbb{K}$  and  $\mathbf{w} = (w_1, w_2) \in (H^2(\Omega))^2$ ,  $\mathbf{w} \neq 0$ , such that

$$\begin{aligned} \mathcal{G}(\mathbf{w}) &= \lambda \mathcal{B}(\mathbf{w}) && \text{in } \Omega, \\ w_1 &= 0 \quad \text{and} \quad w_2 = 0 && \text{on } \Gamma. \end{aligned}$$

**Remark 3.1** On convex domains, the biharmonic eigenvalues are just the squares of the eigenvalues for the Laplace operator with the homogeneous Dirichlet boundary condition with the corresponding eigenfunctions.

#### Application on $\Omega = [0, L] \times [0, H]$

The eigenvalues and the eigenvectors of the Laplacian with Dirichlet boundary conditions are given in (2.6). In Listing 3.3, the part of the file `biharmonic_SSP_rectangle_ex.py`<sup>1</sup> which solve the biharmonic eigenvalue problem for plate vibration with simply supported plate boundary conditions is given

```
bvp=blib.BiharmonicBVP(Th)
for lab in Th.sThlab[Th.find(d=1)]:
    blib.setSimplySupportedPlate(bvp, lab, [0, 0])
Bop=Hoperator(dim=2,m=2)
Bop.H[0][0]=Loperator(dim=2,a0=1)
eigVal, eigVec=elib.eBVPsolve(bvp, RHSop=Bop, k=NumEigs,
    which='LM', sigma=sigma, tol=1e-9, maxiter=Th.nq)
```

Listing 3.3: biharmonic eigenvalue problem for plate vibration with simply supported plate boundary conditions  $\Omega = [0, 3] \times [0, 2]$ .

We represent in Figure 3.1 the twelve first eigenvectors obtained with the following python code:

```
from fc_vfemp1_eigs.examples.biharmonic_SSP_rectangle_ex import run
res=run(N=200,k=12,gsolx=False,gerror=False,trans=False)
```

In Figure 3.3 the orders of convergence of the first twelve eigenvalues are represented.

One can see that a superconvergence phenomena occurs due to regularity of the hypercube mesh. Indeed, for the  $H^1$ -norm an order 1 is expected. To highlight it, gmsh is now used to generate all the meshes of  $\Omega$ : results are given in Figure 3.4.

#### Application on a disk with 5 holes

We represent in Figure 3.5 the twelve first eigenvectors.

### 3.1.2 Mixed boundary conditions

#### eBVP 2 : biharmonic eigenvalue problem for plate vibration with mixed boundary conditions

Find  $\lambda \in \mathbb{K}$  and  $u \neq 0$  such that

$$\begin{aligned} \Delta^2 u &= \lambda u, \text{ in } \Omega \\ u &= 0 \text{ and } \Delta u = 0, \text{ on } \Gamma_{\text{SSP}} \\ u &= 0 \text{ and } \frac{\partial u}{\partial n} = 0, \text{ on } \Gamma_{\text{CP}} \\ \frac{\partial u}{\partial n} &= 0 \text{ and } \frac{\partial \Delta u}{\partial n} = 0, \text{ on } \Gamma_{\text{CH}} \end{aligned}$$

This problem can be rewritten with operators  $\mathcal{G}$  and  $\mathcal{B}$  defined respectively in (3.10) and (3.11). Let  $v = -\Delta u$  and  $\mathbf{w} = (u, v)$ , previous problem becomes

<sup>1</sup>directory: `fc_vfemp1_eigs/examples`

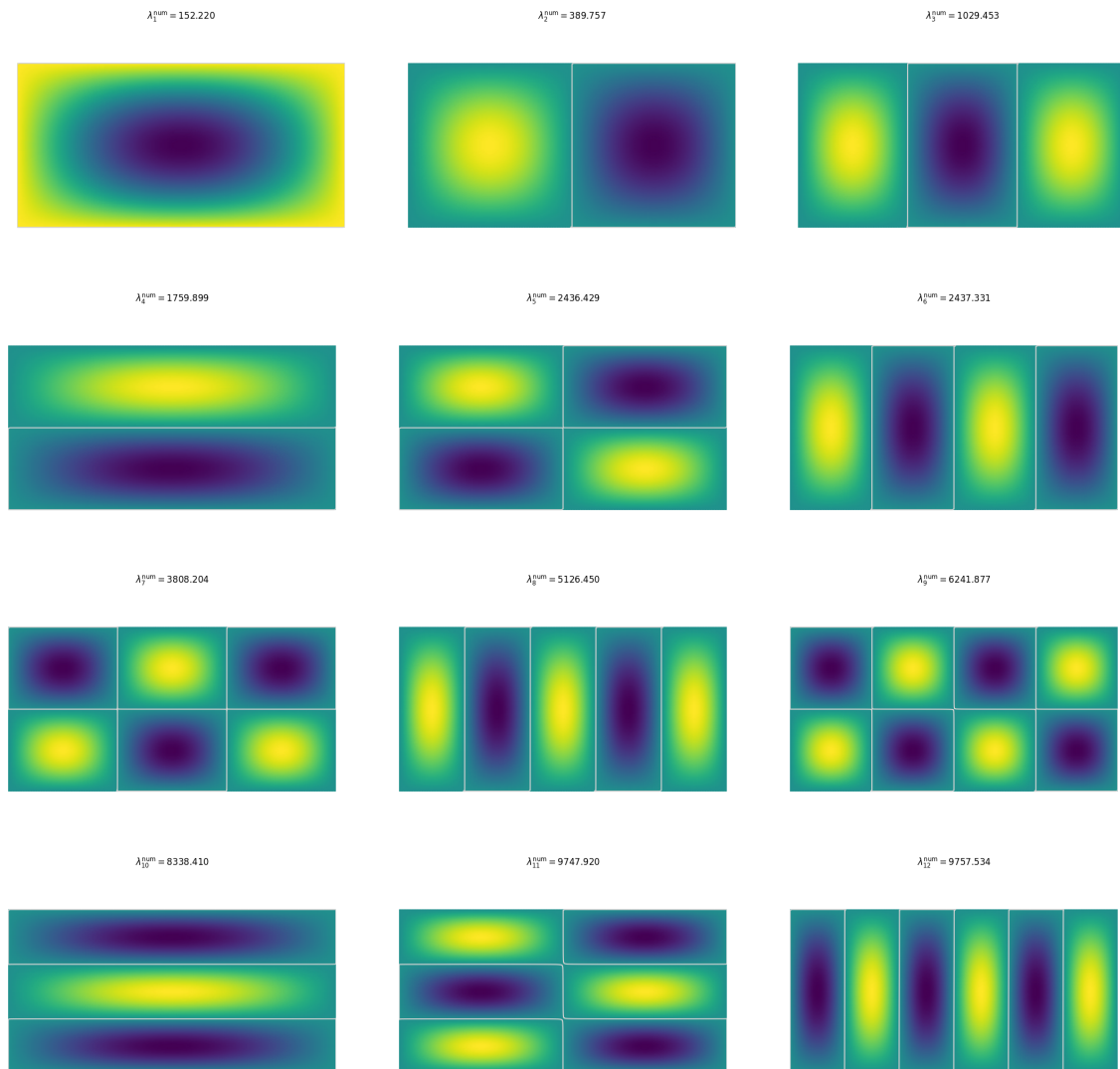


Figure 3.1: title

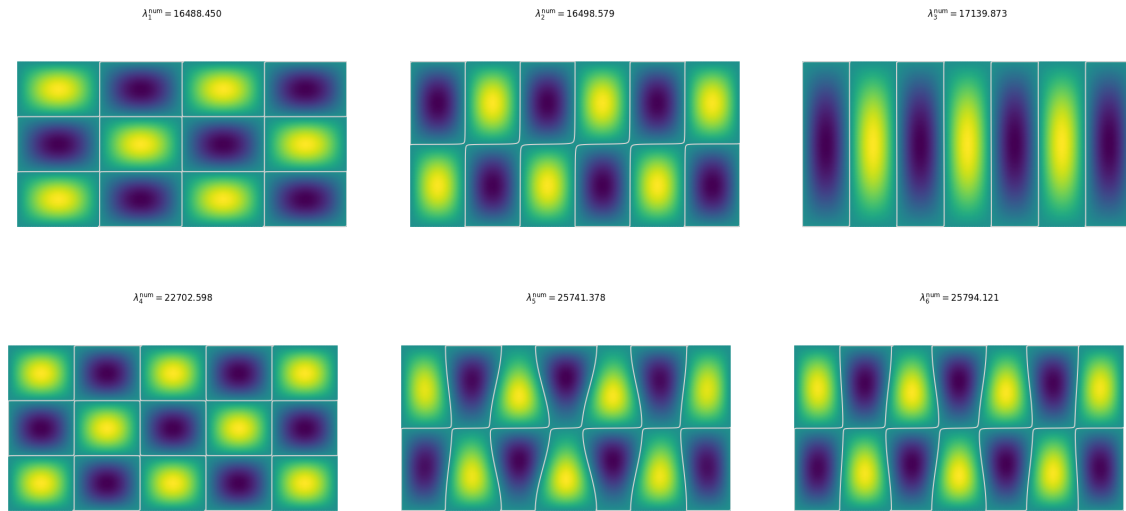


Figure 3.2: title

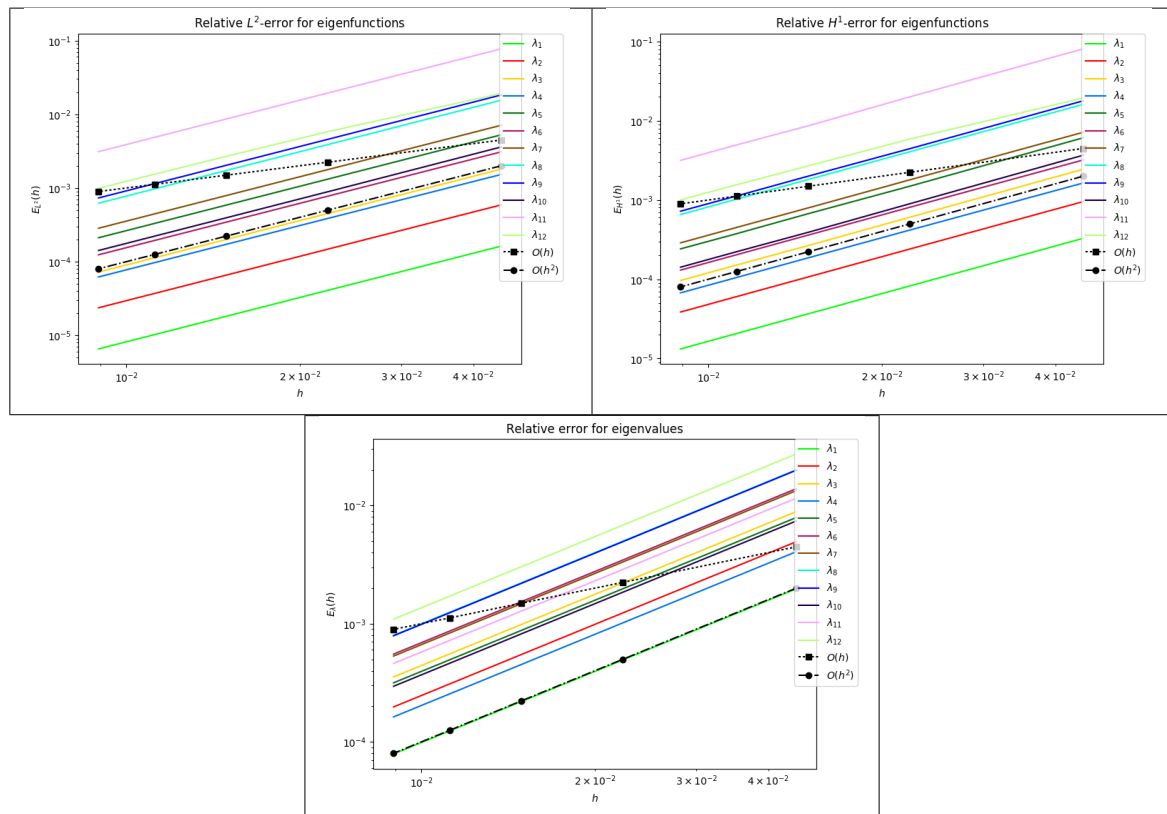


Figure 3.3: eigenvalues and eigenfunctions : order computation for biharmonic ... with (SSP) boundary conditions on rectangle (regular meshes) . Relative errors of eigenfunctions in  $L^2$ -norm (upper left) and  $H^1$ -norm (upper right). Relative errors of eigenvalues (bottom).

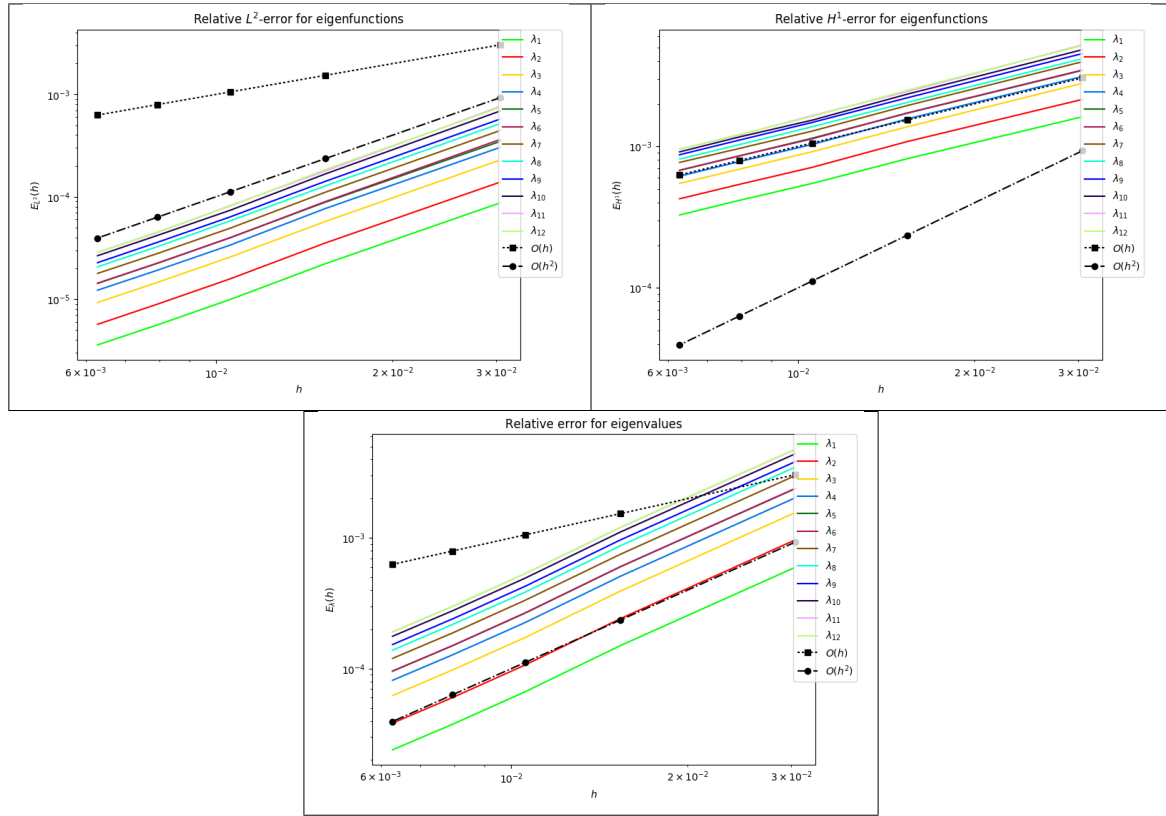


Figure 3.4: eigenvalues and eigenfunctions : order computation for biharmonic ... with (SSP) boundary conditions on rectangle (gmsh meshes) . Relative errors of eigenfunctions in  $L^2$ -norm (upper left) and  $H^1$ -norm (upper right). Relative errors of eigenvalues (bottom).



### Vector EBVP 3 : biharmonic eigenvalue problem for plate vibration with mixed boundary conditions with $\mathcal{H}$ -operators

Find  $\lambda \in \mathbb{K}$  and  $\mathbf{w} = (w_1, w_2) \in (H^2(\Omega))^2$ ,  $\mathbf{w} \neq 0$ , such that

$$\begin{aligned}
 \mathcal{G}(\mathbf{w}) &= \lambda \mathcal{B}(\mathbf{w}) && \text{in } \Omega, \\
 w_1 = 0 \text{ and } w_2 &= 0 && \text{on } \Gamma_{\text{SSP}}, \\
 w_1 = 0 \text{ and } \frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_2}} &= 0 && \text{on } \Gamma_{\text{CP}}, \\
 \frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_2}} \text{ and } \frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_1}} &= 0 && \text{on } \Gamma_{\text{CH}}.
 \end{aligned}$$

#### Application on disk with 5 holes

The eigenvalues and the eigenvectors of the Laplacian with Dirichlet boundary conditions are given in (??).

We represent in Figure 3.7 the twelve first eigenvectors.

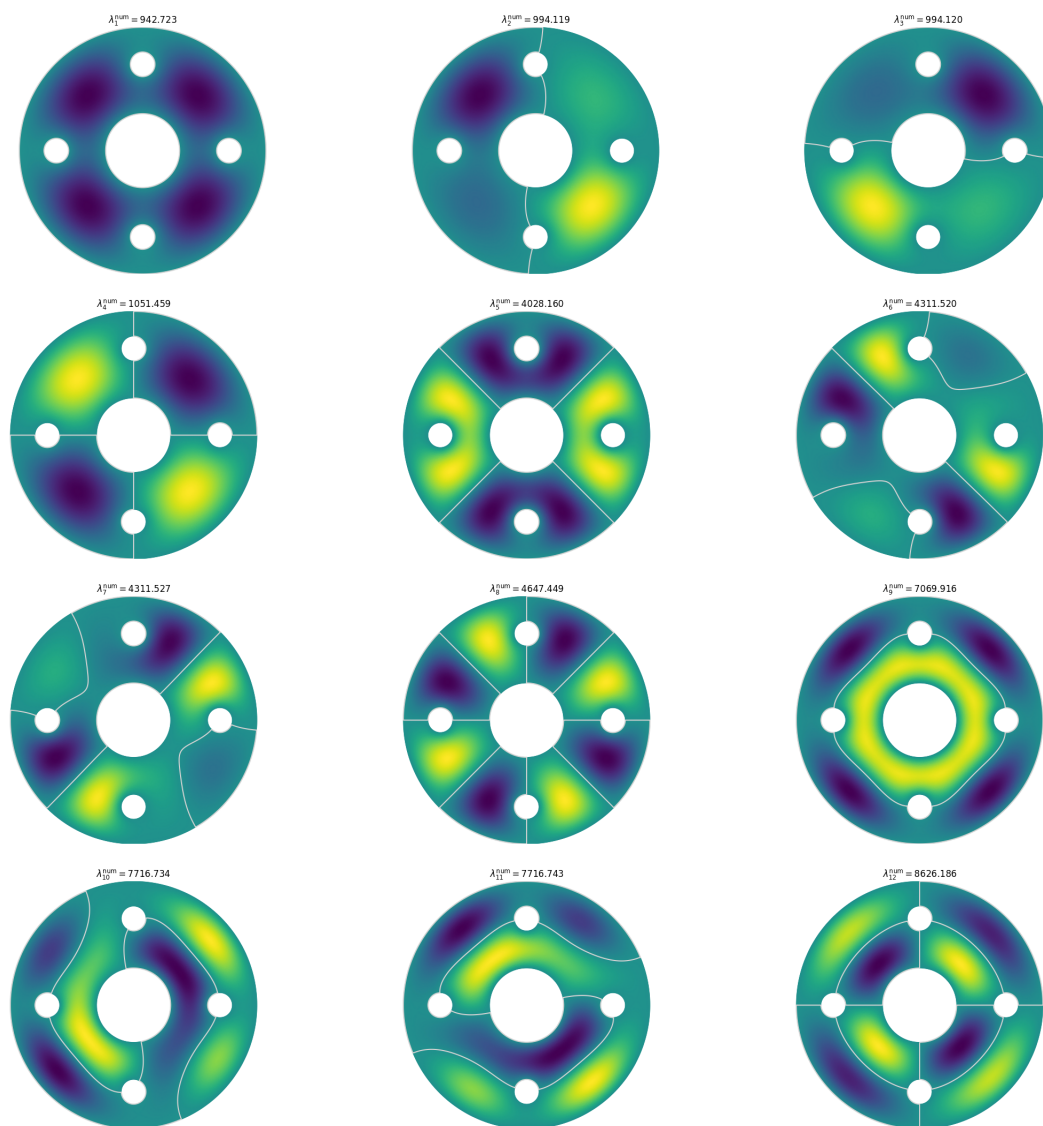


Figure 3.5: title

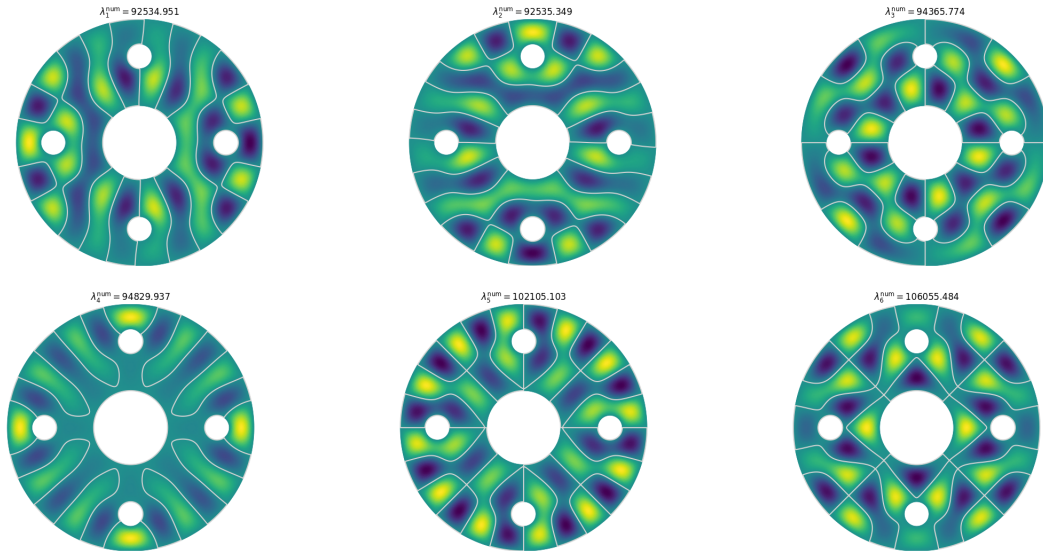


Figure 3.6: title

## 3.2 Linear elasticity

### 3.2.1 Elasticity problem

Let  $d = 2$  or  $d = 3$ . We consider here Hooke's law in linear elasticity, under small strain hypothesis (see for example [6]).

For a sufficiently regular vector field  $\mathbf{u} = (u_1, \dots, u_d) : \Omega \rightarrow \mathbb{R}^d$ , we define the linearized strain tensor  $\underline{\epsilon}$  by

$$\underline{\epsilon}(\mathbf{u}) = \frac{1}{2} (\nabla(\mathbf{u}) + \nabla^t(\mathbf{u})).$$

We set  $\underline{\epsilon} = (\epsilon_{11}, \epsilon_{22}, 2\epsilon_{12})^t$  in 2d and  $\underline{\epsilon} = (\epsilon_{11}, \epsilon_{22}, \epsilon_{33}, 2\epsilon_{12}, 2\epsilon_{23}, 2\epsilon_{13})^t$  in 3d, with  $\epsilon_{ij}(\mathbf{u}) = \frac{1}{2} \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right)$ . Then the Hooke's law writes

$$\underline{\sigma} = \mathbb{C} \underline{\epsilon},$$

where  $\underline{\sigma}$  is the elastic stress tensor and  $\mathbb{C}$  the elasticity tensor.

The material is supposed to be isotropic and could be characterized by its Poisson's ratio  $\nu$ , its Young's modulus  $E$  and its density  $\rho$ . For example, for the Aluminium we have  $\nu = 0.334$ ,  $E = 71 \text{ GPa} = 71 \times 10^{-6} \text{ kg.s}^{-2}.\text{mm}^{-1} = 71 \times 10^{-9} \text{ kg.s}^{-2}.\text{m}^{-1}$  and  $\rho = 2.770 \times 10^{-6} \text{ kg.mm}^{-3} = 2770 \text{ kg.m}^{-3}$ . The Lamé parameters  $\lambda$  and  $\mu$  are given by

$$\lambda = \frac{E * \nu}{(1 + \nu)(1 - 2\nu)} \quad \mu = \frac{E}{2(1 + \nu)}.$$

Let  $\gamma = 2\mu + \lambda$ . The elasticity tensor  $\mathbb{C}$  in dimension 2 or 3 is given by

$$\mathbb{C} = \begin{pmatrix} \lambda \mathbb{1}_2 + 2\mu \mathbb{I}_2 & 0 \\ 0 & \mu \end{pmatrix}_{3 \times 3} \quad \text{or} \quad \mathbb{C} = \begin{pmatrix} \lambda \mathbb{1}_3 + 2\mu \mathbb{I}_3 & 0 \\ 0 & \mu \mathbb{I}_3 \end{pmatrix}_{6 \times 6},$$

respectively, where  $\mathbb{1}_d$  is a  $d$ -by- $d$  matrix of ones, and  $\mathbb{I}_d$  the  $d$ -by- $d$  identity matrix.

For dimension  $d = 2$  or  $d = 3$ , we have:

$$\sigma_{\alpha\beta}(\mathbf{u}) = 2\mu \epsilon_{\alpha\beta}(\mathbf{u}) + \lambda \text{tr}(\underline{\epsilon}(\mathbf{u})) \delta_{\alpha\beta} \quad \forall \alpha, \beta \in \llbracket 1, d \rrbracket$$

The eigenvalue problem to solve is the following

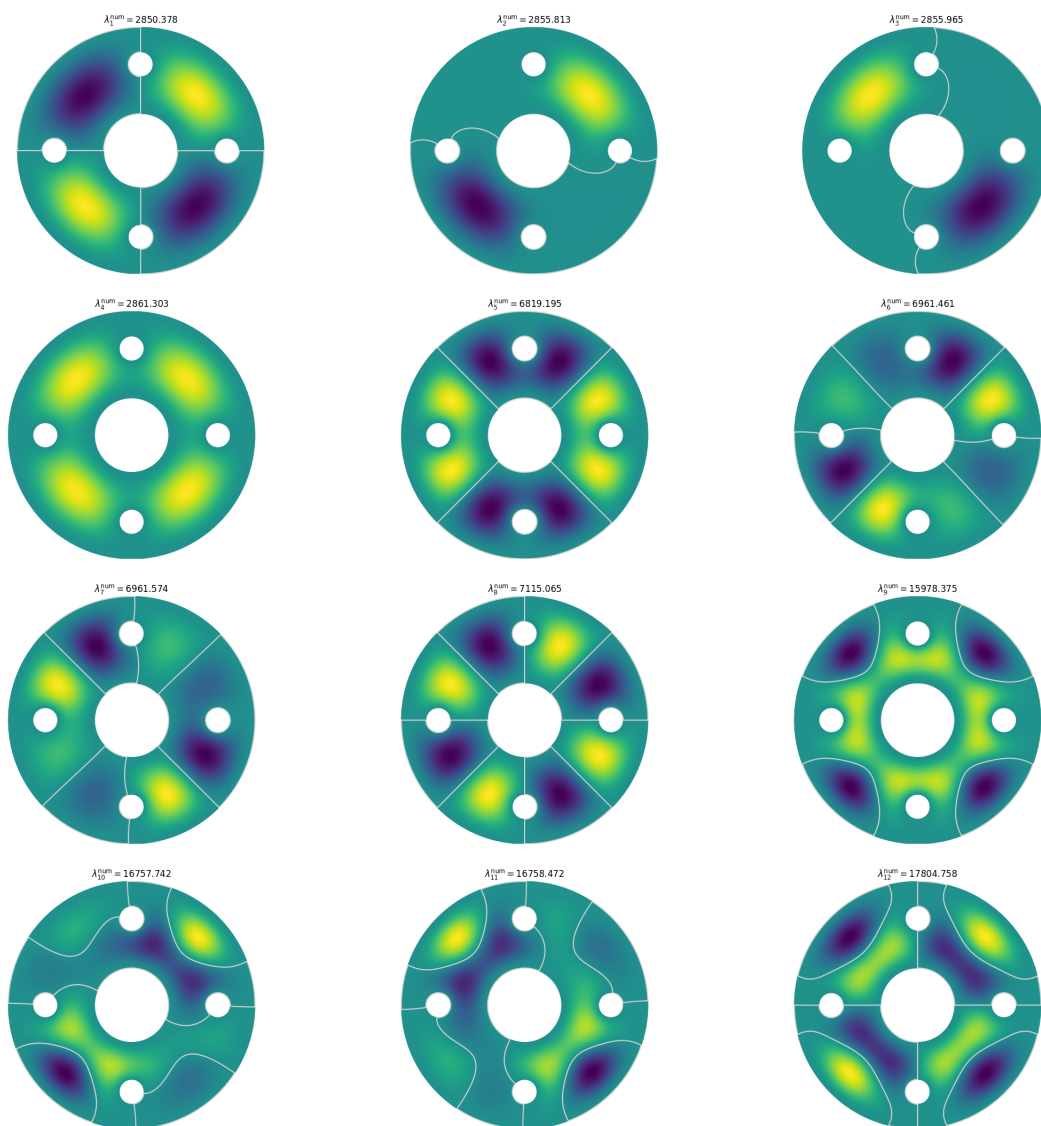


Figure 3.7: title

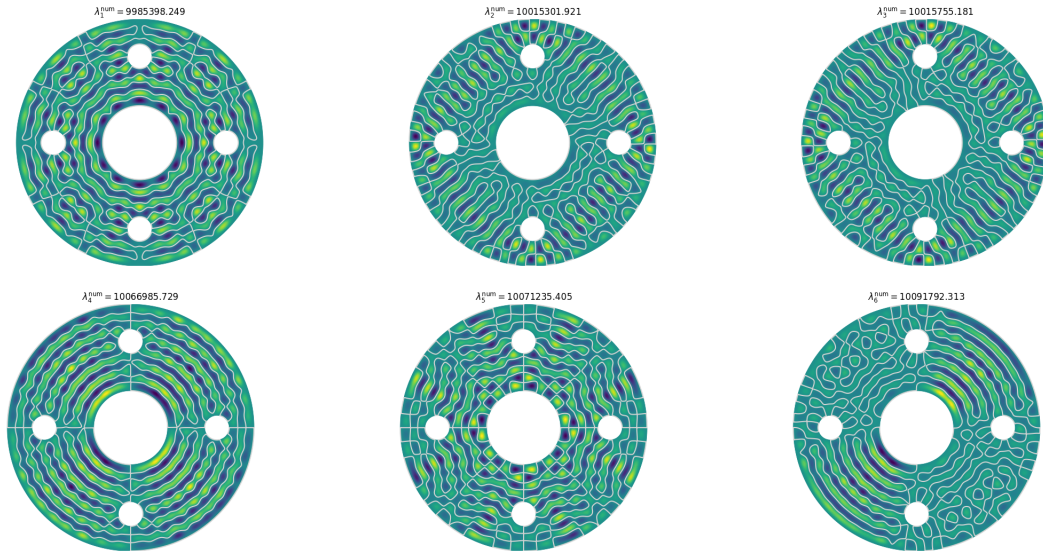


Figure 3.8: title

**Usual EBVP 4 : Elasticity problem**

Find  $(\kappa, \mathbf{u}) = \mathbb{K} \times \mathbf{H}^2(\Omega)^d$  such that

$$-\operatorname{div}(\boldsymbol{\sigma}(\mathbf{u})) = \kappa \rho \mathbf{u}, \quad \text{in } \Omega \subset \mathbb{R}^d, \quad (3.13)$$

$$\boldsymbol{\sigma}(\mathbf{u}) \cdot \mathbf{n} = \mathbf{0} \quad \text{on } \Gamma^R, \quad (3.14)$$

$$\mathbf{u} = \mathbf{0} \quad \text{on } \Gamma^D. \quad (3.15)$$

We recall the following lemma (see [4])

**Lemme 3.2**

Let  $\mathcal{H}^\sigma$  be the  $\mathcal{H}$ -operator defined in (1.10) by

$$\mathcal{H}_{\alpha,\beta} = \mathcal{L}_{\mathbb{A}^{\alpha,\beta}, \mathbf{0}, \mathbf{0}, \mathbf{0}}, \quad \forall (\alpha, \beta) \in \llbracket 1, d \rrbracket^2 \quad (3.16)$$

with

$$(\mathbb{A}^{\alpha,\beta})_{k,l} = \mu \delta_{\alpha\beta} \delta_{kl} + \mu \delta_{k\beta} \delta_{l\alpha} + \lambda \delta_{k\alpha} \delta_{l\beta}, \quad \forall (k, l) \in \llbracket 1, d \rrbracket^2. \quad (3.17)$$

Then, we have

$$\mathcal{H}^\sigma(\mathbf{u}) = -\operatorname{div} \boldsymbol{\sigma}(\mathbf{u}) \quad (3.18)$$

and,  $\forall \alpha \in \llbracket 1, d \rrbracket$ ,

$$\frac{\partial \mathbf{u}}{\partial n_{\mathcal{H}_\alpha^\sigma}} = (\boldsymbol{\sigma}(\mathbf{u}) \cdot \mathbf{n})_\alpha. \quad (3.19)$$

The matrices  $\mathbb{A}^{\alpha,\beta}$  of previous lemma are explicitly given by

- for  $d = 2$ ,

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

- for  $d = 3$ ,

$$\begin{aligned}
\mathbb{A}^{1,1} &= \begin{pmatrix} \gamma & 0 & 0 \\ 0 & \mu & 0 \\ 0 & 0 & \mu \end{pmatrix}, & \mathbb{A}^{1,2} &= \begin{pmatrix} 0 & \lambda & 0 \\ \mu & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, & \mathbb{A}^{1,3} &= \begin{pmatrix} 0 & 0 & \lambda \\ 0 & 0 & 0 \\ \mu & 0 & 0 \end{pmatrix} \\
\mathbb{A}^{2,1} &= \begin{pmatrix} 0 & \mu & 0 \\ \lambda & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, & \mathbb{A}^{2,2} &= \begin{pmatrix} \mu & 0 & 0 \\ 0 & \gamma & 0 \\ 0 & 0 & \mu \end{pmatrix}, & \mathbb{A}^{2,3} &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & \lambda \\ 0 & \mu & 0 \end{pmatrix}, \\
\mathbb{A}^{3,1} &= \begin{pmatrix} 0 & 0 & \mu \\ 0 & 0 & 0 \\ \lambda & 0 & 0 \end{pmatrix}, & \mathbb{A}^{3,2} &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & \mu \\ 0 & \lambda & 0 \end{pmatrix}, & \mathbb{A}^{3,3} &= \begin{pmatrix} \mu & 0 & 0 \\ 0 & \mu & 0 \\ 0 & 0 & \gamma \end{pmatrix}.
\end{aligned}$$

So the elasticity problem (3.13) to (3.23) can be equivalently written as :

#### **Vector EBVP 4 : Linear elasticity in dimension $d = 2$ or $d = 3$**

Find  $(\kappa, \mathbf{u}) \in \mathbb{K} \times (H^2(\Omega))^d$  such that

$$\mathcal{H}^\sigma(\mathbf{u}) = \kappa \mathcal{B}^\sigma(\mathbf{u}), \quad \text{in } \Omega, \quad (3.20)$$

$$\frac{\partial \mathbf{u}}{\partial n_{\mathcal{H}^\sigma}} = 0, \quad \text{on } \Gamma_\alpha^R = \Gamma^R, \quad \forall \alpha \in \llbracket 1, d \rrbracket \quad (3.21)$$

$$\mathbf{u}_\alpha = 0, \quad \text{on } \Gamma_\alpha^D = \Gamma^D, \quad \forall \alpha \in \llbracket 1, d \rrbracket. \quad (3.22)$$

with  $\mathcal{B}_{\alpha,\beta}^\sigma = \delta_{\alpha,\beta} \mathcal{L}_{0,0,0,\rho}$ .

Thereafter, we give the python code which sets the operators  $\mathcal{H}^\sigma$  and  $\mathcal{B}^\sigma$  and initialize the eigenvalue boundary problem with Neumann boundary conditions (3.21) by default on all boundaries.

```

from fc_vfem1.operators import Hmass, StiffElasHoperator
from fc_vfem1.BVP import BVP, PDE
mu = E/(2*(1+nu))
lam = E*nu/((1+nu)*(1-2*nu))
Hop = StiffElasHoperator(2, lam, mu)
pde = PDE(Op=Hop)
bvp = BVP(Th, pde=pde)
Bop = Hmass(Th.dim, Th.dim, a0=rho)

```

On each boundary with a Dirichlet boundary condition (3.22), we just have to do:

```
bvp.setDirichlet(lab, [0., 0.])
```

where `lab` is the label number of the boundary.

### 3.2.2 2D example

Let  $\Omega \subset \mathbb{R}^2$  be a 2d tuning fork obtained from file `tuning-fork2D02.geo` by using `gmsh` (unit is millimeter). One can easily modify the 2d tuning fork parameters  $a$ ,  $b$ ,  $l$ ,  $R$  and  $L$  (see Figure 3.9) when we generate the mesh:

```

import os
from fc_vfem1_eigs.sys import get_geo
geofile = 'tuning-fork2D02'
(geodir, geofile) = get_geo(2, 2, geofile)
options = '-smooth_4_-setnumber_L_75_-setnumber_l_50'
options += '_-setnumber_a_10_-setnumber_b_15_-setnumber_R_15'
meshfile = gmsh.buildmesh2d(geodir+os.sep+geofile+'.geo', 20,
                             force=True, options=options)
Th = siMesh(meshfile)

```

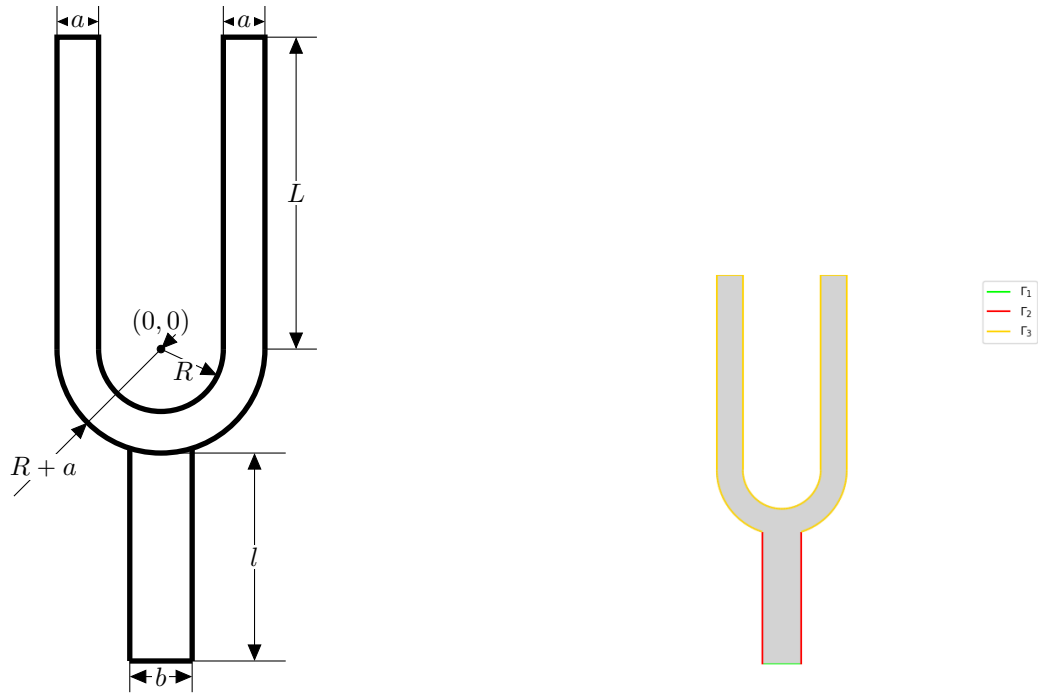


Figure 3.9: 2d tuning fork geometrical parameters (left) and its boundaries (right)

The eigenvalue problem to solve is the following

💡 **Usual EBVP 5 : 2D tuning fork with Dirichlet boundary condition on  $\Gamma_1$**   
 Find  $(\kappa, \mathbf{u}) = \mathbb{K} \times H^1(\Omega)^2$  such that

$$\begin{aligned} -\operatorname{div}(\boldsymbol{\sigma}(\mathbf{u})) &= \kappa \rho \mathbf{u}, \quad \text{in } \Omega \\ \boldsymbol{\sigma}(\mathbf{u}) \cdot \mathbf{n} &= \mathbf{0} \quad \text{on } \Gamma_2 \cup \Gamma_3, \\ \mathbf{u} &= \mathbf{0} \quad \text{on } \Gamma_1. \end{aligned}$$

We represent in Figure 3.10 the first eigenvectors obtained with the following python code:

```
from fc_vfem1_eigs.examples.LinearElasticity_tuning_fork2D02 import run
res=run(k=24,N=60,Dirichlet=[1],max_displacement=5,
        R=10,l=75,L=80.85,cmap='jet',title=False)
```

The displayed frequencies are given by  $f_i = \sqrt{\kappa_i}/(2\pi)$  where  $\kappa_i$  is the  $i$ -th eigenvalue.

In Figure 3.10, we represent the first eigenvectors of the eigenvalue problem

💡 **Usual EBVP 6 : 2D tuning fork with Neumann boundary condition on  $\Gamma_1 \cup \Gamma_2 \cup \Gamma_3$**   
 Find  $(\kappa, \mathbf{u}) = \mathbb{K} \times H^1(\Omega)^2$  such that

$$\begin{aligned} -\operatorname{div}(\boldsymbol{\sigma}(\mathbf{u})) &= \kappa \rho \mathbf{u}, \quad \text{in } \Omega \\ \boldsymbol{\sigma}(\mathbf{u}) \cdot \mathbf{n} &= \mathbf{0} \quad \text{on } \Gamma_1 \cup \Gamma_2 \cup \Gamma_3, \end{aligned}$$

We use the following code:

```
from fc_vfem1_eigs.examples.LinearElasticity_tuning_fork2D02 import run
res=run(k=24,N=60,max_displacement=5, R=10,l=75,L=80.85,cmap='jet',title=False)
```

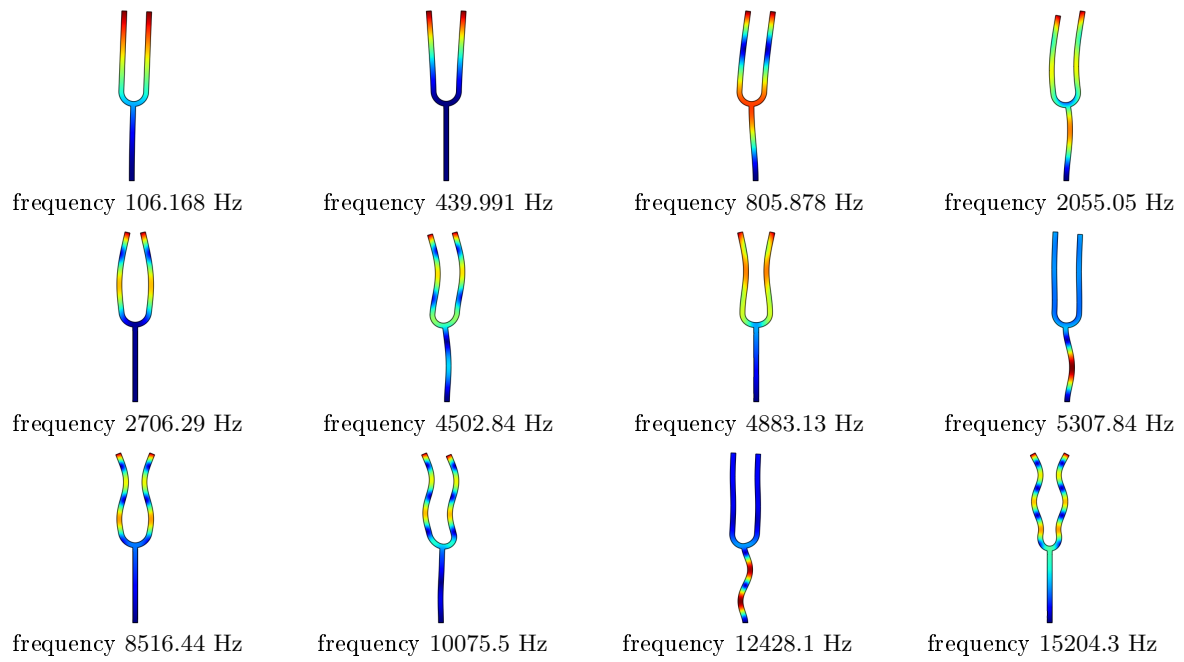


Figure 3.10: 2D Tuning fork with Neumann boundary conditions on  $\Gamma_2 \cup \Gamma_3$  and Dirichlet on  $\Gamma_1$ : eigenvectors of the smallest magnitude eigenvalues stretched to a maximum of 5 mm.

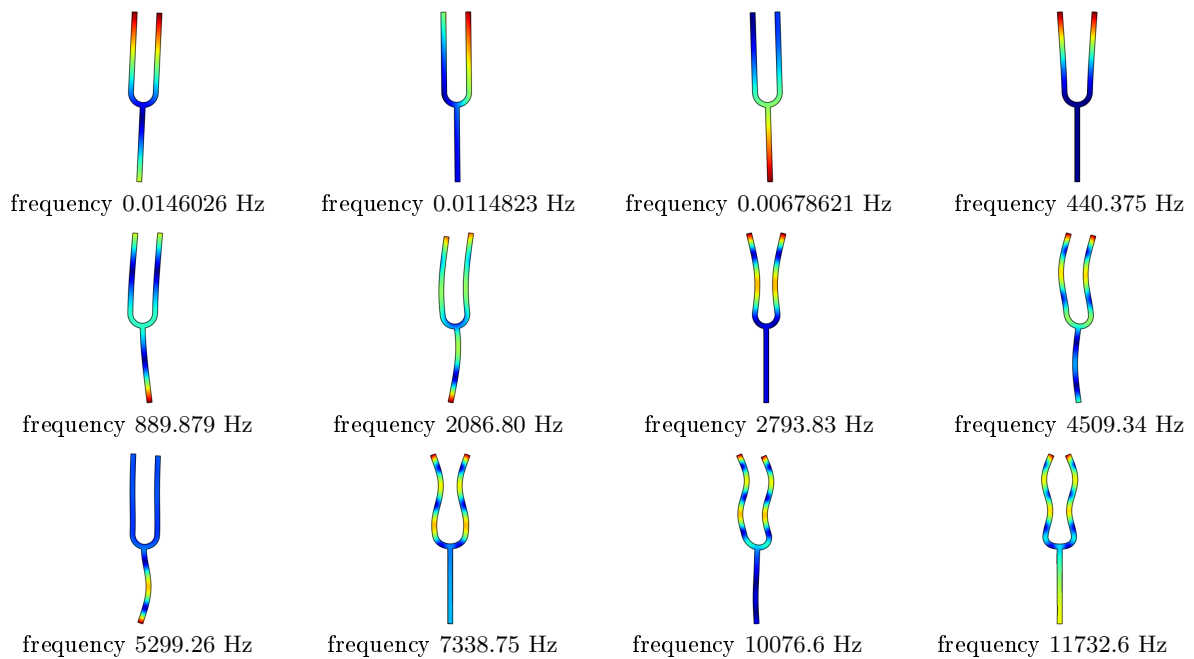


Figure 3.11: 2D Tuning fork with Neumann boundary conditions on  $\Gamma_1 \cup \Gamma_2 \cup \Gamma_3$ : eigenvectors of the smallest magnitude eigenvalues stretched to a maximum of 5 mm.

### 3.2.3 3D example

Let  $\Omega \subset \mathbb{R}^3$  be a 3d tuning fork composed of rods of diameter  $r$ . The geometry is described in Figure 3.12 and the meshes can be obtained by using `gmsh` (version  $\geq 3.0.0$ ) with the file `tuning_fork_02.geo`.

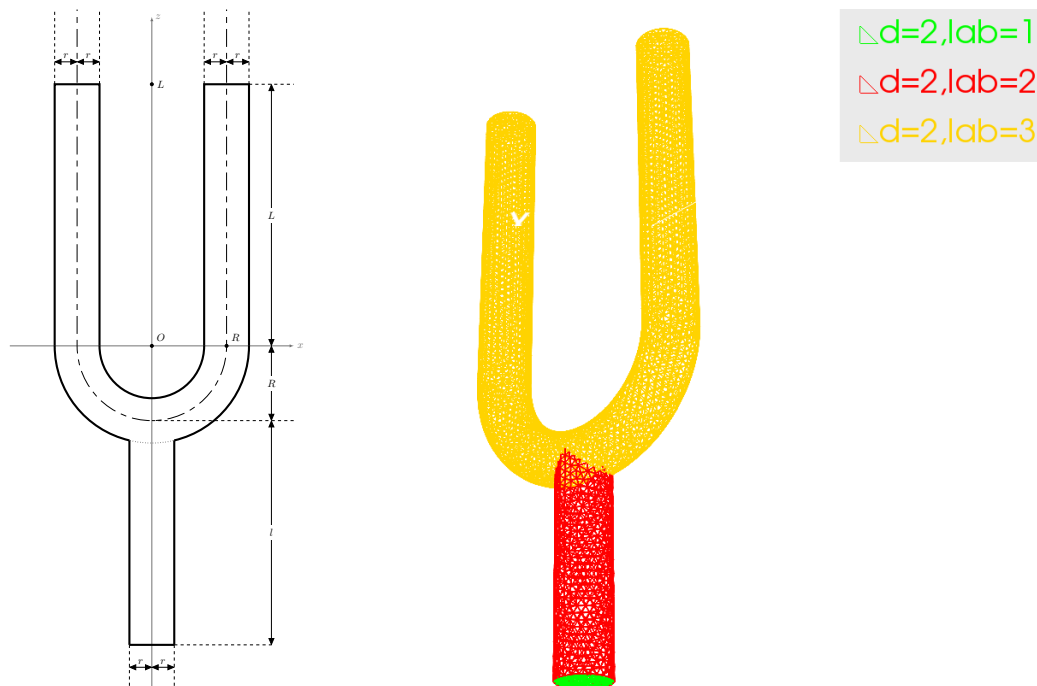


Figure 3.12: 3d tuning fork geometrical parameters (left) and its boundaries (right)

The eigenvalue problem to solve is the following

💡 **Usual EBVP 7 : 3D tuning fork with Dirichlet boundary condition on  $\Gamma_1$**

🌀 Find  $(\kappa, \mathbf{u}) = \mathbb{K} \times H^1(\Omega)^3$  such that

$$\begin{aligned} -\operatorname{div}(\boldsymbol{\sigma}(\mathbf{u})) &= \kappa \rho \mathbf{u}, \quad \text{in } \Omega \\ \boldsymbol{\sigma}(\mathbf{u}) \cdot \mathbf{n} &= \mathbf{0} \quad \text{on } \Gamma_2 \cup \Gamma_3, \\ \mathbf{u} &= \mathbf{0} \quad \text{on } \Gamma_1. \end{aligned}$$

The geometrical parameters, in millimeters, are  $L = 120$ ,  $R = 40$ ,  $l = 100$  and  $r = 10$ . The isotropic material is made of aluminium and so its Poisson's ratio  $\nu$  is 0.334, its Young's modulus  $E$  is  $7.10 \times 10^7 \text{ kg.s}^{-2}.\text{mm}^{-1}$  and its density  $\rho$  is  $2.77 \times 10^{-6} \text{ kg.mm}^{-3}$ .

In Listing 3.4 we give parts of Python code to solve this eigenvalue BVP.

```
import numpy as np
import os
from fc_oogmsh import gmsh
from fc_simesh.siMesh import siMesh
from fc_vfem1.operators import Hmass, StiffElasHoperator
from fc_vfem1.BVP import BVP, PDE
import fc_vfem1_eigs.lib as elib
from fc_vfem1_eigs.sys import get_geo
# Geometrical/Mesh properties:
N=75
L=120 # in mm
r=10 # in mm
R=40 # in mm
l=100 # in mm
# Material properties:
nu=0.334 # Poisson's ratio
```

```

E= 71e6      # Modulus of elasticity in kg/s^2/mm ( GPa=10^9 Pa = 10^6 kg/s^2/mm )
rho=2.770e-6 # Material density in kg/mm^3
# Select Dirichlet boundary:
dirlab=[1] # Dirichlet on Boundary \Gamma_1
# Parameters for eBVPsolve function
NumEigs=9
sigma=0
# Find .geo file
geofile='tuning_fork_02'
(geodir, geofile)=get_geo(3,3, geofile)
geoFile=geodir+os.sep+geofile+'.geo'
options='--smooth_4--setnumber_L_%g--setnumber_l_%g--setnumber_r_%g--setnumber_R_
        %g'%(L,l,r,R)
meshfile=gmshtool.buildmesh3d(geoFile,N, force=True, verbose=3, options=options)
Th=siMesh( meshfile)
# Setting the eBVP
mu= E/(2*(1+nu))
lam = E*nu/((1+nu)*(1-2*nu))
Hop=StiffElastOperator(3, lam, mu)
pde=PDE(Op=Hop)
bvp=BVP(Th, pde=pde)
for lab in dirlab:
    bvp.setDirichlet(lab, [0., 0., 0.])
Bop=Hmass(Th.dim, Th.dim, a0=rho)
# Solving the eBVP
eigVal, eigVec= elib.eBVPsolve(bvp, RHSop=Bop, k=NumEigs, which='LM', sigma=sigma,
    tol=1e-9, maxiter=Th.nq)

```

Listing 3.4: part of the file `LinearElasticity_tuning_fork3D02_light.py`

One can also use the `run` function of the `fc_vfem1_eigs.examples.LinearElasticity_tuning_fork3D02` module to solve this eigenvalue BVP. The twenty-four first eigenfunctions given by the following command are represented in Figure 3.13.

```

from fc_vfem1_eigs.examples.LinearElasticity_tuning_fork3D02 import run
res=run(N=200,L=120,R=40,l=100,r=10,nu=0.334,E=71e6,rho=2.77e-6,
        Dirichlet=[1],k=24,colormap='jet')

```

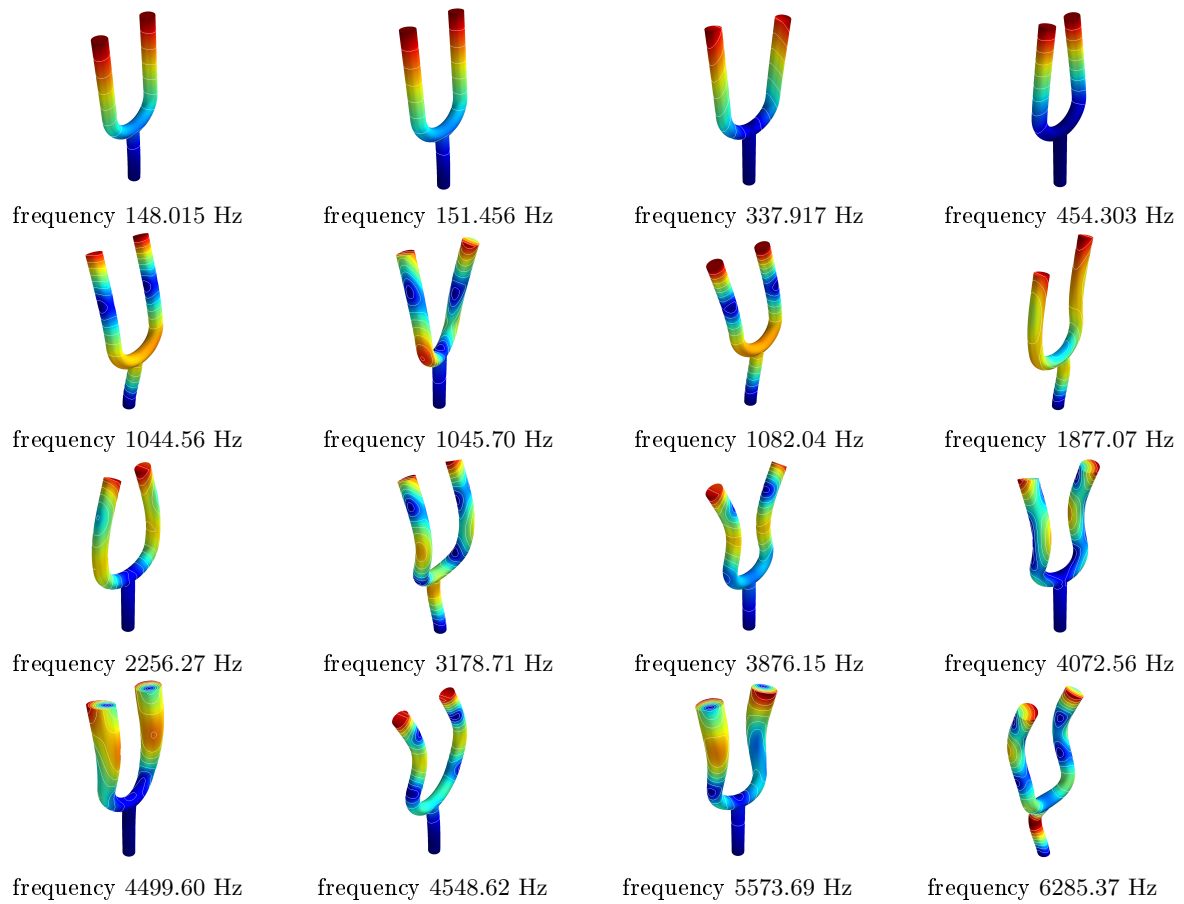


Figure 3.13: 3D Tuning fork with Neumann boundary conditions on  $\Gamma_2 \cup \Gamma_3$  and Dirichlet on  $\Gamma_1$ : eigenvectors of the smallest magnitude eigenvalues stretched to a maximum of 10 mm.

# Chapter A

## Biharmonic BVP

All this section is taken from the ...

Let  $\Omega \subset \mathbb{R}^{\dim}$  and  $\Gamma = \partial\Omega$ . The biharmonic equation is the fourth-order partial PDE given by

$$\Delta^2 u = f, \quad \text{in } \Omega \quad (\text{A.1})$$

where  $\Delta^2 u = \Delta(\Delta u) = \sum_{i=1}^{\dim} \sum_{j=1}^{\dim} \frac{\partial^4 u}{\partial x_i^2 \partial x_j^2} = \sum_{i=1}^{\dim} \frac{\partial^4 u}{\partial x_i^4} + 2 \sum_{i=1}^{\dim} \sum_{j=i+1}^{\dim} \frac{\partial^4 u}{\partial x_i^2 \partial x_j^2}$

The boundary conditions on  $\Gamma$  can be

- *Clamped Plate* (CP) or *pure Dirichlet* type:

$$u = \frac{\partial u}{\partial \mathbf{n}} = g \quad (\text{A.2})$$

- *Simply Supported Plate* (SSP) or *Navier* type :

$$u = \Delta u = g \quad (\text{A.3})$$

- *Pure Hinged Plate* (PHP) or *Steklov* type :

$$u = \Delta u - (1 - \sigma)K \frac{\partial u}{\partial \mathbf{n}} = g \quad (\text{A.4})$$

- *Cahn-Hilliard* (CH) type

$$\frac{\partial u}{\partial n} = \frac{\partial \Delta u}{\partial n} = g \quad (\text{A.5})$$

### A.1 Link with $\mathcal{H}$ -operator

Classically the fourth-order PDE (A.1) is converted to the two second-order PDE

$$-\Delta u = v \quad (\text{A.6})$$

$$-\Delta v = f \quad (\text{A.7})$$

These two equations can be equivalently written as

$$\mathcal{G} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix} \quad \text{or} \quad \mathcal{K} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} 0 \\ f \end{pmatrix} \quad (\text{A.8})$$

where  $\mathcal{G}$  and  $\mathcal{K}$  are the  $\mathcal{H}$ -operators defined by

$$\mathcal{G} = \begin{pmatrix} 0 & \mathcal{L}_{\mathbb{I}, \mathbf{0}, \mathbf{0}, 0} \\ \mathcal{L}_{\mathbb{I}, \mathbf{0}, \mathbf{0}, 0} & \mathcal{L}_{\mathbb{O}, \mathbf{0}, \mathbf{0}, -1} \end{pmatrix} \quad \text{and} \quad \mathcal{K} = \begin{pmatrix} \mathcal{L}_{\mathbb{I}, \mathbf{0}, \mathbf{0}, 0} & \mathcal{L}_{\mathbb{O}, \mathbf{0}, \mathbf{0}, -1} \\ \mathcal{L}_{\mathbb{O}, \mathbf{0}, \mathbf{0}, 0} & \mathcal{L}_{\mathbb{I}, \mathbf{0}, \mathbf{0}, 0} \end{pmatrix} \quad (\text{A.9})$$

The Python code using the FC-VFEMP<sub>1</sub> package to create the operators  $\mathcal{G}$  and  $\mathcal{K}$  are respectively given in Listings A.1 and A.2.

```
from fc_vfemp1.operators import
    Loperator, Hoperator
Lop=Loperator(dim=2,
    A=[[1, None], [None, 1]])
Gop=Hoperator(dim=2,m=2)
Gop.H[0][1]=Gop.H[1][0]=Lop
Gop.H[1][1]=Loperator(dim=2,a0=-1)
```

Listing A.1:  $\mathcal{G}$  operator with the FC-VFEMP<sub>1</sub> package in 2D

```
from fc_vfemp1.operators import
    Loperator, Hoperator
Lop=Loperator(dim=2,
    A=[[1, None], [None, 1]])
Kop=Hoperator(dim=2,m=2)
Kop.H[0][0]=Kop.H[1][1]=Lop
Kop.H[0][1]=Loperator(dim=2,a0=-1)
```

Listing A.2:  $\mathcal{K}$  operator with the FC-VFEMP<sub>1</sub> package in 2D

Let  $\mathbf{w} = (u, v)$ . With the operator  $\mathcal{K}$  given in (A.8), the components of the conormal derivative of  $\mathbf{w}$  defined in (1.17) are given by

$$\begin{aligned} \frac{\partial \mathbf{w}}{\partial n_{\mathcal{K}_1}} &\stackrel{\text{def}}{=} \sum_{\beta=1}^2 \frac{\partial \mathbf{w}_{\beta}}{\partial n_{\mathcal{K}_1, \beta}} = \sum_{\beta=1}^2 \langle \mathbb{A}^{1, \beta} \nabla \mathbf{w}_{\beta}, \mathbf{n} \rangle - \langle \mathbf{b}^{1, \beta} \mathbf{u}_{\beta}, \mathbf{n} \rangle \\ &= \langle \mathbb{I} \nabla \mathbf{w}_1, \mathbf{n} \rangle = \langle \nabla u, \mathbf{n} \rangle \\ &= \frac{\partial u}{\partial \mathbf{n}} \end{aligned} \quad (\text{A.10})$$

and

$$\begin{aligned} \frac{\partial \mathbf{w}}{\partial n_{\mathcal{K}_2}} &\stackrel{\text{def}}{=} \sum_{\beta=1}^2 \frac{\partial \mathbf{w}_{\beta}}{\partial n_{\mathcal{K}_2, \beta}} = \sum_{\beta=1}^2 \langle \mathbb{A}^{2, \beta} \nabla \mathbf{w}_{\beta}, \mathbf{n} \rangle - \langle \mathbf{b}^{2, \beta} \mathbf{u}_{\beta}, \mathbf{n} \rangle \\ &= \langle \mathbb{I} \nabla \mathbf{w}_2, \mathbf{n} \rangle = \langle \nabla v, \mathbf{n} \rangle \\ &= \frac{\partial v}{\partial \mathbf{n}} \end{aligned} \quad (\text{A.11})$$

So with  $\mathcal{K}$  operator one can impose the following boundary conditions

$$\begin{aligned} \mathbf{w}_{\alpha} &= g_{\alpha}^D && \text{on } \Gamma_{\alpha}^D, \quad \forall \alpha \in \llbracket 1, 2 \rrbracket, \\ \frac{\partial \mathbf{w}}{\partial n_{\mathcal{K}_{\alpha}}} + a_{\alpha}^R \mathbf{w}_{\alpha} &= g_{\alpha}^R && \text{on } \Gamma_{\alpha}^R, \quad \forall \alpha \in \llbracket 1, 2 \rrbracket \end{aligned}$$

i.e.

$$\begin{aligned} u &= g_1^D && \text{on } \Gamma_1^D, && v &= g_2^D && \text{on } \Gamma_2^D, \\ \frac{\partial u}{\partial \mathbf{n}} + a_1^R u &= g_1^R && \text{on } \Gamma_1^R, && \frac{\partial v}{\partial \mathbf{n}} + a_2^R v &= g_2^R && \text{on } \Gamma_2^R \end{aligned}$$

**Remark A.1** One can neither impose *clamped plate* (A.2) nor *Pure Hinged Plate* (A.4) boundary conditions with  $\mathcal{K}$  operator. This is why thereafter we will only use the  $\mathcal{G}$  operator.

In the same way, with the operator  $\mathcal{G}$  given in (A.8), the components of the conormal derivative of  $\mathbf{w}$  defined in (1.17) are given by

$$\begin{aligned} \frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_1}} &\stackrel{\text{def}}{=} \sum_{\beta=1}^2 \frac{\partial \mathbf{w}_{\beta}}{\partial n_{\mathcal{G}_1, \beta}} = \sum_{\beta=1}^2 \langle \mathbb{A}^{1, \beta} \nabla \mathbf{w}_{\beta}, \mathbf{n} \rangle - \langle \mathbf{b}^{1, \beta} \mathbf{u}_{\beta}, \mathbf{n} \rangle \\ &= \langle \mathbb{I} \nabla \mathbf{w}_2, \mathbf{n} \rangle = \langle \nabla v, \mathbf{n} \rangle \\ &= \frac{\partial v}{\partial \mathbf{n}} \end{aligned} \quad (\text{A.12})$$

and

$$\begin{aligned}
 \frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_2}} &\stackrel{\text{def}}{=} \sum_{\beta=1}^2 \frac{\partial \mathbf{w}_\beta}{\partial n_{\mathcal{G}_{2,\beta}}} = \sum_{\beta=1}^2 \langle \mathbb{A}^{2,\beta} \nabla \mathbf{w}_\beta, \mathbf{n} \rangle - \langle \mathbf{b}^{2,\beta} \mathbf{u}_\beta, \mathbf{n} \rangle \\
 &= \langle \mathbb{I} \nabla \mathbf{w}_1, \mathbf{n} \rangle = \langle \nabla u, \mathbf{n} \rangle \\
 &= \frac{\partial u}{\partial \mathbf{n}}
 \end{aligned} \tag{A.13}$$

Let us denotes the two partitions avec the boundary  $\Gamma$ :

$$\mathring{\Gamma}_\alpha^D \cap \mathring{\Gamma}_\alpha^R = \emptyset \quad \text{and} \quad \Gamma_\alpha^D \cup \Gamma_\alpha^R = \Gamma, \quad \forall \alpha \in \llbracket 1, 2 \rrbracket \tag{A.14}$$

From *Vector* BVP (1.14)-(1.16), using  $\mathcal{G}$  operator one can impose the following boundary conditions

$$\begin{aligned}
 \mathbf{w}_\alpha &= g_\alpha^D && \text{on } \Gamma_\alpha^D, \quad \forall \alpha \in \llbracket 1, 2 \rrbracket, \\
 \frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_\alpha}} + a_\alpha^R \mathbf{w}_\alpha &= g_\alpha^R && \text{on } \Gamma_\alpha^R, \quad \forall \alpha \in \llbracket 1, 2 \rrbracket
 \end{aligned}$$

i.e.

$$\begin{aligned}
 u &= g_1^D && \text{on } \Gamma_1^D, && v &= g_2^D && \text{on } \Gamma_2^D, \\
 \frac{\partial v}{\partial \mathbf{n}} + a_1^R u &= g_1^R && \text{on } \Gamma_1^R, && \frac{\partial u}{\partial \mathbf{n}} + a_2^R v &= g_2^R && \text{on } \Gamma_2^R
 \end{aligned}$$

To resume, we give a generic biharmonic BVP using the operator  $\mathcal{G}$  in *Vector* BVP (3) which is equivalent to the generic mixed formulation for the biharmonic BVP given in *Vector* BVP (4)



### **Vector BVP 3 : generic biharmonic BVP with $\mathcal{G}$ operator**

Find  $\mathbf{w} = (\mathbf{w}_1, \mathbf{w}_2) \in (\mathbf{H}^2(\Omega))^2$  such that

$$\mathcal{G}(\mathbf{w}) = \begin{pmatrix} f \\ 0 \end{pmatrix} \quad \text{in } \Omega, \tag{A.15}$$

$$\mathbf{w}_1 = g_1^D \quad \text{on } \Gamma_1^D, \quad \mathbf{w}_2 = g_2^D \quad \text{on } \Gamma_2^D, \tag{A.16}$$

$$\frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_1}} + a_1^R \mathbf{w}_1 = g_1^R \quad \text{on } \Gamma_1^R, \quad \frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_2}} + a_2^R \mathbf{w}_2 = g_2^R \quad \text{on } \Gamma_2^R. \tag{A.17}$$



### **Vector BVP 4 : generic mixed formulation for the biharmonic BVP**

Find  $\mathbf{w} = (u, v) \in (\mathbf{H}^2(\Omega))^2$  such that

$$\begin{pmatrix} 0 & \mathcal{L}_{1,0,0,0} \\ \mathcal{L}_{1,0,0,0} & \mathcal{L}_{0,0,0,-1} \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix} \quad \text{in } \Omega, \tag{A.18}$$

$$u = g_1^D \quad \text{on } \Gamma_1^D, \quad v = g_2^D \quad \text{on } \Gamma_2^D, \tag{A.19}$$

$$\frac{\partial v}{\partial \mathbf{n}} + a_1^R u = g_1^R \quad \text{on } \Gamma_1^R, \quad \frac{\partial u}{\partial \mathbf{n}} + a_2^R v = g_2^R \quad \text{on } \Gamma_2^R. \tag{A.20}$$

It's very easy to write (A.15) (or (A.18)) from the generic formulation of the biharmonic BVP with  $\mathcal{G}$  operator with the FC-VFEMP<sub>1</sub> package: the source code is given in Listing A.4 where `Th` is a given `siMesh` object and `f` a given python function or scalar.

```

from fc_vfemp1.operators import Loperator, Hoperator
from fc_vfemp1.BVP import BVP,PDE
Lop=Loperator(dim=2,A=[[1,None],[None,1]])
Gop=Hoperator(dim=2,m=2)
Gop.H[0][1]=Gop.H[1][0]=Lop
Gop.H[1][1]=Loperator(dim=2,a0=-1)
pde=PDE(Op=Gop, f=[f,0])
bvp=BVP(Th, pde=pde)

```

Listing A.3: Writing the  $\mathcal{G}(\mathbf{w}) = \begin{pmatrix} f \\ 0 \end{pmatrix}$  with the FC-VFEMP<sub>1</sub> package

or more concisely FC-VFEM $\mathbb{P}_1$ -BIHARMONIC package:

```
import fc_vfemp1_biharmonic.lib as blib
bvp=blib.BiharmonicBVP(Th, f=f)
```

Listing A.4: Writing shortly the  $\mathcal{G}(\mathbf{w}) = (f, 0)^t$  with the FC-VFEM $\mathbb{P}_1$ -BIHARMONIC package

**Remark A.2** By default the boundary conditions of a `BVP` object are set to **homogeneous Neumann** so we have

$$\frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_1}} = \frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_2}} = 0 \Leftrightarrow \frac{\partial v}{\partial \mathbf{n}} = \frac{\partial u}{\partial \mathbf{n}} = 0$$

which is the homogeneous **Cahn-Hilliard** boundary condition.

Now we will see in the next sections how to set the **Clamped Plate** (CP), **Simply Supported Plate** (SP) and **Cahn-Hilliard** (CH) boundary conditions.

## A.2 Some boundary conditions

Let  $\Gamma_{\text{lab}} \subset \Gamma$ . We want to set one of the following boundary conditions on  $\Gamma_{\text{lab}}$ :

- *Clamped Plate* (CP) or *pure Dirichlet* type:

$$u = g_a \quad \text{and} \quad \frac{\partial u}{\partial \mathbf{n}} = g_b \quad \text{on } \Gamma_{\text{lab}} \quad (\text{A.21})$$

- *Simply Supported Plate* (SSP) or *Navier* type :

$$u = g_a \quad \text{and} \quad \Delta u = -g_b \quad \text{on } \Gamma_{\text{lab}} \quad (\text{A.22})$$

- *Cahn-Hilliard* (CH) type

$$\frac{\partial u}{\partial \mathbf{n}} = g_a \quad \text{and} \quad \frac{\partial \Delta u}{\partial \mathbf{n}} = -g_b \quad \text{on } \Gamma_{\text{lab}} \quad (\text{A.23})$$

Now we will see how to rewrite theses boundary conditions as those in *Vector BVP* 4, equations (A.19)-(A.20), and *Vector BVP* 3, equations (A.16)-(A.17).

### A.2.1 Clamped Plate boundary condition

From (A.19) and (A.20), we deduce that (A.21) imposes

$$u = g_a \quad \text{on } \Gamma_{\text{lab}} \subset \Gamma_1^D \quad \text{and} \quad \frac{\partial u}{\partial \mathbf{n}} = g_b \quad \text{on } \Gamma_{\text{lab}} \subset \Gamma_2^R$$

So with  $\mathcal{G}$  operator and with  $\mathbf{w} = (u, v)$  we obtain

$$w_1 = g_a \quad \text{on } \Gamma_{\text{lab}} \subset \Gamma_1^D \quad \text{and} \quad \frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_2}} = g_b \quad \text{on } \Gamma_{\text{lab}} \subset \Gamma_2^R$$

Let `bvp` be the `BVP` object build in Listing A.4. We want to set this object with the (CP) boundary condition. To set the dirichlet condition on first component  $w_1 = g_a$  on  $\Gamma_{\text{lab}}$ , we can use the `setDirichlet` method of the `BVP` object:

```
bvp.setDirichlet(lab, ga, comps=[0])
```

As python uses 0-based indexing, the first component is selected with `comps=[0]`.

To set the Neumann condition on second component  $\frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_2}} = g_b$  we can use the `setRobin` method of the `BVP` object:

```
bvp.setRobin(lab,gb,comps=[1])
```

As python uses 0-based indexing, the second component is selected with `comps=[1]`.

A more convenient way is to use the dedicated function `setClampedPlate` function:

```
import fc_vfemp1_biharmonic.lib as blib
blib.setClampedPlate(bvp,lab,[ga,gb])
```

Listing A.5: Setting a (CP) boundary condition

## A.2.2 Simply Supported Plate boundary condition

From (A.19) and (A.20), we deduce that (A.22) imposes

$$u = g_a \text{ on } \Gamma_{\text{lab}} \subset \Gamma_1^D \quad \text{and} \quad v = g_b \text{ on } \Gamma_{\text{lab}} \subset \Gamma_2^D$$

So with  $\mathcal{G}$  operator and with  $\mathbf{w} = (u, v)$  we obtain

$$w_1 = g_a \text{ on } \Gamma_{\text{lab}} \subset \Gamma_1^D \quad \text{and} \quad w_2 = g_b \text{ on } \Gamma_{\text{lab}} \subset \Gamma_2^D$$

Let `bvp` be the `BVP` object build in Listing A.4. We want to set this object with the (SSP) boundary condition. To set the dirichlet conditions, we can use the `setDirichlet` method of the `BVP` object:

```
bvp.setDirichlet(lab,ga,comps=[0])
bvp.setDirichlet(lab,gb,comps=[1])
```

or

```
bvp.setDirichlet(lab,[ga,gb])
```

A more convenient way is to use the dedicated function `setSimplySupportedPlate` function:

```
import fc_vfemp1_biharmonic.lib as blib
blib.setSimplySupportedPlate(bvp,lab,[ga,gb])
```

Listing A.6: Setting a (SSP) boundary condition

## A.2.3 Cahn-Hilliard boundary condition

From (A.19) and (A.20), we deduce that (A.23) imposes

$$\frac{\partial v}{\partial \mathbf{n}} = g_b \text{ on } \Gamma_{\text{lab}} \subset \Gamma_1^R \quad \text{and} \quad \frac{\partial u}{\partial \mathbf{n}} = g_a \text{ on } \Gamma_{\text{lab}} \subset \Gamma_2^R$$

where  $v = -\Delta u$ .

So with  $\mathcal{G}$  operator and with  $\mathbf{w} = (u, v)$  we obtain

$$\frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_1}} = g_b \text{ on } \Gamma_{\text{lab}} \subset \Gamma_1^R \quad \text{and} \quad \frac{\partial \mathbf{w}}{\partial n_{\mathcal{G}_2}} = g_a \text{ on } \Gamma_{\text{lab}} \subset \Gamma_2^R$$

Let `bvp` be the `BVP` object build in Listing A.4. We want to set this object with the (CH) boundary condition. To set the Neumann conditions, we can use the `setRobin` method of the `BVP` object:

```
bvp.setRobin(lab,gb,comps=[0])
bvp.setRobin(lab,ga,comps=[1])
```

or more concisely

```
bvp.setRobin(lab,[gb,ga])
```

A more convenient way is to use the dedicated function `setCahnHilliard` function:

```
import fc_vfemp1_biharmonic.lib as blib
blib.setCahnHilliard(bvp,lab,[ga,gb])
```

Listing A.7: Setting a (CH) boundary condition

## 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](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] G. Dhatt, E. Lefrançois, and G. Touzot. *Finite Element Method*. Wiley, 2012.
- [7] L. Fox, P. Henrici, and C. Moler. Approximations and bounds for eigenvalues of elliptic operators. *SIAM Journal on Numerical Analysis*, 4(1):89–102, 1967.
- [8] J. M. Gedicke. *On the Numerical Analysis of Eigenvalue Problems*. PhD thesis, University of Berlin, 2013.
- [9] D. S. Grebenkov and B.-T. Nguyen. Geometrical structure of laplacian eigenfunctions. *SIAM Review*, 55(4):601–667, jan 2013.
- [10] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*, volume 23 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, 1994.
- [11] L. N. Trefethen and T. Betcke. Computed eigenmodes of planar regions. *Manchester Institute for Mathematical Sciences School of Mathematics*, 2006.
- [12] Quan Yuan and Zhiqing He. Bounds to eigenvalues of the laplacian on l-shaped domain by variational methods. *Journal of Computational and Applied Mathematics*, 233(4):1083 – 1090, 2009.