Electromagnetics 2FH4

MATLAB Set (14) – Finite Difference Solution of Laplace Equation

Instructor: Dr. M.H Bakr

Department of Electrical and Computer Engineering
McMaster University

Matthew Jarzynowski – jarzynom – 400455803

MATLAB Set (14) – Finite Difference Solution of Laplace Equation

Problem

Consider the configuration of conductors and potentials shown in Figure 14.7. Write a MATLAB program that solves Laplace equation in the area bounded by the conductors. Plot the contours of the voltage and the lines of the electric field.

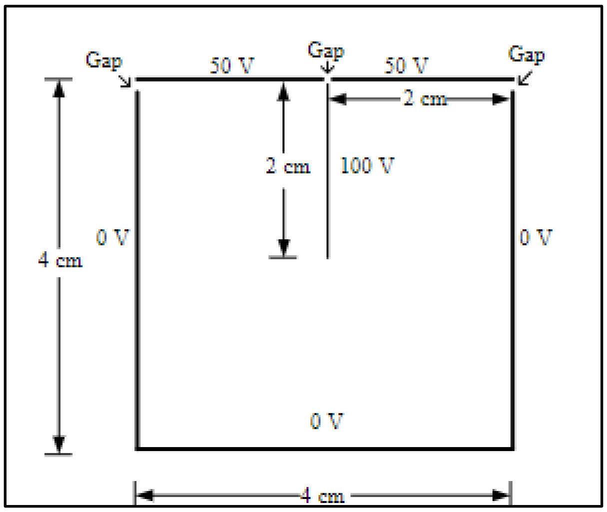

Figure 14.7. Exercise configuration

Solution

Consider the following derived from lab manual solution, in terms of MATLAB,

```
clc; % Clear the command bar
clear; % Remove all prior variables

Vo = 0; % Volatge on the outer conductor, just 0
V = 100; % Voltage on the inner conductor, the "protruding rod"

% Number of Points, in Each Direction
NumX = 100;
NumY = 100;

Unknowns = NumX * NumY; % Total number of unknowns, 10000

% Matrix of Coefficients, Right, Left, and Bottom Relative
A = zeros(Unknowns, Unknowns);
B = zeros(Unknowns, 1);
```

Continued…

```
i_right = (NumX)/2; % Right
j_left = (NumY)/2; % Left


EqCount = 1; % Counter for the equations

% For all rows...
for i=1:NumX

    % For all columns...
    for j=1:NumY
        % For points on the "rod", V = 100
        if(i == i_right && j >= j_left)
            A(EqCount, EqCount) = 1;
            B(EqCount, 1) = V;

        % For points before the "rod", V = 50
        elseif(i < i_right) && (j == NumY)
            A(EqCount, EqCount) = 1;
            B(EqCount, 1) = 50;

        % For points after the "rod", V = 50
        elseif(i > i_right) && (j == NumY)
            A(EqCount, EqCount) = 1;
            B(EqCount, 1) = 50;

        else
            A(EqCount, EqCount) =-4;

            % First column...
            if(j == 1)
                B(EqCount, 1) = B(EqCount, 1) - Vo;
            else % Store the coeffcient of the left-most point
                A(EqCount, EqCount - 1) = 1;
            end

            % The last column...
            if(j == NumY)
                B(EqCount, 1) = B(EqCount, 1) - Vo;
            else % Store the coeffcient of the right-most point
                A(EqCount, EqCount + 1) = 1;
            end

            % The first row...
            if(i == 1)
                B(EqCount, 1) = B(EqCount, 1) - Vo;
            else % Store the coeffcient of the top-most point
                A(EqCount, EqCount - NumX) = 1;
            end
```

Continued…

```
% The last row...
            if(i == NumX)
                B(EqCount, 1) = B(EqCount, 1) - Vo;
            else % Store the coeffcient of the bottom-most point
                A(EqCount, EqCount + NumX) = 1;
            end
        end
    EqCount = EqCount + 1;
    end
end

V = A\B; % The volatge vector, relative to point (X,Y)

% Converts the values into a rectangular matrix
Vs = reshape(V, NumX, NumY);


% Surface Figure (1)
surf(Vs); % Plot the "surface" figure
colormap turbo;
title('Surface Voltage Plot')
figure;

% Contour Figure (2)
[C,h] = contour(Vs); % Plot the "contour" figure
set(h, 'ShowText', 'on', 'TextStep', get(h, 'LevelStep')*2);
colormap turbo;
title('Voltage Contour Plot')
figure;

% Electric Field Figure (3)
contour(Vs);
[px,py] = gradient(Vs);
colormap turbo;
title('Electric Field Map')

% Plot the field map, using the gradient
hold on, quiver(-px, -py), hold off
```

Continued…

With the following plots generated,

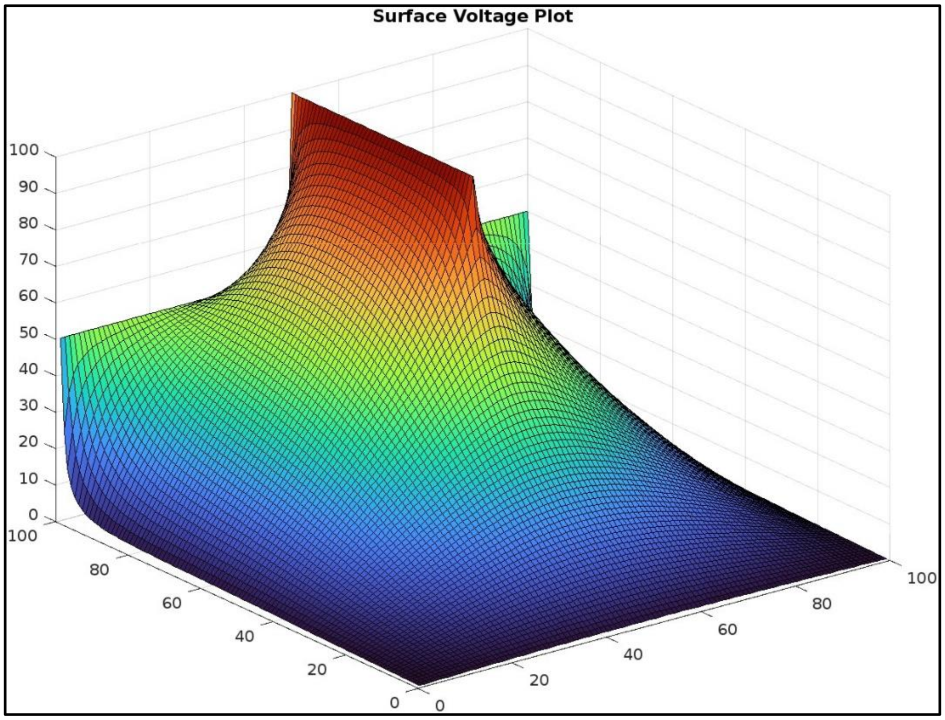

Figure 1. Surface Conductor Plot

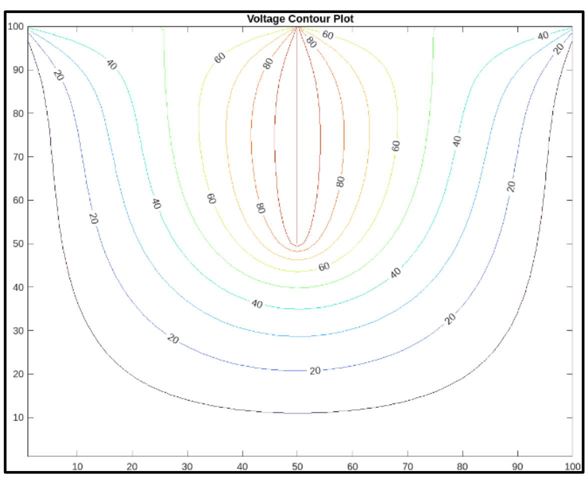

Figure 2. Voltage Contour Plot

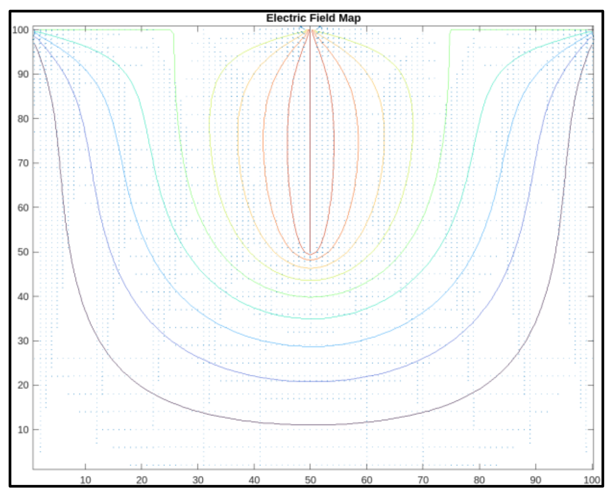

Figure 2. Electric Field Map

Which, per the results, are accurate, in relation to general intuition of the exercise and sample.