Homework 2

February 23rd 2021

Deadline: March 11th 2021, the answer to the questions will be submitted via Canvas including the code. You should submit a PDF report containing your reasoning and plots. I should be able to run the code and generate any plot that you used on your answers. Your code will be graded for efficiency, it should run using resources equivalent to a mid-range modern laptop. You can submit your answers in a jupyter notebook, if that is more convenient for you. No late homework will be accepted.

Rules: You are strongly encouraged to discuss the homework with your peers, in particular, piazza is a very good environment for discussing the homework. However, you need to write your own homework and you need disclose you sources. If you work with some of your classmates you need to disclose your collaborators.

Disclaimer: If you are not used to Python several concepts may not be familiar to you. You may want to check the online tutorials shared in the announcements, including the documentation of numpy and scipy (they will be your best friends). Otherwise you can post in piazza, you are encouraged to share code snippets but not a full answer. The same applies if your code is too slow, i.e., I strongly encourage you to post in piazza and interact with your classmates.

A We will use `FEniCs` to solve the Poisson equation with homogeneous Dirichlet boundary conditions on a circle, and we will study the convergence rate with respect to h, the characteristic lenght of a triangulation. In particular we will solve the following equation:

$$\begin{cases} -\triangle u = 4 & \text{in } \Omega \\ u = 0 & \text{on } \partial\Omega \end{cases} \tag{1}$$

(a) Install the `FEniCs` library, including `mshr`, which is an meshing interface. You can find the links in the announcement tab of Canvas. We will use both extensively in this homework.

(b) Find an analyitical solution to (1). You may want to use polar coordinates.

(c) Complete the file `Poisson_circle.py` and compute the solution to (1).

(d) Using your code above, solve the equation at increasingly fine meshes, and provide a semi-log plot of the error with respect to h. What is the overall order of your scheme?

(e) Change the type of finite elements to a continuous piece-wise quadratic polynomials and check the error. Explain the different behavior of the error compared to continuous piece-wise linear polynomials.

B In this problem we will study of "hearing the shape of a drum". In which, the question can we identify the shape of a drum by hearing the different harmonics.

We represent the drum as a domain Ω, which can be arbitrary. In that case the harmonics of the drum are all the functions that satisfy the eigenvalue problem, i.e., all the pairs (λ, u_λ)

$$\begin{cases} -\triangle u_\lambda = \lambda u_\lambda & \text{in } \Omega \\ u_\lambda = 0 & \text{on } \partial\Omega \end{cases} \tag{2}$$

Where λ are related (via the wave equation) to the frequencies we can hear. In that context we want to study if we can recover the shape given the information on the full spectrum (i.e. all the eigenvalues). However, we can start from a simpler question: can two different shape have the same eigenvalues?

(a) Using `FEniCs` complete the script `Drum_eigen.py`, and compute the spectrum of a circular drum, plot the first four eigenmodes (i.e., the ones with the lowest eigenvalue).

(b) Modify your code above and compute the eigenvalues for a squared drum. Plot a graph showing the spectrum of the circle and square drums.

(c) Find two isospectral domains. I.e., two domains with the same spectrum. Plot both domains, and present on a plot their spectrum.
Hint: google is your best friend to find such domains. You may need to use the `mshr` library to build those geometries.

C Consider the 1D poisson equation

$$\begin{cases} -u'' = f \text{ in } [0,L], \\ u(0) = u(1) = 0. \end{cases} \tag{3}$$

and a grid $\{x_i\}_{i=0}^{N}$, $0 = x_0 < x_1, ..., < x_N = L$.
Consider the Finite element formulation. Find $u_h \in V_{h,0}$, such that

$$\int_0^L u_h' v' = \int_0^L f v \qquad \forall v \in H_0^1(I). \tag{4}$$

where $I = [0,L]$.
Let $G_i \in H_0^1(\mathrm{I})$ satisfy

$$\langle v', \mathrm{G_i'} \rangle := \int_0^L v' G_i' = v(x_i) \quad \forall v \in H_0^1(\mathrm{I}) \tag{5}$$

where x_i is a given node $, i = 1, \ldots, N$.

(a) Prove that G_i is given by

$$G_i(x) = \begin{cases} (1-x_i)x & \text{for } 0 \leqslant x \leqslant x_i \\ x_i(1-x) & \text{for } x_i \leqslant x \leqslant 1 \end{cases} \tag{6}$$

(b) Show that $G_i \in V_{h,0}$.

(c) Show G_i is the Green's function for (3) associated with a delta function $\delta(x_i)$ at node x_i (G_i I.e. show that G_i satisfies

$$\begin{cases} -G_i'' = \delta(x_i) \text{ in } [0,L], \\ G_i(0) = G_i(1) = 0. \end{cases} \tag{7}$$

(d) Now, by choosing $v = e = u - u_h$ in (5), show that

$$e(x_i) = \langle e', G_i' \rangle = 0, \quad i = 1, \ldots, N \tag{8}$$

Thus, u_h is in fact exactly equal to u at the node points x_i. This somewhat surprising fact is a true one-dimensional effect due to the fact that the Green's function $G_i \in V_h$, and does not exist in higher dimensions. The technique of working with a Green's function in this way is however useful in proving for instance pointwise error estimates (maximum norm cstimates) in higher dimensions.

D Consider the problem

$$-\Delta u + u = f, \quad x \in \Omega, \quad u = 0, \quad x \in \partial\Omega$$

(a) Make a variational formulation of this problem in a suitable space V.

(b) Choose a polynomial subspace V_h of V and write down a finite element method based on the variational formulation.

(c) Deduce the Galerkin orthogonality

$$\int_\Omega \nabla(u-u_h) \cdot \nabla v + (u-u_h) v dx = 0, \quad \forall v \in V_h$$

(d) Derive the a priori error estimate

$$\|\nabla(u-u_h)\|_{L^2(\Omega)} + \|u-u_h\|_{L^2(\Omega)} \leq Ch \left\|D^2 u\right\|_{L^2(\Omega)}$$

where h is the maximum mesh size.