MATLAB EXERCISE 2.12 **FD-based MATLAB code – direct solution.** Write the main MATLAB code for the direct finite-difference analysis of a square coaxial cable, Fig.2.4, based on Eq.(2.15) (from the book) – see the previous MATLAB exercise. Compute and plot the same quantities as in MATLAB Exercise 2.10, and compare the results obtained by the two programs. *(ME2_12.m on IR)*

SOLUTION:

Plots of the computed potential, field, and charge distributions, using the direct FD technique, look identical to those in Figs.S2.5–S2.7, and the total p.u.l. charges of conductors are given in Table S2.2, so an excellent agreement with the results by the iterative FD technique (MATLAB Exercise 2.10) is observed.

Table S2.2 Conductor p.u.l. charges obtained by the direct FD code; for MATLAB Exercise 2.12.

N	$Q'_{\rm inner}$[C/m]	$Q'_{\rm outer}$[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	1.0580×10^{-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 - direct solution

clear all;
close all;

a = 0.01;
b = 0.03;
Va = 1;
Vb = -1;
EPS0=8.8542*10^(-12);
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 node
    N2 = b/a *N(m) + 1;
    %K = N2^2; % total nodes

    Vstart = zeros(N2,N2);
    Vstart(1,:) = Vb;
    Vstart(:,1) = Vb;
    Vstart(N2,:) = Vb;
    Vstart(:,N2) = Vb;

    lim1 = (N2-N1)/2 + 1;
    lim2 = (N2+N1)/2;
    Vstart(lim1:lim2,lim1:lim2)=Va;

    % A,C matrix
    [A,C]= mACfd(Vstart,N2);
    V = inv(A)*C;

   %V2D
    for i = 1:N2
    V2D(i,:) = V((i-1)*N2+1:i*N2);
    end;
```

```
   [x,y]= meshgrid(0:d:b);
   [Ex,Ey] = gradient(-V2D,d,d);

   sigmaOut = zeros(1,N2-1);
   sigmaIn = zeros(1,N1-1);

for i = 1:N2-1
sigmaOut(i) = EPS0/2/d*(3/2*V2D(1,i)-2*V2D(2,i)+1/2*V2D(3,i)+...
    3/2*V2D(1,i+1)-2*V2D(2,i+1)+1/2*V2D(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*V2D(k,i)-2*V2D(k-1,i)+1/2*V2D(k-2,i)+...
    3/2*V2D(k,i+1)-2*V2D(k-1,i+1) + 1/2*V2D(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)),', direct FD method']);axis equal;

   figure(4*m - 2);
   surf(x,y,V2D); shading interp; colorbar;
   xlabel('x [m]'); ylabel('y [m]'); zlabel('V [V]');
   title(['Potential distribution for N = ', num2str(N(m))...
       ', direct 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]');

error(m) = (Qinner(m)+Qouter(m))/Qouter(m)*100;
clear Vstart V V2D sigmaIn sigmaOut;
end;
```