UC Berkeley Math 228B, Spring 2020: Problem Set 4

Due April 23

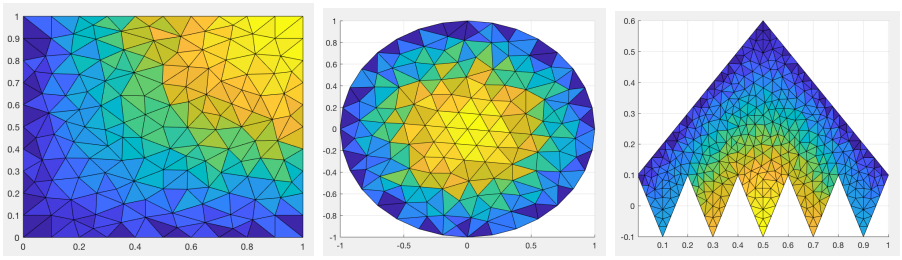
1. Implement a Matlab function with the syntax

```
u = fempoi(p,t,e)
```

that solves Poissons's equation $-\nabla^2 u(x,y) = 1$ on the domain described by the unstructured triangular mesh `p,t`. The boundary conditions are homogeneous Neumann ($n \cdot \nabla u = 0$) except for the nodes in the array `e` which are homogeneous Dirichlet ($u = 0$).

Show your results (test your code) on the following examples:

```
% Square, Dirichlet left/bottom
pv = [0,0; 1,0; 1,1; 0,1; 0,0];
[p,t,e] = pmesh(pv, 0.1, 0);
e = e(p(e,1) < 1e-6 | p(e,2) < 1e-6);
u = fempoi(p,t,e);

subplot(2,2,1);
trisurf(t,p(:,1),p(:,2),u);

% Circle, all Dirichlet
n = 32; phi = 2*pi*(0:n)'/n;
pv = [cos(phi), sin(phi)];
[p,t,e] = pmesh(pv, 2*pi/n, 0);
u = fempoi(p,t,e);

subplot(2,2,2);
trisurf(t,p(:,1),p(:,2),u);

% Generic polygon geometry, mixed Dirichlet/Neumann
x = (0:.1:1)';
y = 0.1 * cos(10*pi*x);
pv = [x,y; .5,.6; 0,.1];
[p,t,e] = pmesh(pv, 0.04, 0);
e = e(p(e,2) > .6 - abs(p(e,1) - 0.5) - 1e-6);
u = fempoi(p,t,e);

subplot(2,2,3);
trisurf(t,p(:,1),p(:,2),u);
```



By inspecting the function `pmesh` you will notice that what is stored in `e` are all the indices to the boundary nodes on Γ. The lines in the Matlab code above that follow `pmesh` "redefine" `e` so that what is stored in the final array `e` are only the indices to the nodes that correspond to the Dirichlet portion of the boundary Γ.

2. Implement a Matlab function with the syntax

```
errors = poiconv(pv, hmax, nrefmax)
```

that solves the all-Dirichlet Poisson problem (Poisson equation with Dirichlet data everywhere) defined on the two polygons `pv` from Problem 1, using the mesh parameters `hmax` and `nref = 0,1,...,nrefmax`, where `nrefmax` denotes the level of refimnent. Consider the solution on the finest mesh the exact solution, and compute the max-norm of the errors at the nodes for all the other solutions (note that this is easy given how the meshes were refined – the common nodes appear first in each mesh). The output `errors` is a vector of length `nrefmax` containing all the errors.

Notice that with all-Dirichlet boundary conditions for the two polygons, you do not need to modify the array `e`, since the output of `pmesh` stored in `e` already contains the pointers to all the boundary nodes on Γ.

Test the function using the commands below, which makes a convergence plot and estimates the rates:

```
hmax = 0.15;
pv_square = [0,0; 1,0; 1,1; 0,1; 0,0];
pv_polygon = [0,0; 1,0; .5,.5; 1,1; 0,1; 0,0];

for pv = {pv_square, pv_polygon}
  errors = poiconv(pv{1}, hmax, 3)
  loglog(hmax ./ 2.^(0:2), errors)
  rate = log2(errors(end-1)) - log2(errors(end))
end
```

3. Consider the boundary value problem

$$u''''(x) = f(x) \equiv 480x - 120, \qquad \text{for } x \in (0,1) \tag{1}$$

$$u(0) = u'(0) = u(1) = u'(1) = 0 \tag{2}$$

(a) Derive the following Galerkin formulation for the problem (1)-(2) on some appropriate function space V_h: Find $u_h \in V_h$ such that

$$\int_0^1 u_h''(x)v''(x)\,dx = \int_0^1 f(x)v(x)\,dx, \qquad \forall v \in V_h. \tag{3}$$

(b) Define the triangulation $T_h = \{K_1, K_2\}$, where $K_1 = [0, \frac{1}{2}]$ and $K_2 = [\frac{1}{2}, 1]$, and the function space

$$V_h = \{v \in C^1([0,1]) : v\big|_K \in \mathbb{P}^3(K)\ \forall K \in T_h,\ v(0) = v'(0) = v(1) = v'(1) = 0\}. \tag{4}$$

Find a basis $\{\varphi_i\}$ for V_h. *Hint*: You will get Hermite polynomials of degree 3 on each element.

(c) Solve the Galerkin problem (3) using your basis functions. Plot the numerical solution $u_h(x)$ and the true solution $u(x)$.

Code Submission: Submit a zip-file on bCourses. The zip-file should contain all the codes, and a pdf file with the results and explanation of the results.