MATH5004 LAB 9
2D Poisson on Unit Circle with Finite Element Method

Consider the following Poisson equation over the unit circle,

$$-u_{xx} - u_{yy} = 4,$$

where u_{xx} is the second x derivative and u_{yy} is the second y derivative. This has known solution

$$u(x, y) = 1 - x^2 - y^2$$

which can be verified by plugging this value of U into the equation above giving the result 4 = 4 .

Although we know the solution we will still use the Finite Element Method to solve this problem and compare the result to the known solution.

The first thing that Finite Elements requires is a mesh for the 2D region bounded by the arbitrary 2D shape.

In order to do this we will be using a mesh generation tool implemented in MATLAB called *distmesh*.

```
fd=@(p) sqrt(sum(p.^2,2)) -1;
[p,t] = distmesh2d(fd,@huniform,0.2,[-1,-1;1,1],[]);
```

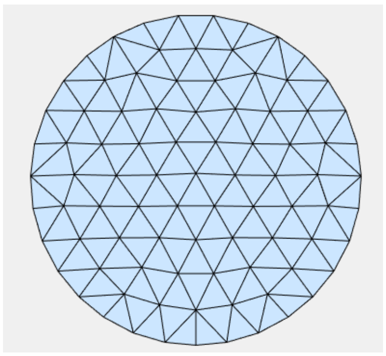


Fig 1. *Triangular mesh of a unit circle from the distmesh tool*

The values [p,t] returned from the distmesh2d command contain the coordinates of each of the nodes in the mesh and the list of nodes for each triangle. Distmesh also has a command for generating a list of boundary points b from [p,t],

```
e = boundedges(p,t);
b = unique(e);
```

The Finite Element method from the first example requires p, t and b as inputs. We will now modify this first example and to use p, t and b generated by distmesh for the region bounded by the unit circle. This can be accomplished by replacing the mesh generation code from the first part of **femcode.m** with the mesh creation commands from distmesh. The **modified code** is shown below and produced the result in Figure 2.

```
% femcode2.m
% Input:  N = number of nodes, T = number of triangles
fd=@(p) sqrt(sum(p.^2,2))-1;
[p,t]=distmesh2d(fd,@huniform,0.2,[-1,-1;1,1],[]);
              % p lists x,y coordinates of N nodes, t lists triangles by 3 node numbers
b=unique(boundedges(p,t));
N=size(p,1);T=size(t,1);
K=sparse(N,N);           % [K,F] = assemble(p,t) % K and F for any mesh of triangles:
F=zeros(N,1);            %linear phi's
for e=1:T                % integration over one triangular element at a time
  nodes=t(e,:);          % row of t = node numbers of the 3 corners of triangle e
  Pe=[ones(3,1),p(nodes,:)]; % 3 by 3 matrix with rows=[1 xcorner ycorner]
  Area=abs(det(Pe))/2;   % area of triangle e = half of parallelogram area
  C=inv(Pe);             % columns of C are coeffs in a+bx+cy to give phi=1,0,0 at nodes

  grad=C(2:3,:);         % now compute 3 by 3 Ke and 3 by 1 Fe for element e
  Ke=Area*grad'*grad;    % element matrix from slopes b,c in grad
  Fe=Area/3*4;           % integral of phi over triangle is volume of pyramid: f(x,y)=4
                         % multiply Fe by f at centroid for load f(x,y):
                         % one-point quadrature! Centroid would be
                         % mean(p(nodes,:)) = average of 3 node coordinates

  K(nodes,nodes)=K(nodes,nodes)+Ke;  % all T element matrices now assembled into K
  F(nodes)=F(nodes)+Fe;              % all vectors now assembled into F

end
K(b,:)=0; K(:,b)=0; F(b)=0; % put zeros in boundary rows/columns of K and F
K(b,b)=speye(length(b),length(b)); % put I into boundary submatrix of K
Kb=K; Fb=F;                 % Stiffness matrix Kb (sparse format) and load vector Fb
U=Kb\Fb;                    % The FEM approximation is U_1 phi_1 + ... + U_N phi_N
% Plot U(x,y) with values U_1 to U_N at the nodes
trisurf(t,p(:,1),p(:,2),0*p(:,1),U,'edgecolor','k','facecolor','interp');
view(2),axis([-1 1 -1 1]),axis equal, colorbar
```

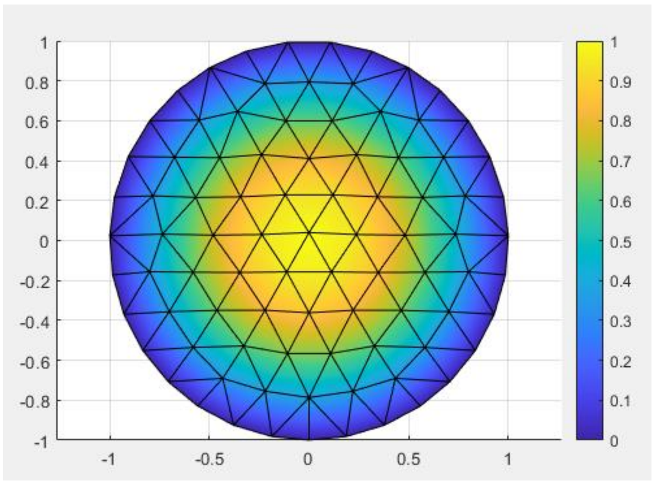


Fig 2. Solution of the Poisson's equation on a unit circle

We can compare this result to the known solution $u(x, y) = 1 - x^2 - y^2$ to the Poisson equation which is plotted below for comparison. We generated this plot with the following MATLAB commands knowing the list of mesh node points p returned by distmesh2d command.

```
u = 1 - p(:,1).^2 - p(:,2).^2
trisurf(t,p(:,1),p(:,2),0*p(:,1),u,'edgecolor','k','facecolor','interp');
view(2),axis([-1 1 -1 1]),axis equal,colorbar
```

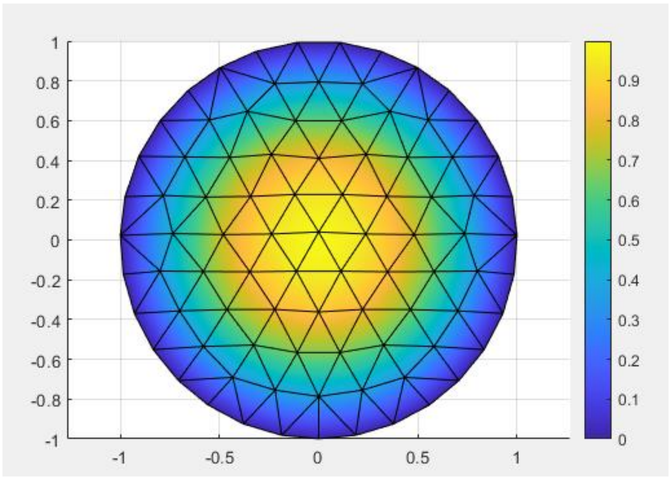
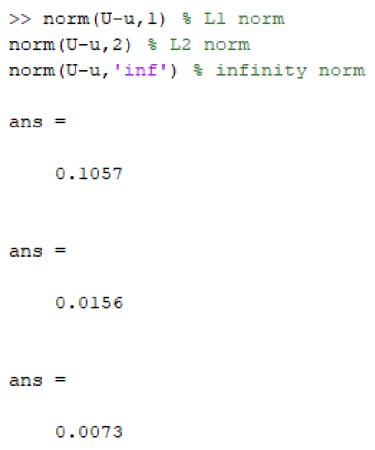


Fig3. *The analytical solution*

Next we can calculate the difference between the Finite Element approximation and the known solution to the Poisson equation on the region bounded by the unit circle. Using MATLAB norm command we can calculate the L_1 norm, L_2 norm and infinity norm of the difference between approximated and known solution (U – u), where capital U is the Finite Element approximation and lowercase u is the known solution.

```
norm(U-u,1) % L1 norm
norm(U-u,2) % L2 norm
norm(U-u,'inf') % infinity norm
```



```
>> norm(U-u,1) % L1 norm
norm(U-u,2) % L2 norm
norm(U-u,'inf') % infinity norm

ans =

    0.1057


ans =

    0.0156


ans =

    0.0073
```

When we repeat this experiment using a finer mesh resolution we get the following results with mesh resolution values of 0.2, 0.15 and 0.1 which are passed as a parameter to the **distmesh2d** command when generating the mesh. As can be seen from the table below, a mesh with a finer resolution results in a Finite Element approximation that is closer to the true solution.

Mesh Resolution	Node Count	L1 Norm	L2 Norm	L infinity Norm
0.20	88	0.10568	0.015644	0.0073393
0.15	162	0.1064	0.012909	0.0032610
0.10	362	0.083466	0.0073064	0.0016123

--------------------------End of Document --------------------------