```
clear all;
close all;
clc;

%% Field Definition
syms theta phi
ks = pi;
AF(theta, phi) = 2*(cos(pi*sin(theta)*sin(phi)) - cos(pi*sin(theta)*cos(phi)));
kl_2 = (2*pi) * 0.25;
Element_Factor(theta, phi) = (cos(kl_2 * cos(theta))) / sin(theta);
E(theta, phi) = Element_Factor(theta, phi) * AF(theta, phi);

%% Directivity
[Prad_value, D0] = compute_directivity(E);
disp(10*log10(D0));
disp(Prad_value);
disp(findUmax(matlabFunction(E)));

%% E Plane, phi=0, theta=[0, pi]
%E(theta, phi) = sin(theta) + 1e-100*sin(phi);

phi = 0;
theta = linspace(0.001, pi-0.001, 1000);

E_eval = abs(eval(E(theta, phi) .^ 2));
E_plot = 10.0 * log10(E_eval ./ max(E_eval));
E_plot(E_plot < -30) = -30;
%subplot(2, 1, 1);
polarplot(theta, E_plot);
rlim([-30, 0]);
title("Question 1, Part A) E-Plane, phi=0, theta=[0, pi]");

%% H Plane
figure
phi = linspace(0, 2*pi, 200);
theta = pi/2;

Hplane_eval = abs(eval(E(theta, phi)));
Hplane_plot = 20 .* log10(Hplane_eval / max(Hplane_eval));
Hplane_plot(Hplane_plot < -30) = -30;
%subplot(2, 1, 2);
polarplot(phi, Hplane_plot);
title("Question 1, Part A) H-Plane, phi=[0, 2pi] theta=pi/2");
rlim([-30, 0]);

function [theta_max, phi_max, U_max] = findUmax(U_func)
    if nargin(U_func) == 2
        % Define a nested function to be minimized
        neg_U_func = @(x) -U_func(x(1), x(2));
        % Initial guess for theta and phi
        x0 = [0, 0];
        % Perform the optimization
        [x_max, U_max] = fminsearch(neg_U_func, x0);
        % Extract theta_max and phi_max
        theta_max = x_max(1);
        phi_max = x_max(2);
        U_max = -U_max;
    elseif nargin(U_func) == 1
        % Define a nested function to be minimized
        neg_U_func = @(x) -U_func(x(1));
        % Initial guess for theta
        x0 = [0];
        % Perform the optimization
        [x_max, U_max] = fminsearch(neg_U_func, x0);
```

```
        % Extract theta_max and phi_max
        theta_max = x_max(1);
        U_max = -U_max;
    else
        error('Incorrect number of input arguments.');
    end
end

function [Prad_value, D0] = compute_directivity(field)
    if nargin(matlabFunction(field)) == 2
        syms theta phi
        U_inten = matlabFunction(field * conj(field));
        Prad = matlabFunction(U_inten(theta, phi) * sin(theta), 'Vars', {theta, phi});
        Prad_value = integral2(Prad, 0, pi, 0, 2*pi);
        [~, ~, Umax] = findUmax(U_inten);

        D0 = (4*pi*Umax)/Prad_value;
    elseif nargin(matlabFunction(field)) == 1
        syms theta
        U_inten = matlabFunction(field * conj(field));
        Prad = matlabFunction(U_inten(theta) * sin(theta), 'Vars', {theta});
        Prad_value = 2*pi*integral(Prad, 0, pi);
        [~, Umax] = findUmax(U_inten);

        D0 = (4*pi*Umax)/Prad_value;
    else
        error('Incorrect number of input arguments.');
    end
end

clear all;
close all;
clc;

%% E Plane
f = figure;
f.Position = [0, 0, 1280, 720];
spacing = [0.66, 0.8, 1.2, 1.5];
spacing1 = [0.66, 1.2, 0.8, 1.5];
centerfig(f);

for i=1:1:length(spacing)
    %% Field Definition
    syms theta phi

    v = spacing(i);
    ks = pi * v;

    AF(theta, phi) = 2*(cos(ks*sin(theta)*sin(phi)) - cos(ks*sin(theta)*cos(phi)));
    E(theta, phi) = sin(theta) * AF(theta, phi);

    phi = pi/2;
    theta = linspace(0, pi, 1000);

    E_eval = abs(eval(E(theta, phi) .^ 2));
    E_plot = 10.0 * log10(E_eval ./ max(E_eval));
    E_plot(E_plot < -15) = -15;
    polarplot(theta, E_plot, 'DisplayName', sprintf("%.2ff_{0}", spacing(i)));
    hold on;
end
title("Varying Spacing Factor, s, Effect on Beamwidth; E-Plane(s), phi=0, theta=[0, pi]");
rlim([-15, 0]);
legend();

half_pwr = zeros([1, length(theta)]) - 3;
```

```
polarplot(theta, half_pwr, "DisplayName", "Half Power");

clear all
close all
clc
% https://www.desmos.com/calculator/7dmxksal1e
%% Design Parameters
lambda = 299792458/800e6;
C = lambda;
alpha = 13.5;
S = C * tan(deg2rad(alpha));
N = 5;
D_0 = 10*log10(15*N*(C^2*S)/(lambda^3));
HPBW = ((52*lambda^(3/2))/(C*sqrt(N*S)));

f = figure;
f.Position = [0,0,1280,720];
centerfig(f);

%% E-Plane Plot
syms theta
k_0 = (2*pi)/lambda;
L_0 = sqrt(S^2 + C^2);
p = (L_0/lambda)/(S/lambda + 1);
psi = k_0 * (S * cos(theta) - L_0/p);

E(theta) = sin(pi/(2*N))*cos(theta)*( (sin((N/2)*psi)) / (sin(psi/2))  );
theta = linspace(-pi, pi, 1000);
E_eval = eval(E) .^ 2;
E_eval = 10.0 .*log10(E_eval ./ max(E_eval));
E_eval(E_eval < -30) = -30;
polarplot(theta, E_eval);
rlim([-30, 0]);
title("E Plane, phi=pi/2")

clear all
close all
clc
% https://www.desmos.com/calculator/7dmxksal1e
%% Design Parameters
lambda = 299792458/800e6;
C = lambda;
alpha = 13.5;
S = C * tan(deg2rad(alpha));
N = 5;
D_0 = 10*log10(15*N*(C^2*S)/(lambda^3));
HPBW = ((52*lambda^(3/2))/(C*sqrt(N*S)));

%% E-Plane Plot
 syms theta phi

k_0 = (2*pi)/lambda;
L_0 = sqrt(S^2 + C^2);
p = (L_0/lambda)/(S/lambda + 1);
psi = k_0 * (S * cos(theta) - L_0/p);

des = linspace(0, 2, 200);
Directivity = zeros([1, length(des)]);
DirectivityLobes = zeros([1, length(des)]);

%% Plot E-Plane
f = figure;
f.Position = [0, 0, 1280, 720];
centerfig(f);
d = 0.91;
```

```
syms theta phi
AF(theta, phi) = 2*cos(pi*d*sqrt(2) * sin(theta)*cos(phi)) + 2*cos(pi*d*sqrt(2)*sin(theta)*sin(phi));
ElementFactor(theta) = sin(pi/(2*N))*cos(theta)*( (sin((N/2)*psi)) / (sin(psi/2))  );
E(theta) = ElementFactor(theta) * AF(theta, phi);
% Plot E-Plane of Element Factor
theta = linspace(-pi, pi, 500);
E_eval = eval(ElementFactor);
E_eval = real(E_eval .* conj(E_eval));
E_eval = E_eval ./ max(E_eval);
E_eval = 10.0 .* log10(E_eval);
E_eval(E_eval < -40) = -40;
polarplot(theta, E_eval, "DisplayName", "Single Element");
rlim([-40, 0]);
hold on
phi = 0;
E_eval = eval(E);
E_eval = real(E_eval .* conj(E_eval));
E_eval = E_eval ./ max(E_eval);
E_eval = 10.0 .* log10(E_eval);
E_eval(E_eval < -40) = -40;
polarplot(theta, E_eval, "DisplayName", "Array Factor * Element");
rlim([-40, 0]);
title("Radiation Pattern of Single Element vs Array System");
legend();

%% Create Design Curve for Directivity
for i=1:1:length(des)
    d = des(i);
    kd = (2*pi)*(d/2);
    fprintf("%d: d=%.2f ", i, d);
    syms theta phi


    AF(theta, phi) = 2*cos(pi*d*sqrt(2) * sin(theta)*cos(phi)) +
2*cos(pi*d*sqrt(2)*sin(theta)*sin(phi));

    phi = 0;

    E(theta) = sin(pi/(2*N))*cos(theta)*( (sin((N/2)*psi)) / (sin(psi/2))  ) * AF(theta, phi);

    U_inten = matlabFunction( E ^ 2 );
    Prad = matlabFunction(abs(U_inten * sin(theta)));
    Prad_value = 2*pi*integral(Prad, 0, pi);
    [theta_max, Umax] = find_max_U(U_inten);

    D0 = 10.*log10((4*pi*Umax)/Prad_value);
    Directivity(i) = D0;
    fprintf("D0=%.2f ", D0);
    theta = linspace(-pi, pi, 500);
    E_eval = eval(E) .^ 2;
    E_eval = 10.0 .*log10(E_eval ./ max(E_eval));
    DirectivityLobes(i) = findLobelLevel(E_eval);
    fprintf("D(sideLobe)=%.2f\n", DirectivityLobes(i));
    %E_eval(E_eval < -30) = -30;
    %polarplot(theta, E_eval, "DisplayName", sprintf("d = %.2f, D0 = %.2fdB", d, D0));
    %hold on
end
 f = figure;
 f.Position = [0, 0, 1280, 720];
 centerfig(f);
yyaxis left
plot(des, Directivity, "DisplayName", "Max Directivity (dB)");
xlabel("d (in lambda)");
ylabel("D_{0} (dB)");
title("Directivity with respect to array spacing factor d");
```

```
hold on
yyaxis right
plot(des, DirectivityLobes, "DisplayName", "Sidelobe Directivity (dB)");
grid on
hold off
%rlim([-30, 0]);
%title("Directivity Design Plot");
legend("FontSize", 14)
yline(-40, '--', 'HandleVisibility','off');
xline(0.91, '--', 'HandleVisibility','off');
%yline(-20, "label", "Max Allowable Sidelobe level", "FontSize", 14)
function [theta_max, U_max] = find_max_U(U_func)
    % Define a nested function to be minimized
    neg_U_func = @(x) -U_func(x(1));

    % Initial guess for theta and phi
    x0 = [0];

    % Perform the optimization
    [x_max, U_max] = fminsearch(neg_U_func, x0);

    % Extract theta_max and phi_max
    theta_max = x_max(1);
    U_max = -U_max;

end

function lobeLevel = findLobelLevel(E_eval)
        % Find the regional maximums
    maxima_indices = find(islocalmax(E_eval));
    % Extract the regional maximums
    regional_maximums = E_eval(maxima_indices);
    % Sort the regional maximums in descending order
    sorted_regional_maximums = sort(regional_maximums, 'descend');
    % Select the second highest value
    second_highest = sorted_regional_maximums(2);
    lobeLevel = second_highest;
end

clear all
close all
clc

syms theta

f = figure;
f.Position = [0,0,1280,720];
centerfig(f);

 Element_Factor(theta) = cos(theta);
%Element_Factor(theta, phi) = sqrt(cos(phi)^2*sin(theta)^2+cos(theta)^2);

Array_Factor(theta) = 0.9*exp(j*0.4*pi*(-cos(theta) + 1)) + ...
    1 + ...
    0.7*exp(j*0.6*pi*(cos(theta) - 1)) + ...
    0.5*exp(j*pi*(cos(theta) - 1));

Electric_Trans(theta) = Element_Factor(theta) * Array_Factor(theta);

% Find the front-to-back ratio in the radiation pattern.
U_inten = matlabFunction( abs(Electric_Trans ^ 2) );
Prad = matlabFunction(abs(U_inten * sin(theta)));
Prad_value = 2*pi*integral(Prad, 0, pi);
[theta_max,  Umax] = find_max_U(U_inten);
U0 = (4*pi)/Prad_value;
```

```
Gain_0degree = abs(U_inten(0))/U0;
Gain_180degree = abs(U_inten(pi))/U0;
fprintf("Front to back ratio is %f\n", Gain_0degree/Gain_180degree);

D0 = (4*pi*Umax)/Prad_value;
fprintf("D0 = %.2f --- D0(dB) = %.2f\n", D0, 10*log10(D0));
theta = linspace(-pi, pi, 1000);

E_eval = abs(eval(Electric_Trans));
E_eval = E_eval ./ max(E_eval);
E_eval = 20.0 .* log10(E_eval);
E_eval(E_eval < -20) = -20;
polarplot(theta, E_eval);
rlim([-20, 0]);
title("E Plane Radiation Pattern " + sprintf("D_{0} = %.2f --- D_{0}(dB) = %.2f\n", D0,
10*log10(D0)));

function [theta_max, U_max] = find_max_U(U_func)
    % Define a nested function to be minimized
    neg_U_func = @(x) -U_func(x(1));

    % Initial guess for theta
    x0 = [0];

    % Perform the optimization
    [x_max, U_max] = fminsearch(neg_U_func, x0);

    % Extract theta_max and phi_max
    theta_max = x_max(1);
    U_max = -U_max;
end
```