HW 6, UCI, MAE 185, Problem 29.4

Nasser M. Abbasi

May 24, 2003

problem

Use Liebmann method (Gauss-Siedel) to solve for the temp of the heated plate shown. Employee overrelaxation with a value of 1.5 for the weighting factor and iterate to $\varepsilon_s = 1$ The plate has the lower left corner changes. See diagram.

Solution

The only difference in this problem is how to calculate T for the point next to the modified lower left corner. Looking at the diagram, the point can be solved by using the following modified 4-point difference equation:

$$T_0 = \frac{T_1}{\left(1+\frac{\beta_1}{\beta_2}\right)\left(1+\frac{\beta_1\beta_2}{\alpha_1\alpha_2}\right)} + \frac{T_3}{\left(1+\frac{\beta_2}{\beta 1}\right)\left(1+\frac{\beta_1\beta_2}{\alpha_1\alpha_2}\right)} + \frac{T_4}{\left(1+\frac{\alpha_2}{\alpha_1}\right)\left(1+\frac{\alpha_1\alpha_2}{\beta_1\beta_2}\right)} + \frac{T_2}{\left(1+\frac{\alpha_1}{\alpha_2}\right)\left(1+\frac{\alpha_1\alpha_2}{\beta_1\beta_2}\right)}$$

In our case, T_0 in the above equation is the same as point $T_{1,1}$. Using $\alpha_1 = 0.732, \alpha_2 = 1, \beta_1 = 0.732, \beta_2 = 1$ the above equation becomes

$$\begin{aligned} T_0 &= \frac{T_1}{\left(1+\frac{0.732}{1}\right)\left(1+\frac{0.732}{0.732}\right)} + \frac{T_3}{\left(1+\frac{1}{0.732}\right)\left(1+\frac{0.732}{0.732}\right)} + \frac{T_4}{\left(1+\frac{1}{0.732}\right)\left(1+\frac{0.732}{0.732}\right)} + \frac{T_2}{\left(1+\frac{0.732}{1}\right)\left(1+\frac{0.732}{0.732}\right)} \\ &= \frac{T_1}{3.464} + \frac{T_3}{4.73224043715847} + \frac{T_4}{4.73224043715847} + \frac{T_2}{3.464} \end{aligned}$$

replacing T_0, T_1, T_2, T_3, T_4 by $T_{1,1}, T_{1,0}, T_{0,1}, T_{1,2}, T_{2,1}$ results in

$$T_{1,1} = \frac{T_{1,0}}{3.464} + \frac{T_{1,2}}{4.73224043715847} + \frac{T_{2,1}}{4.73224043715847} + \frac{T_{0,1}}{3.464}$$

The calculations for one iteration are now given, then a Matlab function is used to solve this problem. Looking at the above grid. For point (1,1)

$$\begin{aligned} T_{1,1} &= \frac{T_{1,0}}{3.464} + \frac{T_{1,2}}{4.732} + \frac{T_{2,1}}{4.732} + \frac{T_{0,1}}{3.464} \\ &= \frac{0}{3.464} + \frac{0}{4.732} + \frac{0}{4.732} + \frac{75}{3.464} \\ &= 21.651 \end{aligned}$$

Now applying the overrelaxation step to improve convergence. Using $\lambda = 1.5$

$$\begin{aligned} T_{i,j} &= \lambda T_{i,j} + (1-\lambda) T_{i,j}^{old} \\ T_{1,1} &= (1.5)\, 21.651 + (1-1.5)\, 0 \\ &= 32.477 \end{aligned}$$

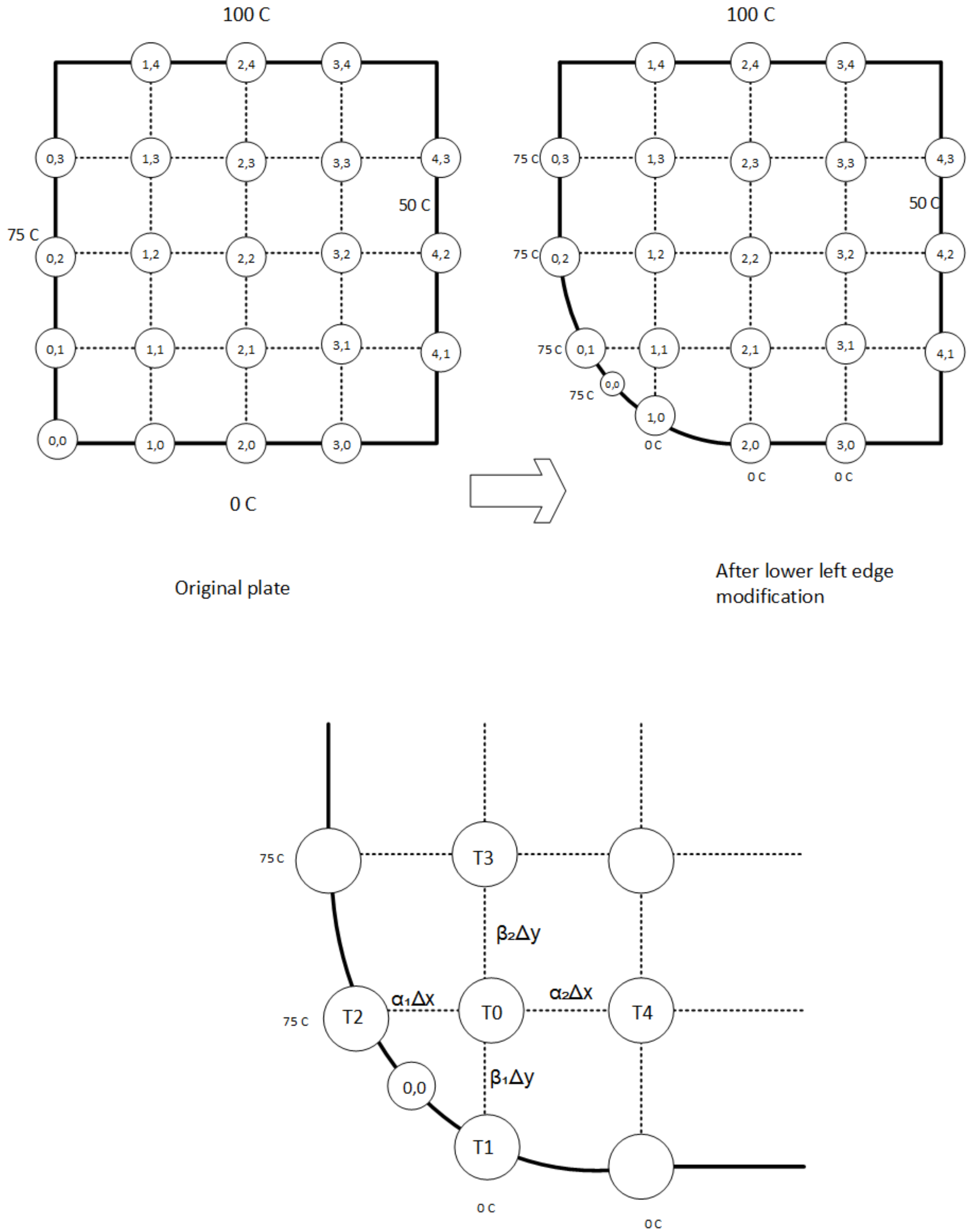

Figure 1: Initial state of system

For point (2,1):

$$
\begin{aligned}
T_{2,1} &= \frac{T_{3,1} + T_{1,1} + T_{2,2} + T_{2,0}}{4} \\
&= \frac{0 + 32.477 + 0 + 0}{4} \\
&= 8.119 \\
T_{2,1} &= (1.5)\,8.119 + (1-1.5)\,0 \\
&= 12.1789
\end{aligned}
$$

For point (3,1)

$$\begin{aligned}
T_{3,1} &= \frac{T_{4,1}+T_{2,1}+T_{3,2}+T_{3,0}}{4} \\
&= \frac{50+12.179+0+0}{4} \\
&= 15.545 \\
T_{3,1} &= (1.5)\,15.545+(1-1.5)\,0 \\
&= 23.317
\end{aligned}$$

For point (1,2)

$$\begin{aligned}
T_{1,2} &= \frac{T_{2,2}+T_{0,2}+T_{1,3}+T_{1,1}}{4} \\
&= \frac{0+75+0+21.651}{4} \\
&= 24.1628175 \\
T_{1,2} &= (1.5)\,24.163+(1-1.5)\,0 \\
&= 36.244
\end{aligned}$$

For point (2,2):

$$\begin{aligned}
T_{2,2} &= \frac{T_{3,2}+T_{1,2}+T_{2,3}+T_{2,1}}{4} \\
&= \frac{0+36.244+0+12.179}{4} \\
&= 12.106 \\
T_{2,2} &= (1.5)\,12.106+(1-1.5)\,0 \\
&= 18.159
\end{aligned}$$

For point (3,2)

$$\begin{aligned}
T_{3,2} &= \frac{T_{4,2}+T_{2,2}+T_{3,3}+T_{3,1}}{4} \\
&= \frac{50+18.159+0+23.317}{4} \\
&= 22.869 \\
T_{3,2} &= (1.5)\,22.869+(1-1.5)\,0 \\
&= 34.303
\end{aligned}$$

For point (1,3)

$$\begin{aligned}
T_{1,3} &= \frac{T_{2,3}+T_{0,3}+T_{1,4}+T_{1,2}}{4} \\
&= \frac{0+75+100+36.244}{4} \\
&= 52.811 \\
T_{1,3} &= (1.5)\,52.811+(1-1.5)\,0 \\
&= 79.217
\end{aligned}$$

For point (2,3)

$$\begin{aligned}
T_{2,3} &= \frac{T_{3,3}+T_{1,3}+T_{2,4}+T_{2,2}}{4}\\
&= \frac{0+79.217+100+18.159}{4}\\
&= 49.344\\
T_{2,3} &= (1.5)\,49.344+(1-1.5)\,0\\
&= 74.016
\end{aligned}$$

For point (3,3)

$$\begin{aligned}
T_{3,3} &= \frac{T_{4,3}+T_{2,3}+T_{3,4}+T_{3,2}}{4}\\
&= \frac{50+74.0157+100+34.303}{4}\\
&= 64.580\\
T_{3,3} &= (1.5)\,64.580+(1-1.5)\,0\\
&= 96.870
\end{aligned}$$

The maximum value above is $T_{3,3}=96.870$ C. We continue this process. The final solution is

0	100	100	100	0
75	81.001646887152	82.5961283099831	73.8902466305191	50
75	83.3811172847313	72.6541192747923	65.376274908799	50
75	78.0664013507249	80.1058182124657	61.9649723876403	50
0	75	75	75	0

I wrote a MATLAB function to solve this. Modifed the function written for problem 29.1 to adjust the calculations for the edge point. The rest of the calculation remained the same. This function accepts as input the number of points in the x-direction, the number of points in the y-direction, and the values of the dependent variable at the boundaries of the plate, 4 values, one for each side and the value λ, and a tolerance value ε_s and the values for $\alpha_1,\alpha_2,\beta_1,\beta_2$

The function returns back a matrix that contains the solution. (ie. it returns back the plate grid). Below it shows a run of the function, it shows how the plate is being filled up with updated values of the solution $T(x,y)$ one iteration at a time, until the approximate solution converges to the desired accuracy.

```
nx=3;ny=3;bot=0;right=50;top=100;left=75;
alpha1=0.732; beta1=0.732; alpha2=1; beta2=1; lambda=1.5; tol=1;
A=nma_laplaceRectDirchletBendCorner(nx,ny,bot,...
  right,top,left,alpha1,beta1,alpha2,beta2,lambda,tol)

Iteration number 1

A =

0                  100                 100                 100     0
75        80.7389398094688    75.1574791606524    97.5119081986143    50
75        40.3038394919169    19.6810046189376    34.8742760356524    50
75        32.4769053117783    12.1788394919169    23.3170648094688    50
0                    0                   0                   0     0

Iteration number 2
```

```
A =
0                   100                100                100    0
75     75.1739688975816   87.7566338750272   67.2776239070609   50
75     57.9583576455235   61.5417854514132   71.6662408086209   50
75     32.8741286155454   22.3626545205235   28.5553165538316   50
0                     0                  0                  0    0

Iteration number 3
A =
0                   100                100                100    0
75     85.6879649692193   76.5087839137823   70.3495371989236   50
75     65.9765645729996   68.0666800605088   50.7934804927617   50
75     41.4995613730355   38.1674215065933   45.6599650912896   50
0                     0                  0                  0    0

epsilonA = 12.270096594562

Iteration number 4

epsilonA = 0.0295038073522684

A =
0                   100                100                100    0
75     77.5382493269215   74.6626898881034   69.1674013454185   50
75     59.8756278094746   53.3011314683695   51.3458754636334   50
75     44.9471820332058   28.5664031443096    33.770694641448   50
0                     0                  0                  0    0
```

Now, I will show just the final solution A for increasing accuracy by making ε_s smaller and smaller to see how small a ε_s it will take to get a better solution.

```
A=nma_laplaceRectDirchletBendCorner(nx,ny,bot,right,...
  top,left,alpha1,beta1,alpha2,beta2,lambda,1)

A =
0                   100                100                100    0
75     77.5382493269215   74.6626898881034   69.1674013454185   50
75     59.8756278094746   53.3011314683695   51.3458754636334   50
75     44.9471820332058   28.5664031443096    33.770694641448   50
0                     0                  0                  0    0
```

```
A=nma_laplaceRectDirchletBendCorner(nx,ny,bot,right,...
    top,left,alpha1,beta1,alpha2,beta2,lambda,0.1)

A =
0                   100                100                100    0
75     77.5382493269215   74.6626898881034   69.1674013454185   50
75     59.8756278094746   53.3011314683695   51.3458754636334   50
75     44.9471820332058   28.5664031443096    33.770694641448   50
0                     0                  0                  0    0
```

```
A=nma_laplaceRectDirchletBendCorner(nx,ny,bot,right,...
  top,left,alpha1,beta1,alpha2,beta2,lambda,0.01)}
```

A =

0	100	100	100	0
75	78.4686673678791	76.0216532976758	69.5960159002225	50
75	62.8480329457962	56.0415823472379	52.3621580287638	50
75	41.9083470938005	32.9325184573665	33.8297814344369	50
0	0	0	0	0

Source code

```
function A=nma_laplaceRectDirchletBendCorner(nx,ny,bot,right,...
   top,left,alpha1,beta1,alpha2,beta2,lambda,tol)

% Function that solves the laplace PDE for rectangular region for
% Dirclet boundary conditions with the lower left corner bend per
% parameters alpha and beta.

%INPUT:
%  nx: number of grid points in the x-direction
%  ny: number of grid points in the y-direction
%  bot: bottom edge boundary value
%  right: right edge boundary value
%  top: top edge boundary value
%  left: left edge boundary value
%  alpha1: bend parameter. see problem notes for more info
%  alpha2: bend parameter. see problem notes for more info
%  beta1: bend parameter. see problem notes for more info
%  beta2: bend parameter. see problem notes for more info
%  lambda: the value lambda to use for the relaxation method of Liebmann
%  tol: the absolute tolerance to check for convergance.
%
%OUTPUT:
%  the solution matrix.
%
%
%Example run:
%  nx=3;ny=3;bot=0;right=50;top=100;left=75;
%  alpha1=0.732; beta1=0.732; alpha2=1; beta2=1; lambda=1.5; tol=1;
%  A=nma_laplaceRectDirchletBendCorner(nx,ny,bot,right,top,left,...
        alpha1,beta1,alpha2,beta2,lambda,tol)
%
%Author: Nasser M. Abbasi
%May 25, 2003

TRUE  = 1;
FALSE = 0;

A     = zeros(nx+2,ny+2);
A_old = A;

% apply dirchlet conditions
A(1,:)   = top;
A(:,1)   = left;
A(end,:) = bot;
```

```
A(:,end) = right;

%patch the 4 corner, just for cosmotics purposes
A(1,1)   = 0;
A(end,1) = 0;
A(1,end) = 0;
A(end,end) = 0;

A_old    = A;

converged = FALSE;
nIter     = 0;
while(~converged)
    nIter = nIter+1;
    %fprintf('Iteration number %d\n',nIter);

    for(i=2+nx-1:-1:2)
        for(j=2:1:2+ny-1)
            if(i==2+nx-1 & j==2) % edge node, handle special case.
                Term1 = A(i+1,j)/(  (1+beta1/beta2)*(1+ (beta1*beta2)/(alpha1*alpha2) ) );
                Term2 = A(i-1,j)/(  (1+beta2/beta1)*(1+ (beta1*beta2)/(alpha1*alpha2) ) ;
                Term3 = A(i,j+1)/(  (1+alpha2/alpha1)*(1+ (alpha1*alpha2)/(beta1*beta2) ) );
                Term4 = A(i,j-1)/(  (1+alpha1/alpha2)*(1+ (alpha1*alpha2)/(beta1*beta2) ) );

                A(i,j) = Term1+Term2+Term3+Term4;
            else
                A(i,j) = A(i,j+1) + A(i,j-1) + A(i-1,j) + A(i+1,j);
                A(i,j) = A(i,j)/4;
            end

            A(i,j) = lambda*A(i,j) + (1-lambda)*A_old(i,j);
        end
    end

    [max_row , max_row_index]     = max(A(2:end-1,2:end-1));
    [max_element , max_col_index] = max(max_row);

    pos = [max_row_index(1)+1,max_col_index+1];

    epsilonA = abs(A(pos(1),pos(2)) - A_old(pos(1),pos(2)))/A(pos(1),pos(2))*100;

    if(  epsilonA  < tol )
        converged = TRUE;
    end

    A_old = A;
    %A
    %epsilonA
end

A=A(2:end-1,2:end-1);
```