Math 371 Winter 2013 Homework 7 due: Tuesday April 2

This exercise concerns the two-dimensional BVP discussed in class. Consider a metal plate on the unit square $D = \{(x,y) : 0 \le x,y \le 1\}$. The plate temperature $\phi(x,y)$ satisfies the Laplace equation $\phi_{xx} + \phi_{yy} = 0$ on D (where $\phi_x = \frac{\partial \phi}{\partial x}$, etc.), with boundary conditions $\phi(x,1) = 1, \phi(x,0) = \phi(0,y) = \phi(1,y) = 0$. This means there are no heat sources inside the plate, and one side of the plate is kept at a high temperature while the other three sides are kept at a low temperature. Solve for the temperature $\phi(x,y)$ inside the plate using the finite-difference scheme $(D_+^x D_- + D_+^y D_-^y) w_{ij} = 0$ with mesh size $h = \frac{1}{n+1}$, for $h = \frac{1}{4}, \frac{1}{8}, \frac{1}{16}$. This yields a linear system $A_h w_h = f_h$, where $w_h = \{w_{ij}\}$ is the numerical solution with components $w_{ij} \approx \phi(x_i, y_j)$. The mesh points are given by $x_i = ih, y_j = jh, i,j = 0 : n+1$. The finite-difference equations can be written in component form as

$$\frac{1}{h^2}(4w_{ij} - w_{i+1,j} - w_{i-1,j} - w_{i,j+1} - w_{i,j-1}) = f_{ij},$$

and in this exercise they are to be solved by Jacobi's method,

$$\frac{1}{h^2}(4w_{ij}^{(k+1)} - w_{i+1,j}^{(k)} - w_{i-1,j}^{(k)} - w_{i,j+1}^{(k)} - w_{i,j-1}^{(k)}) = f_{ij},$$

where $w_{ij}^{(k)}$ is the numerical solution at step k. Do not form the full matrix A_h (because it's sparse and that would be inefficient).

Implementation Details (a MATLAB pseudocode is on back of this sheet)

1. To keep the code simple, the solution vector $w_h = \{w_{ij}\}$ is coded as a matrix of dimension $(n+2) \times (n+2)$ containing the unknown interior temperature values and the known boundary values. Let `w_new(i,j)` be the solution at step $k+1$ and let `w_old(i,j)` be the solution at step k. Since MATLAB doesn't accept zero indices, take `i=1:n+2, j=1:n+2`. The boundary values of the temperature correspond to indices `i=1,n+2, j=1,n+2`.

2. In the case of the two-point BVP in 1D, we put the temperature boundary values in the right hand side vector f_h. However in a 2D problem, it's more convenient to keep the boundary values in the solution vector w_h. Therefore the boundary values and interior values of w_h are set at the initial step and the interior values are updated at every new step. The interior values are set to zero at the initial step.

3. stopping criterion : $||r_k||/||r_0|| \le 10^{-4}$, where $r_k = f_h - A_h w_h^{(k)}$ is the residual at step k

Present the results as follows. Include a copy of the code and give a brief writeup.

a) For each value of h, plot the computed temperature w_{ij} at the final step (including the boundary values) using a contour plot and a mesh plot (type `help contour` and `help mesh` for instructions).

b) Present the following results in a table. column 1: h, column 2: number of iterations needed to reach the stopping criterion, column 3: heat flux through bottom edge of the plate. The heat flux is the integral of the normal derivative of the temperature along an edge of the plate; e.g. on the bottom edge, the heat flux is $F = \int_0^1 \phi_y(x,0)dx \approx \sum_{i=0}^{n+1} D_+^y w_{i0} \cdot h$.

Does the heat flux through the bottom edge converge as $h \to 0$? At what rate?

c) What is the value of the temperature at the corners of the plate in the limit $h \to 0$? Explain your answer.

announcement The second quiz is on Thursday, April 4. It will be 20 minutes long and start promptly at 12:10pm. The quiz is closed book and will cover material up to and including the previous class. Please bring a calculator to do arithmetic. The best way to prepare is to review the lecture notes and homework problems.

```
function m371bvp2d
% Steady state temperature on the unit square.
clear; clf;
tol = ...; % set tolerance for stopping criterion
for icase=1:3
    n = 2^(icase+1)-1; h = 1/(n+1); % set mesh size
    x = 0:h:1; y = 0:h:1; % create x and y arrays for plots
% initialize solution and residual arrays
    w_new = zeros(n+2,n+2);
    w_old = zeros(n+2,n+2);
    res = zeros(n+2,n+2);
% set nonzero boundary values
    for j = ...; w_new(...,...)  = ...; w_old(...,...)  = ...; end
% initialize control variables
    k = 0; ratio = 1;
% start iteration
    while ratio > tol
         k = k+1;
% compute residual vector
         for i = ...; for j = ...;
              res(i,j) = ...;
         end; end
% compute ratio of residual norms using Frobenius norm for convenience
         rn(k) = norm(res,'fro');
         ratio = rn(k)/rn(1);
% compute numerical solution
         for i = ...; for j = ...;
              w_new(i,j) = ...;
         end; end;
         w_old = w_new; % reset numerical solution for next step
         flux = ...; % compute heat flux, hint: use Matlab sum command
    end % end while
% store results for output
    table(icase,1) = h; table(icase,2) = k; table(icase,3) = flux;
% draw contour plot
    subplot(2,3,icase)
    contour(x,y,w_new); axis square
    string = sprintf('h=1/%d',n+1); title(string)
% draw surface plot
    subplot(2,3,3+icase)
    mesh(x,y,w_new)
    string = sprintf('h=1/%d',n+1); title(string)
end
table
```