FSF3561/SF2561 **Finite Element Methods**

Lab Exercise 2

Aim: This lab concerns implementation of the finite element method (FEM) in 2D. The purpose of the lab is to get hands-on experience of implementing a FEM program.

Examination: The lab can be done individually or in groups of two. This lab consist of a set of compulsory problems, is graded by pass/fail (1/0), and should be submitted in time for the deadline.

Format of submissions: Each group has to submit a written report in pdf-format including a first page with: name, email address, and educational program for all group members. The report should be written such that a person taking a similar course in FEM should be able to read and understand the problem and your solution from the report. Please include references to books, websites, etc that you used to complete the lab assignment. Note that it is not allowed to copy existing code on the internet or from another group. A tar-archive of the source files needed to test the program and reproduce the results should be provided. You may write your program in Matlab, Octave, or Python.

For students taking FSF3561: For **Lab 3,** you should: Choose a scientific paper where a finite element method is used or presented, read the paper, and reproduce part of the results from the paper or use the presented finite element method to solve a different problem (consult with the teacher). The paper you choose may be related to your research. You don't have to use Matlab for this problem, you may use a software such as Comsol Multiphysics or FEniCS.
Note: Please consult with me before you choose a project!

To begin, consider the two-dimensional Poisson equation with zero boundary conditions given by

$$-\nabla(k(\mathbf{x})\nabla u) = f(\mathbf{x}), \quad \mathbf{x} \in \Omega \tag{1}$$

$$u = 0, \quad \mathbf{x} \in \partial\Omega. \tag{2}$$

Here, $\mathbf{x} = (x_1, x_2)$ and $\Omega = [0,1] \times [0,1]$, a unit square.

We will be constructing a general finite element code for given functions $k(x)$ and $f(x)$, as well as a general domain.

1. Derive a weak formulation for this problem.
2. Construct a mesh. You may write your own routine or, for example, call the Matlab function `initmesh` which uses a Delaunay triangulation algorithm. The call: `[p, e, t] = initmesh(g,'hmax',0.5)` gives a triangular mesh using the 2D geometry specification `g` where the maximum edge length of the triangles will not exceed 0.5. In the course book FEM-TIA, p.49, you find a routine for creating `g` when the geometry is a rectangle. You can view your generated mesh typing `pdemesh(p, e, t)`. See also https://se.mathworks.com/help/pde/ug/initmesh.html.
3. Define your finite dimensional space. Write functions that return the basis functions and the gradients of the basis functions.
4. To be able to compute integrals with general $k(\mathbf{x})$ and $f(\mathbf{x})$:
   (a) Implement an appropriate quadrature rule
   (b) Explain your choice of quadrature rules
5. Write a routine that assembles the stiffness matrix and load vector. You should loop through all the elements (e.g. triangles) once, compute the local element matrices (use your quadrature rule from Exercise 4), and set up the local-to-global mapping to get the global matrices.
6. Take $k(\mathbf{x}) = 1$ and $f(\mathbf{x}) = 40\pi^2 \sin(2\pi x_1)\sin(4\pi x_2)$. The exact solution is $u(\mathbf{x}) = 2\sin(2\pi x_1)\sin(4\pi x_2)$.
   (a) Find and plot the finite element approximation, $u(\mathbf{x})$, for some mesh size h. (Note that your program should also work for other coefficients $k(\mathbf{x})$ and force terms $f(\mathbf{x})$.)
   (b) Compute the L^2-norm of the error, $e_h = u - u_h$, for 4 different mesh sizes.
   (c) Study the order of convergence of the approximation, u_h, with respect to the mesh size. Verify the theoretical second order convergence, i.e verify that
   $$\|e_h\|_{L^2(\Omega)} \leq Ch^p$$
   with $p = 2$.
7. Take $f(\mathbf{x}) = 4 \cdot (500)^2(1 - 500 \cdot r^2)e^{-500r^2}$, with $r = \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2}$.
   (a) Determine the finite element approximation, u_h.
   (b) Compare the finite element approximation with the exact solution, $u(\mathbf{x}) = 500e^{-500r^2}$, and show how the error changes as the mesh is refined for 3 refinements (4 different meshes).

(c) Determine the location of the maximal error.

(d) Now, utilize *a posteriori error estimates* to adapt the mesh in elements where the residual is large.

i. Mark all cells with at least one node inside the circle defined by: all $\mathbf{x} = (x_1, x_2)$ such that $(x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.05^2$, for refinement.

ii. Compute the residual $R(u_h) = f + \Delta u_h$. Use the approximation $\Delta u_h \approx \Delta_h u_h$, where $\Delta u_h \approx \Delta_h u_h$ is the discrete Laplacian of the approximation.

iii. Compare the L^2-norm of the error. Use 2 different mesh refinement algorithms: 1) Three uniform mesh refinements (done in 7b); 2) Five local mesh refinements with red-green mesh refinement, where in each step you refine 50% of the cells with the largest residual, $R(u_h)$. How many nodes are used in each step of the algorithm in the two approaches? Which approach is the most efficient in terms of using as few nodes as possible to obtain as low error as possible? Plot the final meshes for the two algorithms.

(e) Discuss how to efficiently obtain an accurate finite element approximation. (You don't need to implement your method.)

8. Modify your finite element approximation to be able to handle Neumann boundary conditions. That is, for the regions of the boundary when $x_1 = 0$ and $x_1 = 1$, change the zero boundary condition to a homogeneous Neumann boundary condition:

$$\nabla u \cdot \hat{n} = 0,$$

with $\hat{n} = (n_1, n_2)$ being the unit outward pointing normal to the boundary.

(a) Test your code for when the source term is

$$f(\mathbf{x}) = 5\pi^2 \cos(\pi x_1) \cos(2\pi x_2).$$

In this case, the exact solution is given by

$$u(\mathbf{x}) = \cos(\pi x_1) \cos(2\pi x_2).$$

(b) Plot the finite element approximation, $u_h(\mathbf{x})$, and study the order of convergence.

9. Change the geometry to a circle with radius $r_0 = 0.75$ centered at $(x_c, y_c) = (0, 0)$. You need to change G in the call to initmesh. Take $\alpha = 9$, $f(x) = -4$, $u(x) = \frac{1}{16}$, $\forall x \in \partial\Omega$. The exact solution is $u(x, y) = \frac{(x-x_c)^2+(y-y_c)^2}{\alpha}$.

(a) Plot the finite element approximation u_h for $h = 0.1$.

(b) Compute the L^2-norm of the error $e_h = u - u_h$ for three different mesh sizes. In this example, the boundary of the domain is not approximated exactly. The convergence will depend on how you refine your mesh. You should be able to achieve second order convergence.

(c) Describe how one can increase the accuracy of the computed solution and achieve a higher order approximation? You don't need to compute a higher order approximation but specify what you need to change in your code.