MATH 517 PROJECT 5: LAPLACE EQUATION

Turn in your Matlab/Scilab script on Blackboard by the end of Sunday 11/23.

Goal: obtain a numeric approximation to the solution of the PDE $u_{xx} + u_{yy} = 0$ with Dirichlet boundary conditions in a rectangle $0 < x < 5, 0 < y < 3$.

Answer the following questions, based on output:

(1) In what parts of the rectangle the gradient of the solution has largest magnitude?
(2) What is the value of the solution at the center of rectangle? (look up the appropriate entry of u after the program finishes)
(3) Is the value of the center closer to the average of u on the vertical sides, or on horizontal sides? Can you suggest why?

You can include the answers as comments when submitting the file on Blackboard, or as comments within the script.

Method: Use space step $h = 0.1$ to discretize the variables x and y.

Set up the matrix of u values (so far, filled with zero), computing its size using the time and space steps, similar to previous projects. Example:

```
h = 0.1;
x = 0:h:5;
y = 0:h:3;
xsize = length(x);
ysize = length(y);
u = zeros(ysize, xsize);
```

(Note that in matrix notation, the vertical coordinate comes first.)

For the boundary values, use the last four digits of your SUID number (does not matter in what some order):

```
for i=1:ysize
    u(i,1) =    ...
    u(i,xsize) =   ...
end
for j=1:xsize
    u(1,j) =      ...
    u(ysize,j) =    ...
end
```

Preparation for visualizing the solution can look like this in Matlab:

```
clf()
hold on
colormap(hot(128));
scale = 120/max(u(:));
```

and like this in Scilab:

```
clf()
f = gcf();
f.color_map = hotcolormap(128)
scale = 120/max(u);
```

The scale coefficient will be used to translate the values of u into colors on the "hot" color map.

The main computation consists in a double loop that replaces every entry (except those on the boundary) by the average of its neighbors:

```
for i=2:ysize-1
    for j=2:xsize-1
        u(i,j)=(u(i-1,j)+u(i+1,j)+u(i,j-1)+u(i,j+1))/4;
    end
end
```

The calculated values should then be displayed using `image(u*scale)` in Matlab or `Matplot(u*scale)` in Scilab.

Repeat this compute-and-plot process at least 500 times (this will require one more `for` loop around the double loop). To see how the process converges to the solution, it may be necessary to slow down this outer loop by inserting a pause command after the plot command, e.g., `pause(0.01)` in Matlab or `xpause(1e4)` in Scilab.