TMA026, Partial Differential Equations - second course
Computer Exercise 1
FEM, Convergence and Adaptivity

3 Bonus Points

March 19, 2019

Hand-in format The hand-in format is an informal report. Each task should be presented and then followed by its solution. Make sure that everything that is asked for is included such as tables, figures, conclusions, etc. The code should be presented in an appendix at the end of the report and it should be well-structured, well-commented, and easy to read and understand. Use of LaTeX and `anslistings.sty` is encouraged.

Introduction The purpose of this computer exercise is to get acquainted with the FEniCS software by performing a convergence study for an elliptic problem and adaptive mesh refinement based on a goal functional. For a domain $\Omega \subset \mathbb{R}^2$ whose boundary $\partial\Omega$ is partitioned into the three parts Γ_D, Γ_N, and Γ_R, consider the following boundary value problem: Find $u : \Omega \to \mathbb{R}$ such that:

$$\begin{cases} -\nabla \cdot (a\nabla u) + bu = f & \text{in } \Omega, \\ u = u_D & \text{on } \Gamma_D, \\ -a\frac{\partial u}{\partial n} = u_N & \text{on } \Gamma_N, \\ -a\frac{\partial u}{\partial n} = k(u - u_R) & \text{on } \Gamma_R, \end{cases} \tag{1}$$

for some non-constant functions a, b, f, k and some non-zero functions u_D, u_N, u_R.

Task 1 (Model Problem) Create an "exact solution" in the following way. Choose a rather simple domain Ω (rectangle, circle, square), a simple partition of the boundary $\partial\Omega = \Gamma_D \cup \Gamma_N \cup \Gamma_R$, and some simple coefficient functions a, b, k. Then choose a nontrivial function u and compute the corresponding data functions f, u_D, u_N, u_R. That is $f = -\nabla \cdot (a\nabla u) + bu$ and so on.

Task 2 (FEniCS-implementation) Derive a variational formulation for your problem (1). Implement and solve the discrete variational problem using FEniCS. Save both the exact solution u and the finite element solution u_h to file and make a visual comparison using, e.g., ParaView.

Implementation hint: Have a look at some suitable FEniCS demos to find out how functions, boundary conditions, forms, etc. are implemented.

Task 3 (Error Convergence) Compute the finite element solution u_h to your discrete problem in Task 2 for a sequence of meshes, and create a table of the errors $\|u-u_h\|_{L^2}$ and $|u-u_h|_1$ as functions of the mesh-size h. Determine the order of convergence. This kind of convergence study is not only an illustration of the theory but more importantly it is used as a test of the computer program: a minor programming error can result in a non-optimal convergence rate.

Implementation hint: Create a callable Python-function and put the relevant code from Task 2 in its function body. The Python-function should take as input a parameter for defining a mesh of some size and return the finite element solution and mesh-size or relevant computed norms. Useful commands: `mesh.hmax()`, `errornorm`, `numpy.polyfit`.

Task 4 (Devious Domain) Let the computational domain Ω be a circle sector with exterior angle α. Consider the following Dirichlet problem for Poisson's equation in Ω. In problem (1), let $\Gamma_D = \partial\Omega$, and $\Gamma_N = \Gamma_R = \emptyset$, which means that we can neglect k, u_N, and u_R. Furthermore, let $a = 1$, $b = 0$, $f = 1$, and $u_D = 0$. For some decreasing exterior angles, α_i, of the circle sector, e.g., $\alpha_i = \pi/2^i$ for $i = 0, 1, 2, \ldots, 7$, compute the order of convergence in the energy norm $|u - u_h|_1$ with respect to the number of degrees of freedom. Use the solution on the finest mesh as a reference solution in the convergence study. How is the order of convergence affected by the angle α?

Implementation hint: Import the FEniCS module `mshr` to create more interesting domains. Useful commands: `Circle`, `Polygon`, `generate_mesh`, `len(u.vector())`.

Task 5 (Adaptivity) Consider again the problem in Task 4 and take α to be the smallest angle used in the convergence study in Task 4, e.g., $\alpha = \pi/100$. There is a convenient adaptive mesh refinement framework in FEniCS based on supplying a goal functional. Use this framework to adaptively refine a coarse starting mesh for the problem considered. The goal functional can be the same as in the recommended demo below. For the sequence of meshes obtained, compute the order of convergence in the energy norm $|u - u_h|_1$ with respect to the number of degrees of freedom. Use the solution on the finest mesh as a reference solution in the convergence study. Compare the results with the results from Task 4 for the same angle α. For comparison one may plot the errors versus the degrees of freedom using, e.g., `matplotlib`, and/or put them in a table. Comment on the results.

Implementation hint: Have a look at the FEniCS demo "Auto adaptive Poisson Equation". Useful commands: `u.root_node()`, `u.leaf_node()`, `u.child()`, `u.depth()`, `len(u.vector())`.

Notice that the FEniCS book is freely available online[1]. The study of the first chapter, "A FEniCS tutorial", pages $1 - 73$, is strongly recommended.

[1] https://fenicsproject.org/book/