MATH 575 ASSIGNMENT 4

In this assignment, you will use the program FEniCS to numerically solve several partial differential equations. To do this assignment, it will be helpful to read at least the first part of the FEniCS tutorial document, which can be found at

`http://fenicsproject.org/documentation/tutorial/`

and if possible, to install FEniCS on your computer. Instructions for installing FEniCS can be found at

`http://fenicsproject.org/`

If you are unable to install FEniCS on your computer, you will be able to use FEniCS on a computer in a public Mathematics Department lab.

1a. Use the code `poisson_convergence1.py`, found at

`http://www.math.rutgers.edu/~falk/math575/fenics-codes.html`

to obtain the finite element solution using continuous, piecewise linear elements of the boundary value problem

$$-\Delta u = f \quad \text{in} \ \ \Omega, \qquad u = 0 \quad \text{on} \ \ \partial\Omega,$$

where Ω is the unit square, the true solution $u = \sin(\pi x)\sin(2\pi y)$ and $f = 5\pi^2 u$. Print out and hand in the study of convergence in L^2 and H^1 produced by the program.

To run this program on a computer with FEniCS installed, and the program `poisson_convergence1.py` in your current directory, type

`python poisson_convergence1.py`

1b. Modify the code to produce analogous results for approximation by continuous, piecewise quadratic elements. Print out and hand in the study of convergence in L^2 and H^1 produced by your program.

2a. Use the code `poisson_convergence2.py`, found at

`http://www.math.rutgers.edu/~falk/math575/fenics-codes.html`

to obtain the finite element solution using continuous, piecewise linear elements of the boundary value problem

$$-\Delta u = f \quad \text{in} \ \ \Omega, \qquad u = 0 \quad \text{on} \ \ \partial\Omega,$$

where Ω is the unit square, and $f = [(x-1/3)^2 + (y-1/3)^2]^p$. For each of the choices $p = -.9$, $p = -.5$, and $p = -.1$, print out and hand in the study of convergence in L^2 and H^1 produced by the program. In this case the true solution is not known exactly, so we compute the errors by comparing to the finite element solution on a very high mesh with high degree elements.

2b. In this problem, we determine for which values of p, $f \in L^2(\Omega)$. Since f cannot be integrated exactly on Ω, we proceed as follows. (i) Show that $(x-1/3)^2 + (y-1/3)^2 \le 1$ on Ω and hence $\|f\|_{L^2(\Omega)} \le 1$ for $p \ge 0$. (ii) For $-\infty < p < 0$, let $B = \{(x,y) : (x-1/3)^2 + (y-1/3)^2 < 1/9\}$ and $\Omega \setminus B$ be the set of points in Ω that are not in B. By finding a lower bound

on $(x-1/3)^2+(y-1/3)^2$ on $\Omega \setminus B$, show that $\|f\|_{L^2(\Omega\setminus B)} < \infty$. (iii) From (ii), we know $f \in L^2(\Omega)$ if and only if $f \in L^2(B)$. By changing to polar coordinates: $x-1/3 = r\cos\theta$, $y-1/3 = r\sin\theta$, find $\|f\|_{L^2(B)}$ exactly in terms of p, and then determine for which values of p, $f \in L^2(\Omega)$.

2c. How does the result of 2b help explain your numerical results? Note that if $f \in L^2(\Omega)$, $u \in H^2(\Omega)$.

3. In this problem, we demonstrate the importance of adaptive mesh refinement for problems whose solutions are rapidly changing in some part of the domain.

3a. Use the code `adaptive_poisson.py` and the mesh file `l-shape-mesh.xml`, found at

`http://www.math.rutgers.edu/~falk/math575/fenics-codes.html`

to obtain the finite element solution using continuous, piecewise linear elements of the boundary value problem

$$-\Delta u = 1 \quad \text{in} \quad \Omega, \qquad u = 0 \quad \text{on} \quad \partial\Omega,$$

where Ω is the L-shaped domain formed from the union of the three squares $(-1,0)\times(0,1)$, $(0,1)\times(0,1)$, and $(0,1)\times(-1,0)$.

Note that to run the program, you may need to hit the ENTER key after each new mesh to get the program to continue.

Hand in a copy of the final refined mesh. This can be done by typing `i` in the mesh window. A .png file is produced which can be converted to a .pdf file using the (Linux) `convert` command. One can write the mesh directly to the file `finalmesh` using the command

`viz_mesh.write_ps('finalmesh', format='eps')` This produces a file in .eps format, which can be converted to .pdf using the (Linux) command `ps2pdf`. Also hand in the mesh level, number of triangles, and the maximum and minimum values of h used in the final mesh. These numbers are produced by the program.

3b. Modify the computer program to use only uniform mesh refinement. Note that the command `mesh=refine(mesh)` with no additional arguments will do uniform refinement. The Mark step will no longer be needed. Hand in the same items as in 3a.