MATLAB EXERCISE 2.10 **FD-based MATLAB code – iterative solution.** Write a computer program in MATLAB for the iterative finite-difference analysis of a coaxial cable of square cross section, Fig.2.4 (from the book), based on Eq.(2.16) (from the book). Assume that $a = 1$ cm, $b = 3$ cm, $V_a = 1$ V, and $V_b = -1$ V. Plot the results for the distribution of the potential and the electric field intensity in the space between the conductors, and the surface charge density on the surfaces of conductors, taking the grid spacing to be $d = a/10$ and the tolerance of the potential $\delta_V = 10^{-8}$ V. Compute the total charge per unit length of the inner and the outer conductor, taking $d = a/N$ and $N = 2, 3, 5, 7, 9, 10, 12,$ and 25, respectively. *(ME2_10.m on IR)*

SOLUTION:

Simulation results for the distribution of the potential and the electric field intensity in the space between the conductors and for the charge distribution of the conductors of the cable are plotted using MATLAB functions `surf`, `quiver`, and `plot`, respectively, and the plots, for $d = a/10$ and $\delta_V = 10^{-8}$ V, are shown in Figs.S2.5–S2.7. The computed total per-unit-length charges of conductors, taking $d = a/N$ and $N = 2, 3, 5, 7, 9, 10, 12,$ and 25, respectively, are tabulated in Table S2.1.

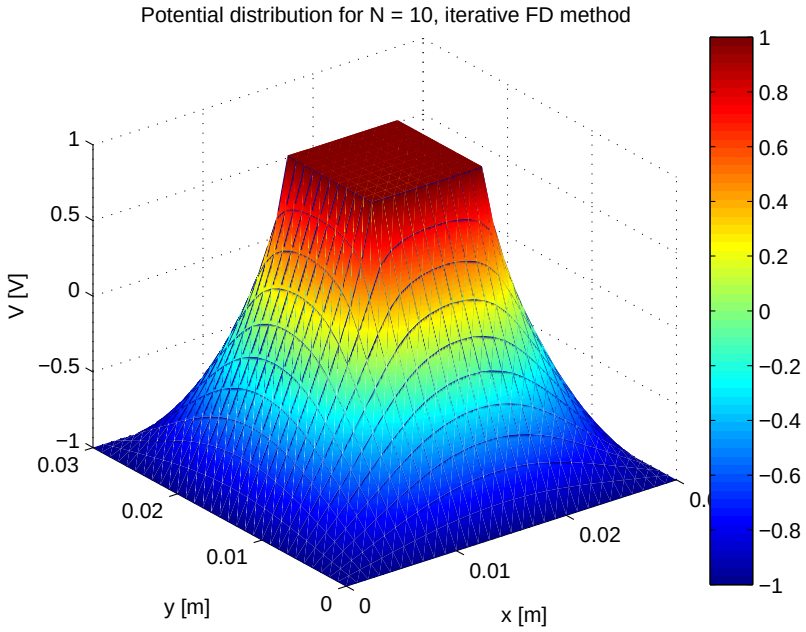

Figure S2.5 Electric potential in the space between the conductors of the coaxial cable of square cross section in Fig.2.4 (from the book) – results by the iterative finite-difference technique implemented in MATLAB; for MATLAB Exercise 2.10.

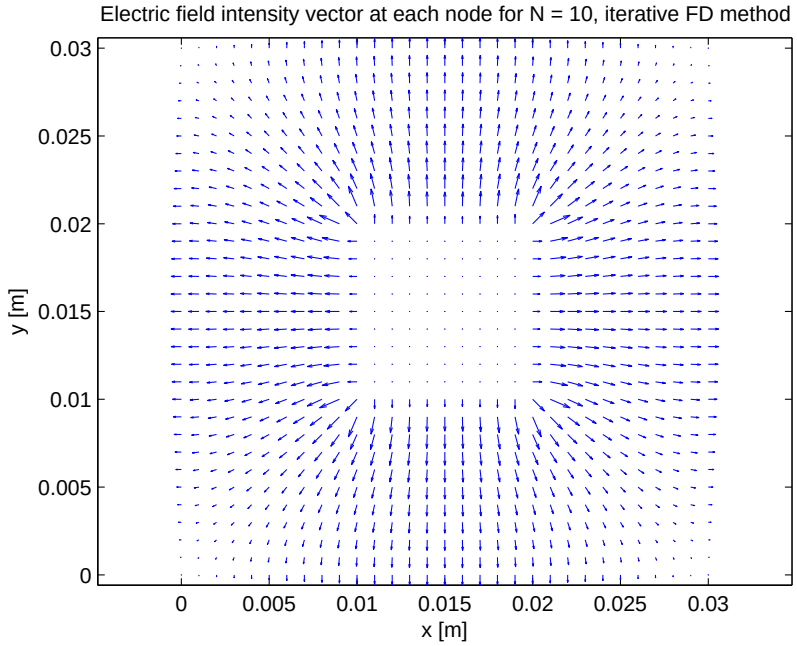

Figure S2.6 Electric field intensity vector corresponding to the potential in Fig.S2.5; for MATLAB Exercise 2.10.

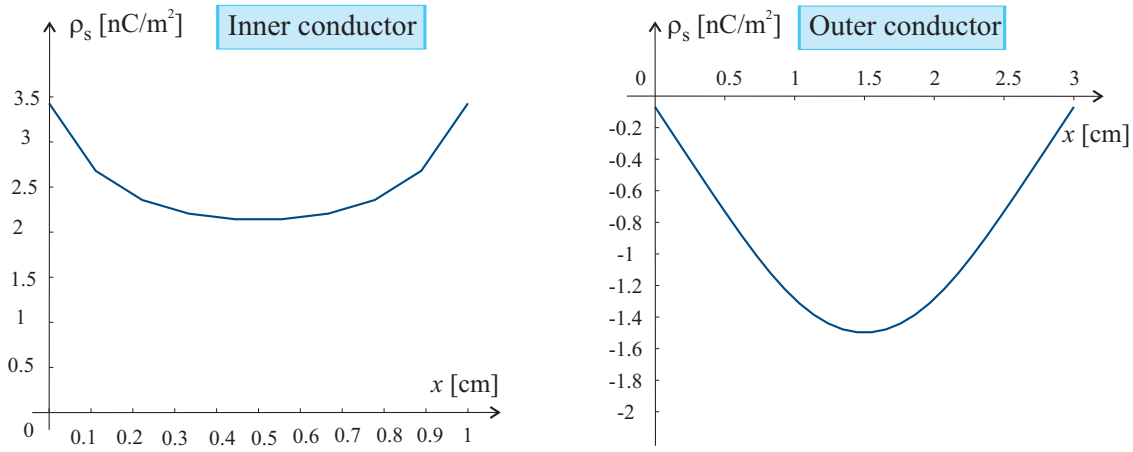

Figure S2.7 Computed surface charge density on the surfaces of conductors of the square coaxial cable in Fig.2.4 (from the book); for MATLAB Exercise 2.10.

Table S2.1 MATLAB FD results for conductor p.u.l. charges; for MATLAB Exercise 2.10.

N	Q'_{inner}[C/m]	Q'_{outer}[C/m]
2	0.8854×10^{-10}	-1.0625×10^{-10}
3	0.9391×10^{-10}	-1.0983×10^{-10}
5	0.9847×10^{-10}	-1.1084×10^{-10}
7	1.0066×10^{-10}	-1.1083×10^{-10}
9	1.0201×10^{-10}	-1.1074×10^{-10}
10	1.0252×10^{-10}	-1.1069×10^{-10}
12	1.0332×10^{-10}	-1.1060×10^{-10}
25	1.0579×10^{-10}	-1.1032×10^{-10}

```
%
%   Book: MATLAB-Based Electromagnetics (Pearson Prentice Hall)
%   Author: Branislav M. Notaros
%   Instructor Resources
%   (c) 2011
%
%   This MATLAB code or any part of it may be used only for
%   educational purposes associated with the book
%
%
%

%  FD computer program - iterative solution

clear all;
close all;

format long;
a = 0.01;
b = 0.03;
Va = 1;
Vb = -1;
deltaV = 10^(-8);
EPS0 = 8.8542*10^(-12);
maxIter = 500000;

N = [2 3 5 7 9 10 12 25];
%N = 10;
%m = 1;
for m = 1 : length(N)
    d = a/N(m);
   %number of inner nodes
    N1 = N(m) + 1;
    %number of outer nodes
    N2 = b/a *N(m) + 1;
    V = ones(N2,N2)*(Va+Vb)/2;
    %outer boundary
    V(1,:) = Vb; V(:,1) = Vb; V(:,N2)=Vb; V(N2,:) = Vb;
    %inner boundary
    V((N2-N1)/2+1:(N2+N1)/2,(N2-N1)/2+1:(N2+N1)/2) = Va;


    iterationCounter = 0;
    maxError = 2*deltaV;
    while (maxError > deltaV)&&(iterationCounter < maxIter)
        Vprev = V;
        for i = 2 : N2-1
            for j = 2 : N2-1
                if V(i,j)~=Va
                     V(i,j)=(Vprev(i-1,j)+ Vprev(i,j-1)+Vprev(i+1,j)+Vprev(i,j+1))/4;
                end;
            end;
```

```
        end;
        difference = max(abs(V-Vprev));
        maxError = max(difference);
       iterationCounter = iterationCounter + 1;
     end;

   [x,y]= meshgrid(0:d:b);
   [Ex,Ey] = gradient(-V,d,d);

sigmaOut = zeros(1,N2-1);
sigmaIn = zeros(1,N1-1);
for i = 1:N2-1
sigmaOut(i) = EPS0/2/d*(3/2*V(1,i)-2*V(2,i)+1/2*V(3,i)+...
    3/2*V(1,i+1)-2*V(2,i+1)+1/2*V(3,i+1));
end;
k = (N2-N1)/2 + 1;

for i = k:(N2 + N1)/2 -1
sigmaIn(i-k+1)=EPS0/2/d*(3/2*V(k,i)-2*V(k-1,i)+1/2*V(k-2,i)+...
    3/2*V(k,i+1)-2*V(k-1,i+1) + 1/2*V(k-2,i+1));
end;

Qouter(m) = 4*d*sum(sigmaOut);
Qinner(m) = 4*d*sum(sigmaIn);


Cout(m)= Qouter(m)/(Vb - Va);
Cin(m) = Qinner(m)/(Va - Vb);

 figure(4*m - 3);
 quiver (x,y,Ex,Ey); xlabel('x [m]'); ylabel('y [m]');
 title(['Electric field intensity vector at each node for N = '...
     ,num2str(N(m)),', iterative FD method']);axis equal;

   figure(4*m - 2);
   surf(x,y,V); shading interp; colorbar;
   xlabel('x [m]'); ylabel('y [m]'); zlabel('V [V]');
   title(['Potential distribution for N = ', num2str(N(m)),...
       ', iterative FD method']);

   figure(4*m-1);
   dinner = a/(length(sigmaIn)-1);
   innerCond = 0:dinner:a;
   plot(innerCond, sigmaIn);
   xlabel('x [m]'); ylabel('\rho_{in} [C/m^2]');

   figure(4*m);
   douter = b/(length(sigmaOut)-1);
   outerCond = 0:douter:b;
   plot(outerCond,sigmaOut);
   xlabel('x [m]'); ylabel('\rho_{out} [C/m^2]');
```

```
    clear Vstart V ;
   error(m) = (Qinner(m) + Qouter(m))/Qouter(m)*100;
end;
```