Return your written solutions either in person or by email
to vesa.kaarnioja@fu-berlin.de by Tuesday 6 May 2025, 10:15 am
Please make sure to return your source code for all programming tasks

1. Let $D \subset \mathbb{R}^d$, $d \in \{2,3\}$, be a bounded Lipschitz domain, $f \in L^2(D)$, and let $a \in L^\infty(D)$ be such that $0 < a_{\min} \leq a(\boldsymbol{x}) \leq a_{\max} < \infty$ for almost every $\boldsymbol{x} \in D$ for some constants $a_{\max}, a_{\min} > 0$. Let $u \in H_0^1(D)$ be the unique solution to the weak formulation
$$\int_D a(\boldsymbol{x})\nabla u(\boldsymbol{x}) \cdot \nabla v(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_D f(\boldsymbol{x})v(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \quad \text{for all } v \in H_0^1(D).$$
Let V_m be a finite dimensional subspace of $H_0^1(D)$ and consider the Galerkin solution $u_m \in V_m$ satisfying
$$\int_D a(\boldsymbol{x})\nabla u_m(\boldsymbol{x}) \cdot \nabla v(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_D f(\boldsymbol{x})v(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} \quad \text{for all } v \in V_m.$$
Show that
$$\|u - u_m\|_{H_0^1(D)} \leq \sqrt{\frac{a_{\max}}{a_{\min}}} \inf_{v \in V_m} \|u - v\|_{H_0^1(D)}.$$
Hint: Modify the proof of Céa's lemma from the lecture notes of week 3 by upper bounding the term $B(u-u_m, u-u_m)$ with $B(u-v, u-v)$ for all $v \in V_m$, where $B(u,v) := \int_D a(\boldsymbol{x})\nabla u(\boldsymbol{x}) \cdot \nabla v(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$.

2. Let $D \subset \mathbb{R}^2$ be a nonempty, bounded polygon and let us consider the Poisson problem with an *inhomogeneous* Dirichlet boundary condition. That is, find $u\colon D \to \mathbb{R}$ such that
$$\begin{cases} -\Delta u(\boldsymbol{x}) = f(\boldsymbol{x}), & \boldsymbol{x} \in D, \\ u|_{\partial D} = g, \end{cases} \tag{1}$$
where $f\colon D \to \mathbb{R}$ and $g\colon \partial D \to \mathbb{R}$ are known functions. We are interested in solving this problem using piecewise linear FEM. To this end, suppose that $\mathcal{T}_h$ is a triangulation of the computational domain D with mesh size $h > 0$ and vertices/FE nodes $(\boldsymbol{n}_i)_{i=1}^N$. The FE space $V_h = \mathrm{span}(\phi_i)_{i=1}^N$ is spanned by piecewise linear global FE basis functions $(\phi_i)_{i=1}^N$ such that $\phi_i(\boldsymbol{n}_j) = \delta_{i,j}$, $i,j \in \{1,\ldots,N\}$. Suppose that $\texttt{bnd} := \{i \in \{1,\ldots,N\} \mid \boldsymbol{n}_i \in \partial D\}$ is a set of indices containing the labels of the FE nodes located on the boundary of the domain D and suppose that $\texttt{int} := \{1,\ldots,N\} \setminus \texttt{bnd}$ contains the labels of the interior FE nodes. Let $A \in \mathbb{R}^{N\times N}$ denote the stiffness matrix defined elementwise by setting $A_{i,j} = \int_D \nabla\phi_i(\boldsymbol{x}) \cdot \nabla\phi_j(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}$, $i,j \in \{1,\ldots,N\}$. Finally, let us define the submatrices $A_{\texttt{int},\texttt{int}} = (A_{i,j})_{i,j\in\texttt{int}}$ and $A_{\texttt{int},\texttt{bnd}} = (A_{i,j})_{i\in\texttt{int},j\in\texttt{bnd}}$.

(a) Suppose that g is a piecewise linear function and consider the FE approximation $u_h(\cdot) = \sum_{j=1}^{N} c_j\phi_j(\cdot) \in V_h$ to the Poisson problem (1). The boundary condition can now be imposed *exactly* by setting $c_i = g(\boldsymbol{n}_i)$ for all $i \in \texttt{bnd}$. Show that the expansion coefficients $\boldsymbol{c}_{\texttt{int}} = (c_i)_{i\in\texttt{int}}$ can be solved from the equation

$$A_{\texttt{int},\texttt{int}}\boldsymbol{c}_{\texttt{int}} = \boldsymbol{F}_{\texttt{int}} - A_{\texttt{int},\texttt{bnd}}\boldsymbol{G}_{\texttt{bnd}},$$

where $\boldsymbol{F}_{\texttt{int}} = \big(\int_D f(\boldsymbol{x})\phi_i(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}\big)_{i\in\texttt{int}}$ is the loading vector and $\boldsymbol{G}_{\texttt{bnd}} = (g(\boldsymbol{n}_i))_{i\in\texttt{bnd}}$.

(b) If g is not piecewise linear, then the above method of imposing the boundary condition is no longer exact but instead corresponds to a nodal interpolation of the boundary values. The resulting approximation error can be controlled as long as g is sufficiently smooth, so the above method may still be reasonable in practice. However, an alternative way to mitigate the approximation error is to consider a *Dirichlet lift*:

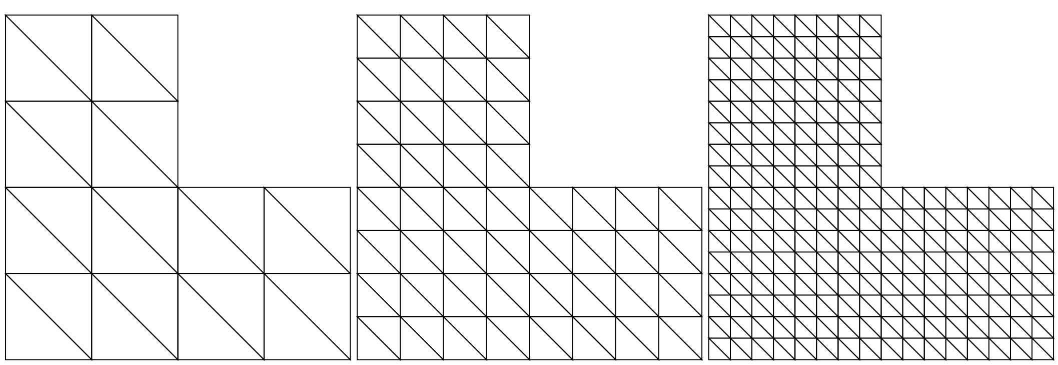
(i) First find a function $\widetilde{g} \in H^2(D)$ such that $\widetilde{g}|_{\partial D} = g$.

(ii) Solve the PDE $-\Delta\widetilde{u} = f + \Delta\widetilde{g}$ in D with $\widetilde{u}|_{\partial D} = 0$ (in practice using FEM).

(iii) The function $u = \widetilde{u} + \widetilde{g}$ now satisfies $-\Delta u = f$ in D with $u|_{\partial D} = g$.

Let $D = (0,1)^2$, $f(\boldsymbol{x}) = x_1 + x_2$, and $g(\boldsymbol{x}) = 1 - x_1^3 - 2x_2$. Solve the problem (1) using both nodal interpolation of the boundary values (method described in part (a)) as well as using a Dirichlet lift to impose the boundary values. You can modify the script `fem.py` available on the course webpage for your computations. Plot the solutions you obtained and compare them visually.

3. Let $D := \{(x_1,x_2) \in \mathbb{R}^2 \mid 0 < x_1 < 1,\ 0 < x_2 < 2\} \cup \{(x_1,x_2) \in \mathbb{R}^2 \mid 1 \leq x_1 < 2,\ 0 < x_2 < 1\} \subset \mathbb{R}^2$ be an *L-shaped domain*. Modify the function `generateFEmesh` in the script `fem.py` on the course page[†] to create a uniform, regular triangulation $\mathcal{T}_h$ of the L-shaped domain with mesh widths $h \in \{2^{-1}, 2^{-2}, 2^{-3}, \ldots\}$. Try also plotting your triangulations. The goal is to obtain something like these triangulations:



The exercises continue on the next page.

[†]Or write your own implementation using your favorite programming language! For this task, it is enough to reproduce the finite element vertex array `nodes` and mesh element connectivity array `element` appearing in `fem.py` for the L-shaped domain.

4. Let $D \subset \mathbb{R}^2$ be a bounded polyhedron. Let us consider the *spectral eigenvalue problem* of finding the eigenvalues $\lambda \in \mathbb{R}$ and eigenfunctions $u\colon D \to \mathbb{R}$ such that

$$\begin{cases} -\Delta u = \lambda u & \text{in } D, \\ u|_{\partial D} = 0, \\ \int_D u(\boldsymbol{x})^2 \, \mathrm{d}\boldsymbol{x} = 1. \end{cases}$$

The weak formulation of this problem is to find $(\lambda, u) \in \mathbb{R} \times (H_0^1(D) \setminus \{\mathbf{0}\})$, $\|u\|_{L^2(D)} = 1$, such that

$$\int_D \nabla u(\boldsymbol{x}) \cdot \nabla v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = \lambda \int_D u(\boldsymbol{x}) v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \quad \text{for all } v \in H_0^1(D).$$

If V_m is a finite dimensional subspace of $H_0^1(D)$, then the goal is to find $(\lambda, u_m) \in \mathbb{R} \times (V_m \setminus \{\mathbf{0}\})$, $\|u_m\|_{L^2(D)} = 1$, such that

$$\int_D \nabla u_m(\boldsymbol{x}) \cdot \nabla v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} = \lambda \int_D u_m(\boldsymbol{x}) v(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \quad \text{for all } v \in V_m. \tag{2}$$

(a) Let $V_h = \operatorname{span}(\phi_i)_{i=1}^m$ be a finite element subspace of $H_0^1(D)$ spanned by continuous, piecewise linear finite element basis functions such that $\phi_i(\boldsymbol{n}_j) = \delta_{i,j}$, where $\boldsymbol{n}_i$ are vertices of the mesh elements lying in the interior of the domain D. Show that (2) can be solved by considering the *generalized eigenvalue problem*

$$S\boldsymbol{c} = \lambda M \boldsymbol{c}, \quad \boldsymbol{c}^{\mathrm{T}} M \boldsymbol{c} = 1, \tag{3}$$

where $\boldsymbol{c} := [c_1, \ldots, c_m]^{\mathrm{T}}$ are the finite element expansion coefficients of the corresponding finite element discretized eigenfunction $u_h(\boldsymbol{x}) = \sum_{i=1}^m c_i \phi_i(\boldsymbol{x})$ and $S = (S_{i,j})_{i,j=1}^m$ and $M = (M_{i,j})_{i,j=1}^m$ are defined by the formulae $S_{i,j} = \int_D \nabla \phi_i(\boldsymbol{x}) \cdot \nabla \phi_j(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}$ and $M_{i,j} = \int_D \phi_i(\boldsymbol{x}) \phi_j(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}$.

(b) Your task is to find the *smallest eigenpair* satisfying (3) and plot the corresponding eigenfunction. As the computational domain D, consider the L-shaped domain from task 3.

It is probably the most convenient to solve the smallest eigenpair of a generalized eigenvalue problem using a command like

```
evals,evecs = scipy.sparse.linalg.eigsh(S,k=1,M=M,which='SM')
```

where `S` is the stiffness matrix corresponding to the Dirichlet–Laplacian $-\Delta$ and `M` is the mass matrix. To obtain a FE mesh, you can either use the script you wrote for task 3 or download the file `femdata.mat` from the course webpage which contains a precomputed mesh. The file contains an array containing the FE nodes, the element connectivity array, a list containing the indices of the interior FE nodes, and the element centers. You can access these via

```
data = scipy.io.loadmat('femdata.mat')
nodes = data['nodes']; element = data['element']
interior = data['interior'][0]; centers = data['centers']
```

You can obtain the appropriate stiffness and mass matrices using the function `generateFEmatrices` in the `fem.py` script available on the course webpage. Meanwhile, you can enforce the homogeneous Dirichlet boundary conditions by slicing the matrices `S` and `M` using the list `interior` corresponding to the labels of the interior FE nodes.

Hint: the eigenfunction should look like the function below: