MATH 272A: Numerical PDE
HW5

University of California, San Diego

Zihan Shao

In this homework set, our main goal is to solve the 2D Poisson equation

$$\begin{cases} -\Delta u = f & \text{on } \Omega = (0,1)^2, \\ u = g & \text{on } \partial\Omega, \end{cases} \tag{1}$$

using the finite element method with continuous piecewise linear elements.

Code source: Codes for below numerical results can be found at https://github.com/EddyShao/272-numericalPDE.

1. (3 pts) Write a code that automatically generates a uniform finite element mesh for Ω with $2n^2$ triangles for a given positive integer n. The data structure corresponding to the mesh should include:

   (a) A node matrix of size (total number of vertices) $\times$ 2.

   (b) An element matrix of size (total number of triangles) $\times$ 3.

   (c) A `bdNode` vector of size (total number of vertices) which specifies which vertices are on the boundary (give a value 1 if a vertex is on the boundary, otherwise 0).

   You may number the vertices and triangles in the way shown in the following picture (take $n = 3$ as an example).

   To test your code, take $n = 5$ and print out the 7th row of the node matrix, the 8th row of the element matrix, and the 12th component of the `bdNode` vector.

   Solution. Below is the output of the tests

```
7th row of the node matrix:
        0    0.2000

8th row of the element matrix:
    9    15    16

12th component of the bdNode vector:
    1
```

2. Write a code to compute the local stiffness matrix $A_K = (a_{ij}^K)_{i,j=1}^3$ and the local load vector $F_K = (f_j^K)_{j=1}^3$ for a given triangle K with three vertices v_1, v_2, and v_3. To test your code, print out A_K and F_K for $v_1 = (0,0)^T$, $v_2 = (0.5, 0.5)^T$, and $v_3 = (0,1)^T$.

   Solution. Below is the output of the tests (Note that here the results for load vector F_K depends on f, where we use the f computed with u given below.)

```
Local stiffness matrix A_K:
    0.5000   -0.5000         0
   -0.5000    1.0000   -0.5000
         0   -0.5000    0.5000

Local load vector F_K:
    0.2350
    0.5711
    0.6330
```

3. Implement the finite element method with continuous piecewise linear elements (Algorithm 2) to solve equation (1). Use the exact solution $u(x_1, x_2) = (x_1 + x_2)^2 \cos(x_1 + 2x_2)$ (with f and g computed accordingly) to test your code. Test your code for $h = 1/4, 1/8, 1/16, 1/32$ (corresponding to $n = 4, 8, 16, 32$). You may plot the numerical solution u_h to see whether it looks like the exact solution u.

   For a quantitative study on the order of convergence, we want to compare the difference between u_h and u in the two norms:

$$\|u - u_h\|_{L^2} = \left(\int_\Omega |u(x) - u_h(x)|^2 \, dx \right)^{1/2}$$

   and

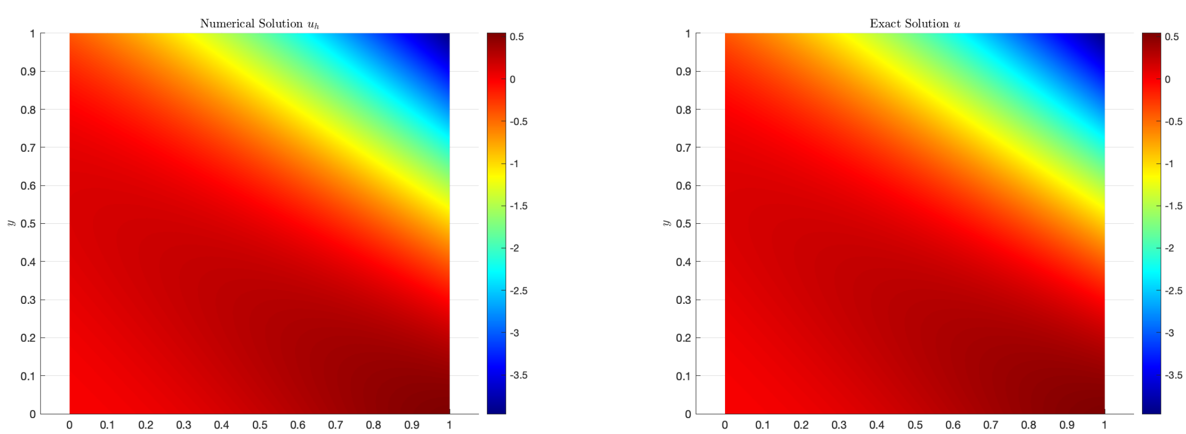
$$|u - u_h|_{H^1} = \left(\int_\Omega (\nabla u - \nabla u_h) \cdot (\nabla u - \nabla u_h) \, dx \right)^{1/2}.$$

   To evaluate these norms approximately, we can approximate the integral over each triangle K by the following rule:

$$\int_K v(x) \, dx \approx \frac{|K|}{3} \sum_{i=1}^{3} v(m_i),$$

   where $|K|$ is the area of K and $\{m_i\}_{i=1}^3$ are the three midpoints of the edges of K.

   Solution. We implement the finite element method based on Algorithm 2 in the lecture. We plot the numerical solution ($n = 64$) and exact solution to check if they look the same (See Fig.1)



(a) Numerical Solution u_h

(b) Exact Solution u

Figure 1: Comparison of Numerical and Exact Solutions ($n = 64$)

We also offer quantitative results. Note that when doing the quadrature, we use the same discretization used for numerical solutions (i.e. the triangle meshes) for simplicity concern (See Tab.1).

n	h	$\|u - u_h\|_{L^2}$	$|u - u_h|_{H^1}$
4	1/4	0.100371	1.079165
8	1/8	0.026010	0.546943
16	1/16	0.006565	0.274450
32	1/32	0.001645	0.137349

Table 1: L_2 and H_1 (semi) Norm Errors for Different Values of n

Log-Log error-h plot for both norms is shown in Fig.2.

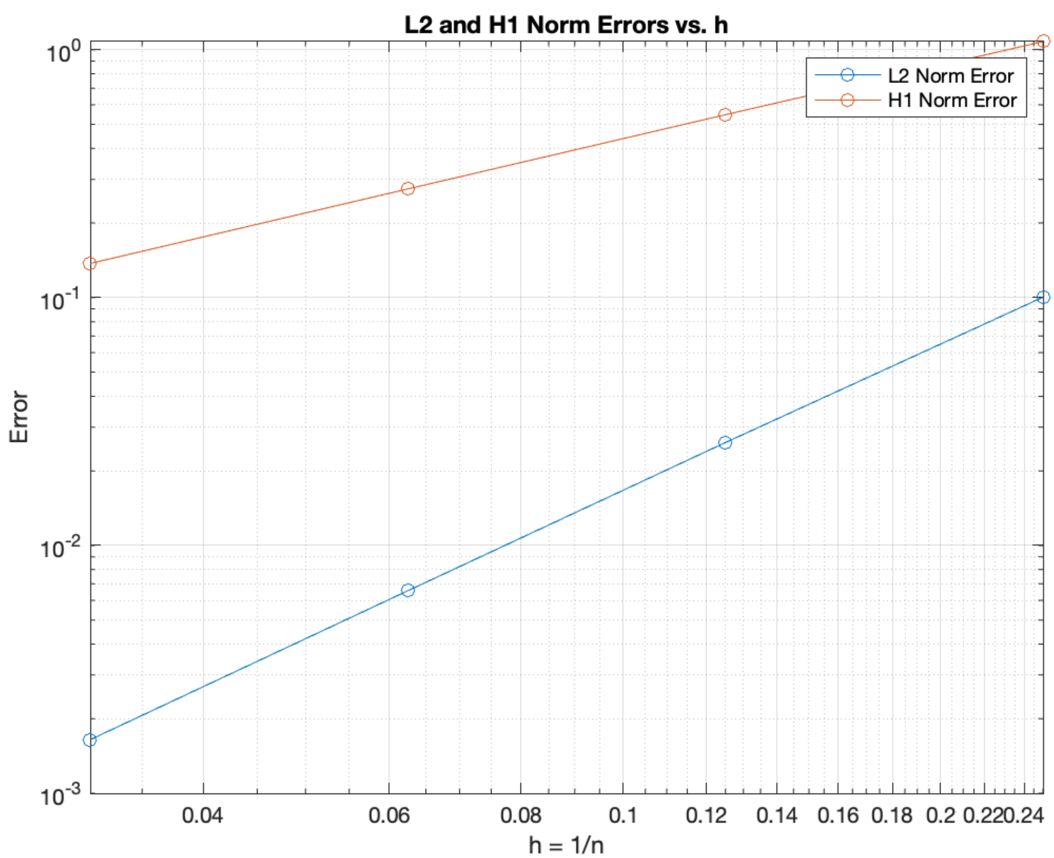


Figure 2: Log-Log error-h plot for both norms

With least square estimation, we validte that $\|u - u_h\|_{L^2} = O(h^2)$ and $|u - u_h|_{H^1} = O(h)$.