Lab Report 3

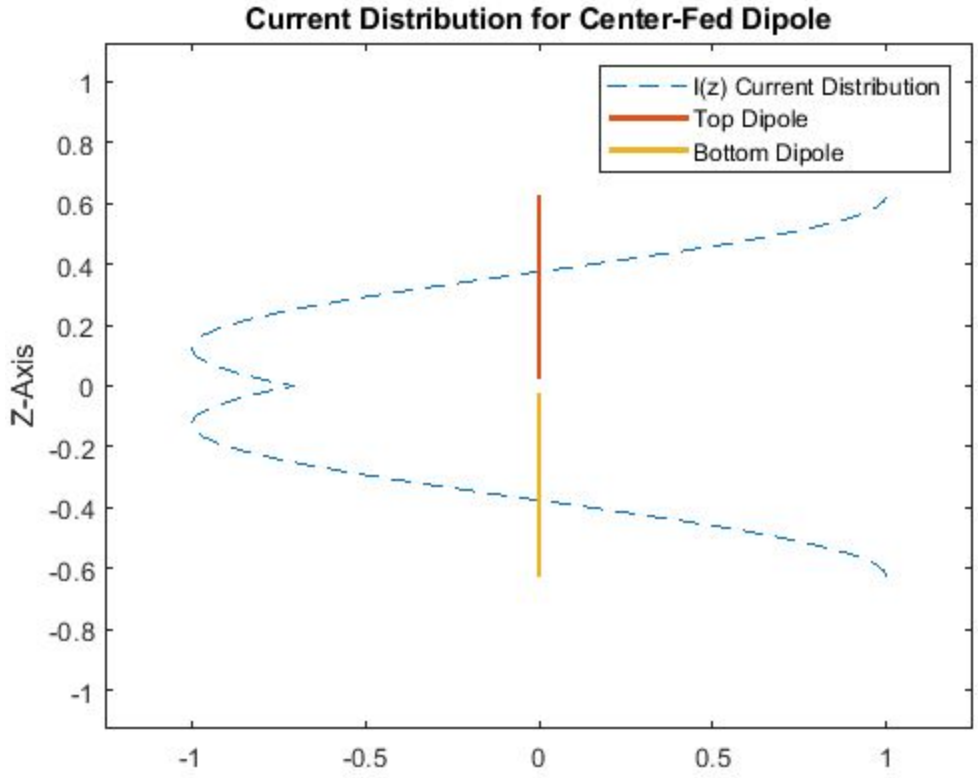
Ryan Nand
ECE332L

1. Current Distribution

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Ryan Nand
% 5/23/2020
% Dipole Antenna Current Distribution
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
lambda = 1;             % Chosen Lambda Value
k = 2*pi/lambda;        % k constant equation
l = lambda*1.25;        % Length in terms of Lambda
n = 2000;               % Number of points evaluated in plot
z = linspace(-l/2, l/2, n);     % Equal spaced points for current line
z2 = linspace(.02, l/2, n);     % Equal spaced points for top dipole
z3 = linspace(-l/2, -.02, n);   % Equal spaced points for bottom dipole
I_o = 1;                        % Current amplitude
x = zeros(1, n);                % preallocation for current line array
top_dipole = zeros(1, n);       % Preallocation for top dipole array
bottom_dipole = zeros(1, n);    % Preallocation for bottom dipole array
for i = 1:n             % Loop to calculate current line
    if z(i) < 0     % Equation to use if less than zero
        x(i) = I_o*sin(k*((l/2)-z(i)));
    else            % Equation to use if >= to zero
        x(i) = I_o*sin(k*((l/2)+z(i)));
    end
end
% Plot results from calculation
p=plot(x, z, '--', top_dipole, z2, bottom_dipole, z3);
% Plot details
legend('I(z) Current Distribution', 'Top Dipole', 'Bottom Dipole')
title('Current Distribution for Center-Fed Dipole')
ylabel('Z-Axis')
ylim([-l/2-.5 l/2+.5])
xlim([-1.25 1.25])
p(2).LineWidth = 2;
p(3).LineWidth = 2;
```

Lab Report 3

Ryan Nand
ECE332L



Here is the current distribution when lambda = 1 and l = lambda*1.25. I produced the code in a way where it will plot the dipoles as well. Everything will update properly with the changing of lambda or the length. I checked the plots from the book (fig 9-16) and they all matched with the proper length equation. Therefore, I am pretty confident with this current distribution for this length requirement.

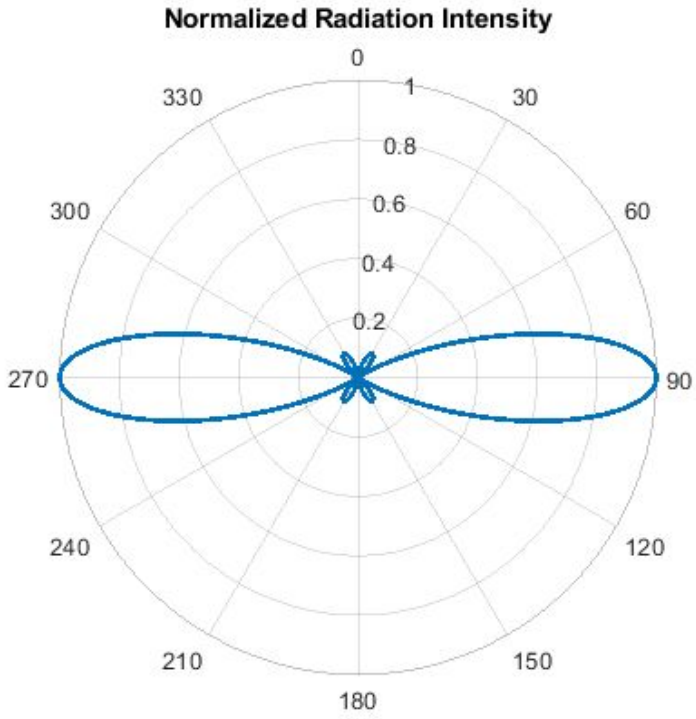
Ryan Nand
ECE332L

2. Normalized Radiation Intensity

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Ryan Nand
% 5/23/2020
% Dipole Antenna Normalized Radiation Intensity
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
lambda = 1;                % Chosen value for Lambda
l = lambda*1.25;           % Length in terms of Lambda
n = 2000;                  % Number of points evaluated
z = linspace(0, 2*pi, n);  % Equal distribuion of points between 0 and 2*pi
s = zeros(1, n);           % Preallocate variable with zeros
s_max = 0;                 % Preallocate variable with zero
for i = 1:n    % Loop to update s_max and calculate s(theta) for each point
    % Equation for s(theta) with arbitrary length
    s(i) = ((cos(pi*l/lambda*cos(z(i)))-cos(pi*l/lambda))/sin(z(i)))^2;
    if s_max < s(i)        % Determines the largest value of s(theta)
        s_max = s(i);
    end
end
ax = polaraxes;                % To configure plot details
% Intensity is s/s_max from the equation/results above
p = polarplot(s/s_max);        % Plot results of intensity
% Configure polar plot details
ax.ThetaDir = 'clockwise';
ax.ThetaZeroLocation = 'top';
p(1).LineWidth = 2;
title('Normalized Radiation Intensity')
```

Lab Report 3

Ryan Nand
ECE332L



I produced the code where each plot matched the proper length from the table 9-17 in the book. Therefore, I am pretty confident with this plot. The intensity shows small lobes towards the center where the largest intensity value is at 90 degrees. Also, the large lobes are similar to the case where l = lambda.

Ryan Nand
ECE332L

3. Pattern Solid Angle and Directivity

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Ryan Nand
% 5/23/2020
% Dipole Antenna Pattern Solid Angle and Directivity
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
syms theta phi                  % Create symbolic variables
lambda = 1;                     % Chosen Lambda value
l = lambda*1.25;                % Length in terms of Lambda
n = 1000;                       % Number of plot points
z = linspace(0, 2*pi, n);       % Create equal spaced points
s = zeros(1, n);                % Preallocate array variable
s_max = 0;                      % Preallocate variable
for i = 1:n                     % Loop to update needed s_max
    s(i) = ((cos(pi*l/lambda*cos(z(i)))-cos(pi*l/lambda))/sin(z(i)))^2;
    if s_max < s(i)             % Update if >= current s_max
        s_max = s(i);
    end
end
% Equation needed for s(theta)
s_theta = ((cos(pi*l/lambda*cos(theta))-cos(pi*l/lambda))/sin(theta))^2;
% Equation needed for finding angle of half-power of intensity
eqn = s_theta/s_max == .5;
A = vpasolve(eqn, theta, [0 pi]);       % Solve for half-power angle
% Equation needed for pattern solid angle integral
f_theta = s_theta/s_max*sin(theta);
temp = int(f_theta, theta, [0 A*2]);    % First integral (dtheta)
temp = int(temp, phi, [0 2*pi]);        % Second integral (dphi)
Solid_Angle_Pattern = vpa(temp);        % Approx. symbolic integral
D = 4*pi/Solid_Angle_Pattern;           % Equation for Directivity
% Print values calculated
fprintf('Pattern Solid Angle: %.3f\n', Solid_Angle_Pattern)
fprintf('Directivity: %.3f\n', D)
```

```
Pattern Solid Angle: 3.764
Directivity: 3.339
```

Here I used the proper equation for the solid angle, instead of the approximation. This was done using symbolic functions and symbolic integration. There was a tricky part where I had to configure vpasolve to only solve within a certain window for the correct result. The result is shown in the code (0 to pi). This will find the value of the half-power using the lobe shown in the plot above, where the 90 degree lobe is.

These two values give an identity to how narrow the radiation pattern is. Comparing it with the case where l = lambda. This solid angle is less and the directivity is greater. This may be because the lobes for the case where l = lambda is wider than this case. Even though there are small lobes near the center. I can find this angle using the code above.

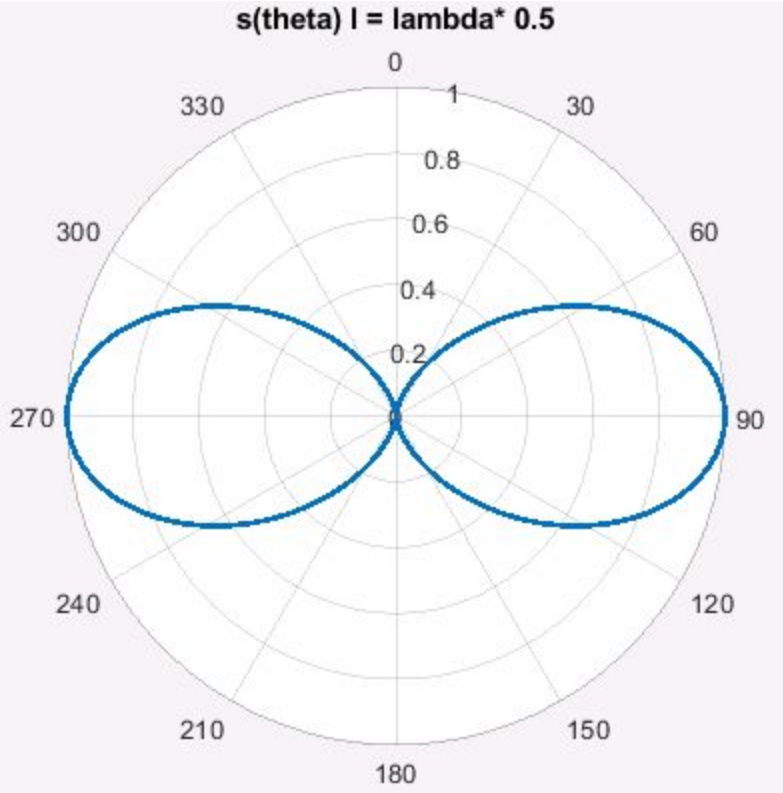
4. s(theta) for varying length

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Ryan Nand
% 5/23/2020
% s(theta) for varying length
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
lambda = 1;                 % Chosen Lambda value
n = 2000;                   % Number of points for plot
z = linspace(0, 2*pi, n);   % Equally spaced pts between 0 and 2*pi
s = zeros(1, n);            % Preallocate array
s_max = 0;                  % Preallocate variable
for j = .1:.1:2.1           % 20 steps between .1 and 2.1
l = j*lambda;               % Step length accordingly
for i = 1:n                 % For each step calculate s(theta)
    s(i) = ((cos(pi*l/lambda*cos(z(i)))-cos(pi*l/lambda))/sin(z(i)))^2;
end
ax = polaraxes;             % Variable used to configure plot
p = polarplot(s);           % Plot s(theta)
% Configure plot details
ax.ThetaDir = 'clockwise';
ax.ThetaZeroLocation = 'top';
p(1).LineWidth = 2;
title(['s(theta) l = lambda* ',num2str(l)])
pause                       % Pause command to step through plots
close all
end
```

L = lambda/2

Lab Report 3

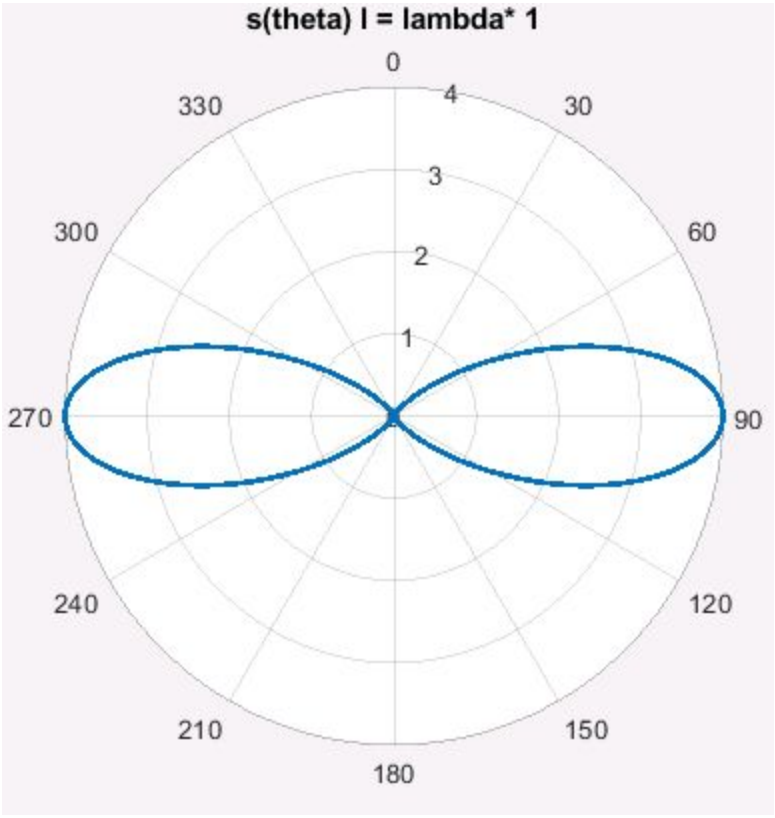
Ryan Nand
ECE332L



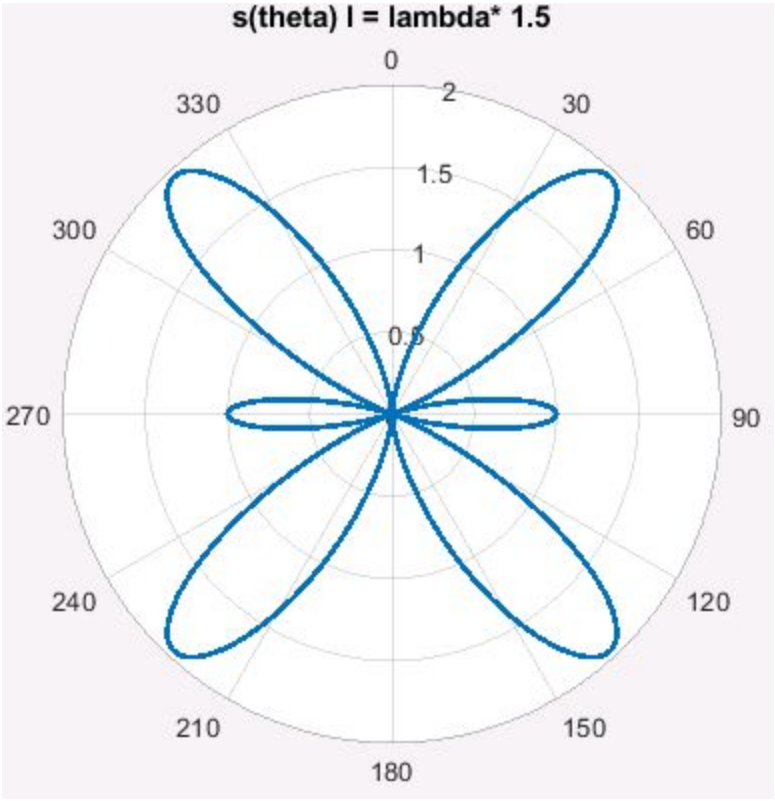
L = lambda

Lab Report 3

Ryan Nand
ECE332L



L = lambda*(3/2)



Ryan Nand
ECE332L

The plots show a lot of variation when length is varied. This shows the s(theta) not the normalized radiation intensity. So that we can see the different radiation patterns and the power levels in each case. Looks like among the three cases shown, when l = lambda has the greatest power level. L = lambda*.5 has the smallest power. While L = lambda*1.5 has the craziest radiation pattern.

5. Extra Credit

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Ryan Nand
% 5/23/2020
% Movie for s(theta) with varying length
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all
index = 81;                 % The indexing for the movie
k = .1;                     % Initial value for length
% Create variable to store frames
m(index) = struct('cdata',[],'colormap',[]);
lambda = 1;                 % Chosen value for Lambda
n = 2000;                   % Number of points for plot
z = linspace(0, 2*pi, n);   % Equally spaced points for plot
s = zeros(1, n);            % Preallocate array
s_max = 0;                  % Preallocate variable
for j = 1:index             % Loop to store each frame
l = k*lambda;               % Update length in terms of Lambda
for i = 1:n                 % Loop to calculate s(theta) and s_max
    s(i) = ((cos(pi*l/lambda*cos(z(i)))-cos(pi*l/lambda))/sin(z(i)))^2;
    if s_max < s(i)         % Update s_max for largest value
        s_max = s(i);
    end
end
ax = polaraxes;             % Create to configure plots
p = polarplot(s);           % Plot s(theta)
% Configure plot details
ax.ThetaDir = 'clockwise';
ax.ThetaZeroLocation = 'top';
p(1).LineWidth = 2;
title('Normalized Radiation Intensity')
m(index) = getframe;        % Store completed plot
k = k + .025;               % Increment length
end
movie(m)                    % Play movie
```