Computer Implementation 5.1 ***(Matlab)*** *Heat flow over L-shape using rectangular elements (p. 342)*

The analysis of two dimensional bounadry value problems using rectangular elements can be performed conveniently by writing three *Matlab* functions, one for defining the element $\boldsymbol{k}_k + \boldsymbol{k}_p$ and $\boldsymbol{r}_q$ vectors, one for evaluating natural boundary terms, and the third for computing the element solution.

MatlabFiles\Chap5\BVPRectElement.m

```
function [ke, rq] = BVPRectElement(kx, ky, p, q, coord)
% [ke, rq] = BVPRectElement(kx, ky, p, q, coord)
% Generates for a rectangular element for 2d BVP
% kx, ky, p, q = parameters defining the BVP
% coord = coordinates at the element ends

x1=coord(1,1); y1=coord(1,2);
x2=coord(2,1); y2=coord(2,2);
x3=coord(3,1); y3=coord(3,2);
x4=coord(4,1); y4=coord(4,2);
a = abs((x2 - x1))/2; b = abs((y4 - y2))/2;
kk = (kx*b)/(6*a)*[2, -2, -1, 1; -2, 2, 1, -1;
    -1, 1, 2, -2; 1, -1, -2, 2] + ...
    (ky*a)/(6*b)*[2, 1, -1, -2; 1, 2, -2, -1;
    -1, -2, 2, 1; -2, -1,1, 2];
kp = -((p*a*b)/9)*[4, 2,1, 2; 2, 4, 2, 1; 1, 2, 4, 2; 2, 1, 2, 4];
ke = kk + kp;
rq = a*b*q*[1; 1; 1; 1];
```

MatlabFiles\Chap5\BVPRectNBCTerm.m

```
function [ka, rb] = BVPRectNBCTerm(side, alpha, beta, coord)
% [ka, rb] = BVPRectNBCTerm(side, alpha, beta, coord)
% Generates kalpha and rbeta when NBC is specified along a side
% side = side over which the NBC is specified
% alpha and beta = coefficients specifying the NBC
% coord = coordinates at the element ends

x1=coord(1,1); y1=coord(1,2);
x2=coord(2,1); y2=coord(2,2);
x3=coord(3,1); y3=coord(3,2);
x4=coord(4,1); y4=coord(4,2);
ka = zeros(4); rb = zeros(4,1);
switch (side)
case 1
    lm=[1,2]; L=abs(x2-x1);
```

```
case 2
   lm=[2,3]; L=abs(y3-y2);
case 3
   lm=[3,4]; L=abs(x4-x3);
case 4
   lm=[4,1]; L=abs(y4-y1);
end
ka(lm, lm) = -(1/3)*alpha*L/2*[2, 1; 1, 2];
rb(lm) = beta*L/2*[1; 1];
```

MatlabFiles\Chap5\BVPRectResults.m

```
function results = BVPRectResults(coord, dn)
% results = BVPRectResults(coord, dn)
% Computes element solution for a rectangular element for 2D BVP
% coord = nodal coordinates
% dn = nodal solution
% Output variables are u and its x and y derivatives at element center
x1=coord(1,1); y1=coord(1,2);
x2=coord(2,1); y2=coord(2,2);
x3=coord(3,1); y3=coord(3,2);
x4=coord(4,1); y4=coord(4,2);
s = 0; t = 0;
a = abs((x2 - x1))/2; b = abs((y4 - y2))/2;
u = 1/(4*a*b)*[(a - s)*(b - t), (a + s)*(b - t), ...
      (a + s)*(b + t), (a - s)*(b + t)]*dn;
dudx= 1/(4*a*b)*[-b + t, b - t, b + t, -b - t]*dn;
dudy = 1/(4*a*b)*[-a + s, -a - s, a + s, a - s]*dn;
results=[u, dudx, dudy];
```

Using these functions now we consider solution of the heat flow problem.

MatlabFiles\Chap5\LShapedHeatEx.m

```
% Heat flow through an L-shaped body
h = 55; tf = 20; htf = h*tf;
kx = 45;  ky = 45; Q = 5*10^6; ql = 8000; t0 = 110;
nodes = 1.5/100*[0, 2; 1,2; 2, 2; 0, 1; 1, 1; 2, 1; 3, 1; 4, 1;
   0, 0; 1, 0; 2, 0; 3, 0; 4, 0];
lmm = [9, 10, 5, 4; 4, 5, 2, 1; 10, 11, 6, 5;
   5, 6, 3, 2; 11, 12, 7, 6; 12, 13, 8, 7];
debc = [9:13]; ebcVals=t0*ones(length(debc),1);
dof=length(nodes); elems=size(lmm,1);
K=zeros(dof); R = zeros(dof,1);
% Generate equations for each element and assemble them.
for i=1:elems
   lm = lmm(i,:);
```

```
    [k, r] = BVPRectElement(kx, ky, 0, Q, nodes(lm,:));
    K(lm, lm) = K(lm, lm) + k;
    R(lm) = R(lm) + r;
end
% Compute and assemble NBC contributions
lm = lmm(1,:);
[k, r] = BVPRectNBCTerm(4, 0, ql, nodes(lm,:));
K(lm, lm) = K(lm, lm) + k;
R(lm) = R(lm) + r;

lm = lmm(2,:);
[k, r] = BVPRectNBCTerm(4, 0, ql, nodes(lm,:));
K(lm, lm) = K(lm, lm) + k;
R(lm) = R(lm) + r;
[k, r] = BVPRectNBCTerm(3, -h, htf, nodes(lm,:));
K(lm, lm) = K(lm, lm) + k;
R(lm) = R(lm) + r;

lm = lmm(4,:);
[k, r] = BVPRectNBCTerm(2, -h, htf, nodes(lm,:));
K(lm, lm) = K(lm, lm) + k;
R(lm) = R(lm) + r;
[k, r] = BVPRectNBCTerm(3, -h, htf, nodes(lm,:));
K(lm, lm) = K(lm, lm) + k;
R(lm) = R(lm) + r;

lm = lmm(5,:);
[k, r] = BVPRectNBCTerm(3, -h, htf, nodes(lm,:));
K(lm, lm) = K(lm, lm) + k;
R(lm) = R(lm) + r;

lm = lmm(6,:);
[k, r] = BVPRectNBCTerm(3, -h, htf, nodes(lm,:));
K(lm, lm) = K(lm, lm) + k;
R(lm) = R(lm) + r;

% Nodal solution
d = NodalSoln(K, R, debc, ebcVals)
results=[];
for i=1:elems
    results = [results; BVPRectResults(nodes(lmm(i,:),:), d(lmm(i,:)))];
end
results
```

```
>> LShapedHeatEx

d =
```

```
  154.9620
  151.2283
  148.6731
  145.4325
  142.5208
  134.8705
  122.4359
  121.0878
  110.0000
  110.0000
  110.0000
  110.0000
  110.0000


results =

  1.0e+003 *

    0.1270   -0.0971    2.2651
    0.1485   -0.2215    0.6079
    0.1243   -0.2550    1.9130
    0.1443   -0.3402    0.7503
    0.1193   -0.4145    1.2435
    0.1159   -0.0449    0.7841
```