Design Optimization of a Six-element Yagi-Uda Antenna

Jegan Mani VU3JYU, Vishwanath Iyer KB1SPX

This example optimizes a 6-element Yagi-Uda antenna for both directivity and 50Ω input match using a global optimization technique. The radiation patterns and input impedance of antennas are sensitive to the parameters that define their shapes. The multidimensional surface over which such optimizations must be performed have multiple local optima. This makes the task of finding the right set of parameters satisfying the optimization goals particularly challenging and requires the use of global optimization techniques. In this paper the 2 such techniques are considered pattern search, a direct search based optimization technique that has yielded impressive results for antenna design optimization and surrogate model optimization, where the optimizer internally builds an inmemory model of the problem space and uses that for optimization.

The Yagi-Uda antenna is a widely used radiating structure for a variety of applications in commercial and military sectors. This antenna has been popular for reception of TV signals in the VHF-UHF range of frequencies [1]. The Yagi is a directional traveling-wave antenna with a single driven element, usually a folded dipole or a standard dipole, which is surrounded by several passive elements. The passive elements form the *reflector* and *director.* These names identify the positions relative to the driven element. The reflector dipole is behind the driven element in the direction of the back lobe of the antenna radiation, while the director is in front of the driven element, in the direction where a main beam forms.

Design Parameters

Choose initial design parameters in the center of the VHF band . The design lists a 50Ω input impedance after taking into account a balun. Our model does not account for the presence of the balun but will still match an input impedance of 50Ω , balun will be used in the final installation prevent RFI and choke currents on the coaxial liine.

```
fc = 144.5e6;
wirediameter = 12e-3;
c = physconst('lightspeed');
lambda = c/fc;
Z0 = 50;
BW = 0.015*fc;
fmin = fc - 2*(BW);
fmax = fc + 2*(BW);
Nf = 101;
freq = linspace(fmin,fmax,Nf);
```

Create Yagi-Uda Antenna

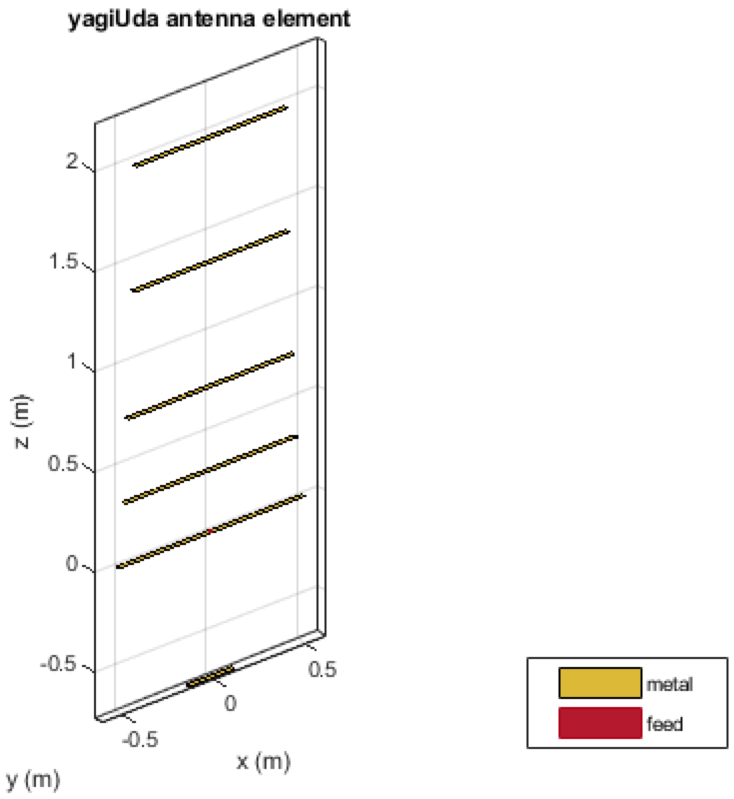
The driven element for the Yagi-Uda antenna is a dipole. This is a standard exciter for such an antenna. Adjust the length and width parameters of the dipole. Since we model cylindrical structures as equivalent metal strips, the width is calculated using a utility function available in the Antenna Toolbox™. The length is chosen to be $\lambda/2$ at the design frequency.

```
d = dipole;
d.Length = 0.982.*(lambda/2);
d.Width = cylinder2strip(wirediameter/2);
```

```
d.Tilt = 90;
d.TiltAxis = 'Y';
```

Create a Yagi-Uda antenna with the exciter as the dipole. Choose the reflector to be greater than $\lambda/2$ and director length to be less $\lambda/2$. Set the number of directors to four. Choose the reflector and director spacing to be 0.3λ, 0.25λ respectively. These choices are an initial guess and will serve as a start point for the optimization procedure. Show the initial design.

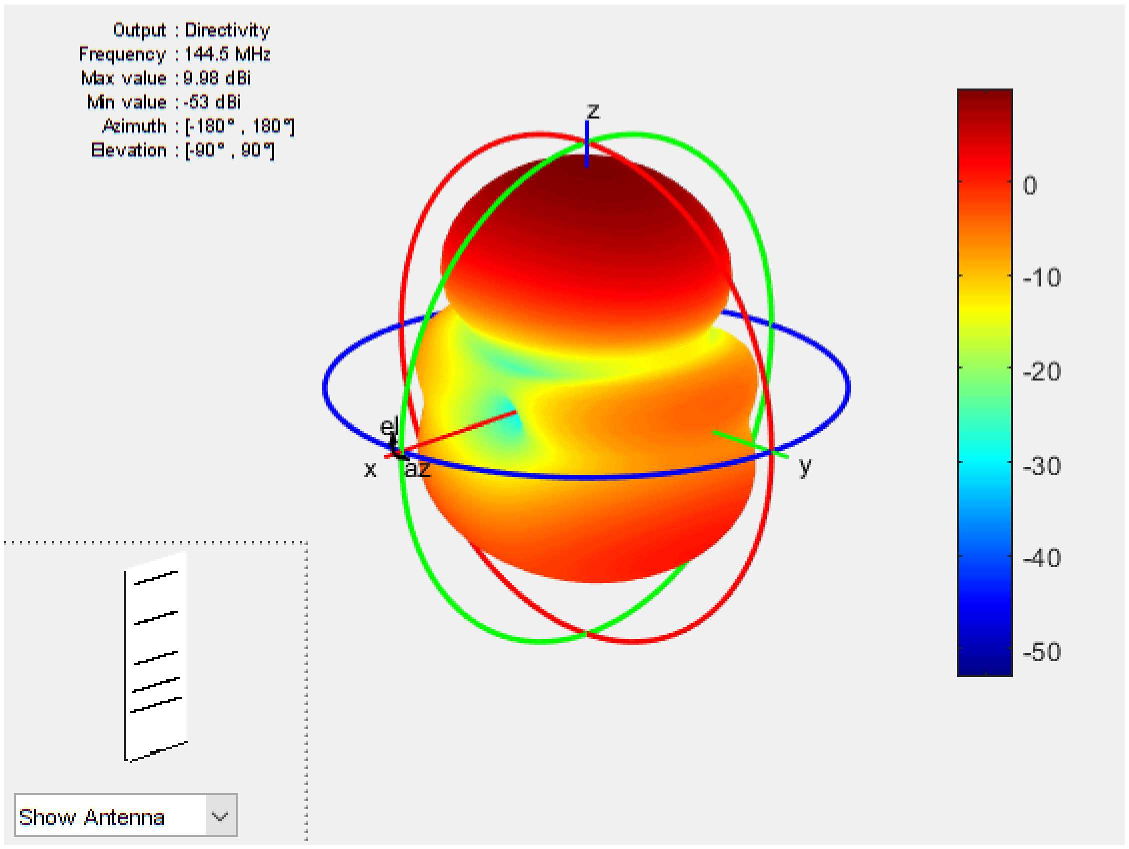
```
Numdirs = 4;
refLength = 0.25; %1.05;
dirLength = [0.940 0.910 0.850 0.830]; %0.5*ones(1,Numdirs);
refSpacing = 0.35;
dirSpacing = [0.15 0.2 0.3 0.3 ];

initialdesign = [refLength refSpacing dirSpacing].*lambda;
yagidesign = yagiUda;
yagidesign.Exciter = d;
yagidesign.NumDirectors = Numdirs;
yagidesign.ReflectorLength = refLength; %.*(lambda/2);
yagidesign.DirectorLength = dirLength; %.*(lambda/2);
yagidesign.ReflectorSpacing = refSpacing*lambda;
yagidesign.DirectorSpacing = dirSpacing*lambda;
show(yagidesign)
```



Plot Radiation Pattern at Design Frequency

Prior to executing the optimization process, plot the radiation pattern for the initial guess in 3D.

```
fig1 = figure;
pattern(yagidesign,fc);
```



This initial Yagi-Uda antenna does not have a higher directivity in the preferred direction, meaning at zenith (elevation = 90 deg) and is therefore a poorly designed radiator.

Set Up Optimization

Use the following variables as control variables for the optimization:

- Reflector length (1 variable)
- Director lengths (4 variables)
- Reflector spacing (1 variable)
- Director spacings (4 variables)
- Exciter length (1 variable)
- Exciter spacing (1 variable)

In terms of a single vector parameter `controlVals`, set

- Reflector length = `controlVals(1)`
- Director lengths = `controlVals(2:5)`
- Reflector spacing = `controlVals(6)`
- Director spacings = `controlVals(7:10)`
- Exciter length = `controlVals(11)`
- Exciter spacing = `controlVals(12)`

In terms of `controlVals`, set an objective function that aims to have a large directivity value in the 90 degree direction, a small value in the -90 degree direction, and a large value of maximum power between the elevation beamwidth angle bounds. In addition to the directivity goal an impedance match condition is also included as a constraint. Any constraint violations will penalize the objective.

```
type yagi_objective_function.m
```

```
function objectivevalue = yagi_objective_function(y,controlVals,fc,BW,ang,Z0,constraints)
% YAGI_OBJECTIVE_FUNCTION returns the objective for a 6 element Yagi
% OBJECTIVE_VALUE =
% YAGI_OBJECTIVE_FUNCTION(Y,CONTROLVALS,FREQ,ANG,Z0,constraints), assigns
% the appropriate parasitic dimensions, CONTROLVALS to the Yagi antenna Y,
% and uses the frequency FREQ, angle pair,ANG, reference impedance Z0 and
% the constraints to calculate the objective function value.

% The YAGI_OBJECTIVE_FUNCTION function is used for an internal example.
% Its behavior may change in subsequent releases, so it should not be
% relied upon for programming purposes.

% Copyright 2018 The MathWorks, Inc.

% y.ReflectorLength = controlVals(1);
% y.DirectorLength = controlVals(2:y.NumDirectors+1);
% y.ReflectorSpacing = controlVals(y.NumDirectors+2);
% y.DirectorSpacing = controlVals(y.NumDirectors+3:end-2);
% y.Exciter.Length = controlVals(end-1);
% y.Exciter.Spacing = controlVals(end);

y.ReflectorLength = controlVals(1);
y.ReflectorSpacing = controlVals(2);
y.DirectorSpacing = controlVals(3:6);

% Unpack constraints
Gmin = constraints.Gmin;
Gdev = constraints.Gdeviation;
FBmin = constraints.FBmin;
K = constraints.Penalty;

% Calculate antenna port and field parameters
output = analyzeAntenna(y,fc,BW,ang,Z0);

% Form objective function
Gain = output.MaxDirectivity+output.MismatchLoss;    % Directivity/Gain at zenith
Gain1 = output.MaxDirectivity1+output.MismatchLoss1;    % Directivity/Gain at zenith
Gain2 = output.MaxDirectivity2+output.MismatchLoss2;    % Directivity/Gain at zenith

% Gain constraint, e.g. G > 10
c1 = 0;
c1 = (abs((Gain-Gmin))+abs((Gain1-Gmin))+abs((Gain2-Gmin)));

% Gain deviation constraint, abs(G-Gmin)<0.1;
```

```
c1_dev = 0;
c1_dev_temp = 0;
c1_dev_temp1 = 0;
c1_dev_temp2 = 0;
if abs(Gain-Gmin)>Gdev
    c1_dev_temp = -Gdev + abs(Gain-Gmin);
end
if abs(Gain-Gmin)>Gdev
    c1_dev_temp1 = -Gdev + abs(Gain1-Gmin);
end
if abs(Gain-Gmin)>Gdev
    c1_dev_temp2 = -Gdev + abs(Gain2-Gmin);
end
c1_dev = (c1_dev_temp+c1_dev_temp1+c1_dev_temp2)/3;

% Front to Back Ratio constraint, e.g. F/B > 15
c2 = 0;
% if output.FB < FBmin
%     c2 = FBmin-output.FB;
% end
c2 = (abs(FBmin-output.FB)+abs(FBmin-output.FB1)+abs(FBmin-output.FB2))/3;

% impedance constraint
c3 = (abs((output.Z-50))+abs((output.Z1-50))+abs((output.Z2-50)));

% Form the objective + constraints
objectivevalue = -((Gain+Gain1+Gain2)/3) + max(0,(c1+c1_dev+c2+c3))*K;
end

function output = analyzeAntenna(ant,fc,BW,ang,Z0)
%ANALYZEANTENNA calculate the objective function
% OUTPUT = ANALYZEANTENNA(Y,FREQ,BW,ANG,Z0) performs analysis on the
% antenna ANT at the frequency, FC, and calculates the directivity at the
% angles specified by ANG and and the front-to-back ratio. The reflection
% coefficient relative to reference impedance Z0, and impedance are
% computed over the bandwidth BW around FC.

fmin = fc - (BW/2);
fmax = fc + (BW/2);
Nf = 10;
freq = unique([fc,linspace(fmin,fmax,Nf)]);
fcIdx = freq==fc;
fcIdx1 = freq==fmin;
fcIdx2 = freq==fmax;
s = sparameters(ant,freq,Z0);
Z = impedance(ant,fc);
Z1 = impedance(ant,fmin);
Z2 = impedance(ant,fmax);
az = ang(1,:);
el = ang(2,:);
Dmax = pattern(ant,fc,az(1),el(1));
Dmax1 = pattern(ant,fmin,az(1),el(1));
Dmax2 = pattern(ant,fmax,az(1),el(1));
Dback = pattern(ant,fc,az(2),el(2));
Dback1 = pattern(ant,fmin,az(2),el(2));
Dback2 = pattern(ant,fmax,az(2),el(2));

% Calculate F/B
F_by_B = (Dmax-Dback);
F_by_B1 = (Dmax1-Dback1);
F_by_B2 = (Dmax2-Dback2);

% Compute S11 and mismatch loss
s11 = rfparam(s,1,1);
S11 = mean((20*log10(abs(s11))));
```

```
T = max(10*log10(1 - (abs(s11(fcIdx))).^2));
T1 = max(10*log10(1 - (abs(s11(fcIdx1))).^2));
T2 = max(10*log10(1 - (abs(s11(fcIdx2))).^2));

% Form the output structure
output.MaxDirectivity= Dmax;
output.BackLobeLevel = Dback;
output.MaxDirectivity1= Dmax1;
output.BackLobeLevel1 = Dback1;
output.MaxDirectivity2= Dmax2;
output.BackLobeLevel2 = Dback2;
output.FB = F_by_B;
output.FB1 = F_by_B1;
output.FB2 = F_by_B2;
output.S11 = S11;
output.MismatchLoss = T;
output.MismatchLoss1 = T1;
output.MismatchLoss2 = T2;
output.Z = Z;
output.Z1 = Z1;
output.Z2 = Z2;
end
```

Set bounds on the control variables.

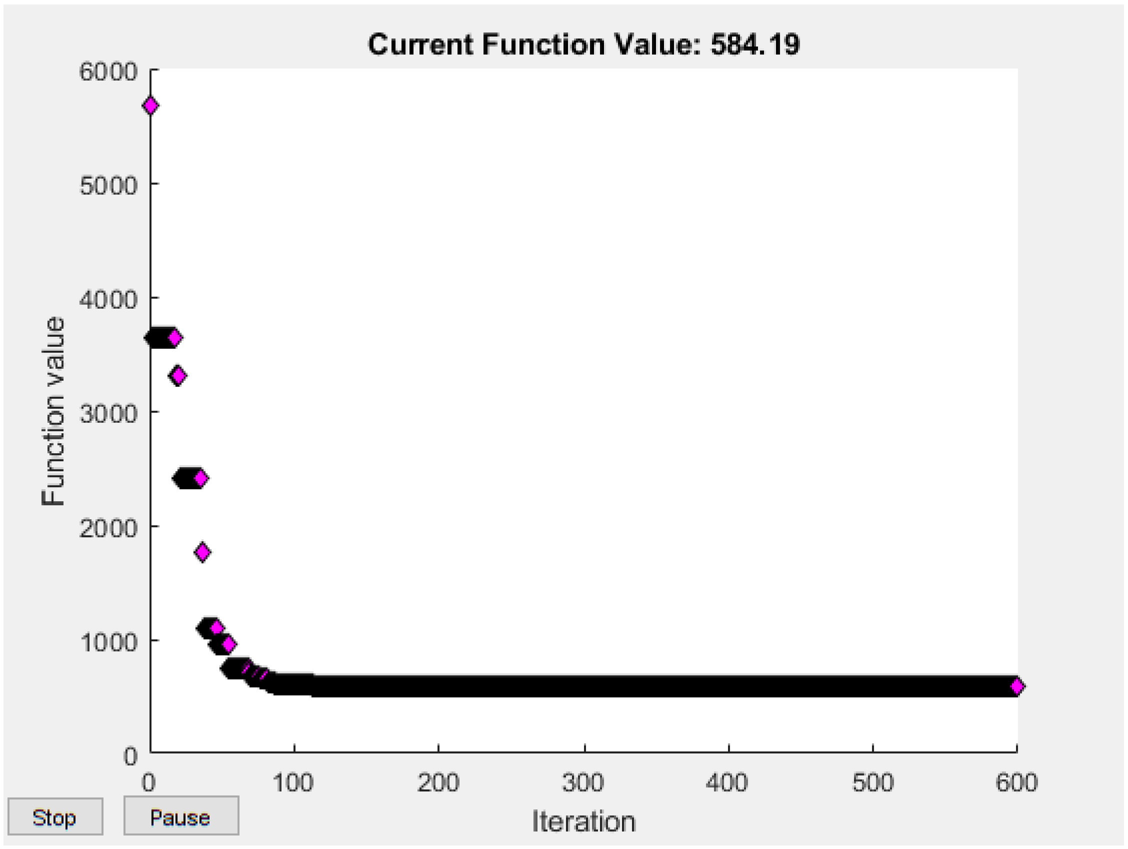
```
refLengthBounds = [0.1;                            % lower bound on reflector length
                   0.6];                           % upper bound on reflector spacing
dirLengthBounds = [0.3 0.3 0.3 0.3;                 % lower bound on director length
                   0.7 0.7 0.7 0.7];               % upper bound on director length
refSpacingBounds = [0.25;                          % lower bound on reflector spacing
                    0.65];                         % upper bound on reflector spacing
dirSpacingBounds = [0.01 0.1 0.1 0.1;           % lower bound on director spacing 0.2 0.25 0.3 0.3
                    0.2 0.25 0.35 0.35];             % upper bound on director spacing
exciterLengthBounds = [0.45;                       % lower bound on exciter length
                       0.6];                       % upper bound on exciter length
exciterSpacingBounds = [.004;
                        .008];


LB = [refLengthBounds(1) refSpacingBounds(1) dirSpacingBounds(1,:) ].*lambda;
UB = [refLengthBounds(2) refSpacingBounds(2) dirSpacingBounds(2,:) ].*lambda;
parameterBounds.LB = LB;
parameterBounds.UB = UB;
ang = [0 0;90 -90];                      % azimuth,elevation angles for main lobe and back lobe [a
```

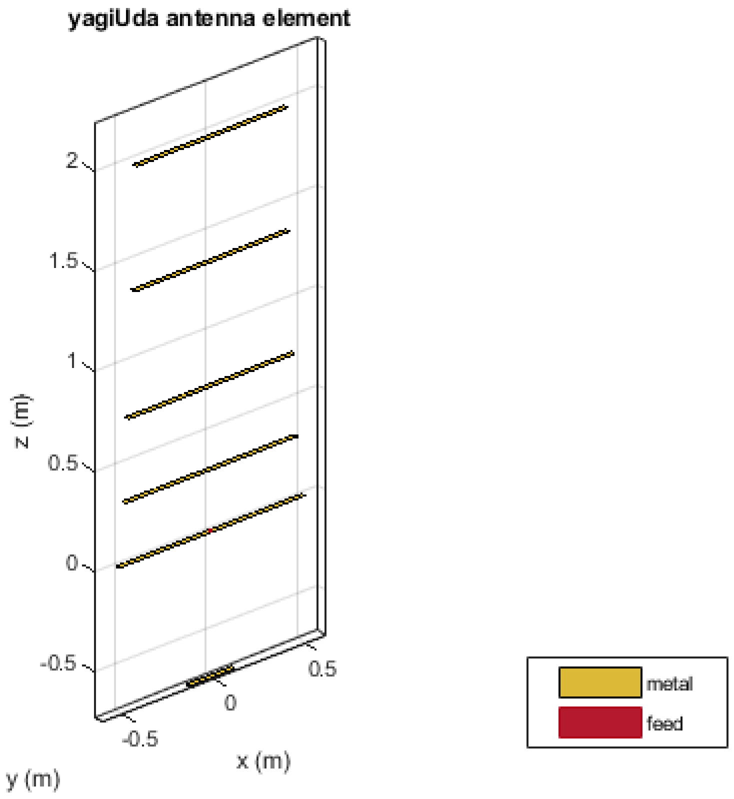
Direct Search Based Optimization

The Global Optimization Toolbox™ provides a direct search based optimization function called `patternsearch`. We use this function with options specified with the `psoptimset` function. At every iteration, plot the best value of the objective function and limit the total number of iterations to 300. Pass the objective function to the patternsearch function by using an anonymous function together with the bounds and the options structure.The objective function used during the optimization process by `patternsearch` is available in the file `yagi_objective_function`.

The evaluation of the directivity in different directions corresponding to the angular region defined for maximum radiation as well as maximum sidelobe/ and the backlobe level is given in the function `calculate_objectives` available within `yagi_objective function`.

```
% Optimizer options
optimizer = 'Surrogate'; % PatternSearch

if strcmpi(optimizer,'PatternSearch')
    optimizerparams = optimoptions(@patternsearch);
    optimizerparams.UseCompletePoll = true;
    optimizerparams.PlotFcns = @psplotbestf;
    optimizerparams.UseParallel = true;
    optimizerparams.Cache = 'on';
%     optimizerparams.MaxIter = 300;      % each iter is 2N times # of variables
    optimizerparams.MaxFunctionEvaluations = 1200;
    optimizerparams.FunctionTolerance = 1e-2;
elseif strcmpi(optimizer,'Surrogate')
    optimizerparams = optimoptions(@surrogateopt);
    optimizerparams.UseParallel = true;
    optimizerparams.MaxFunctionEvaluations = 600; % 1-1 with iterations
    optimizerparams.MinSurrogatePoints = 12; % N+1 <2N
    optimizerparams.InitialPoints = initialdesign;
else
    error('Optimizer not supported');
end

% Antenna design parameters
designparams.Antenna = yagidesign;
designparams.Bounds = parameterBounds;

% Analysis parameters
analysisparams.CenterFrequency = fc;
analysisparams.Bandwidth = BW;
analysisparams.ReferenceImpedance = Z0;
analysisparams.MainLobeDirection = ang(:,1);
analysisparams.BackLobeDirection = ang(:,2);

% Set constraints
constraints.S11min = -15;
constraints.Gmin = 10;
constraints.Gdeviation = 0.1;
constraints.FBmin = 20;
constraints.Penalty = 75;
poolobj = gcp;
% addAttachedFiles(poolobj,{'optimizeAntenna.m','yagi_objective_function.m'})
optimdesign = optimizeAntenna(designparams,analysisparams,constraints,optimizerparams);
```



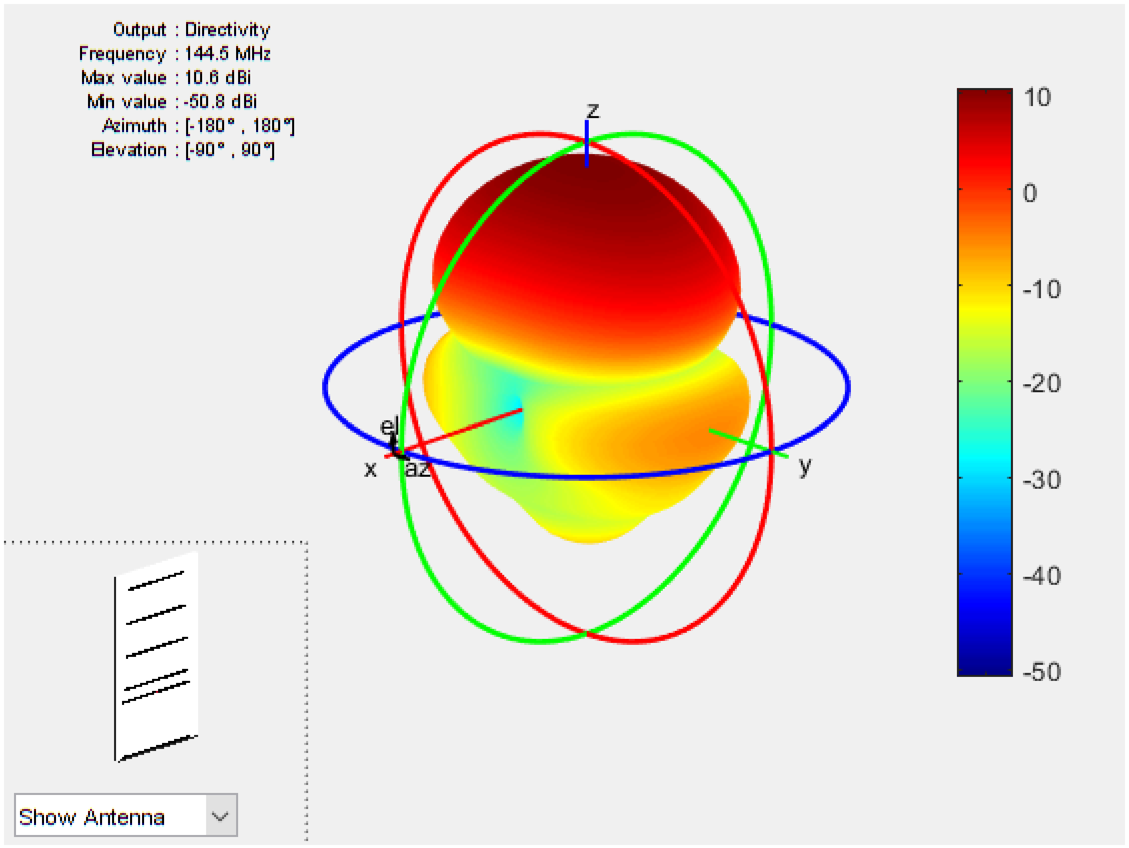


```
Surrogateopt stopped because it exceeded the function evaluation limit set by
'options.MaxFunctionEvaluations'.
```

Plot Optimized Pattern

Plot the optimized antenna pattern at the design frequency.

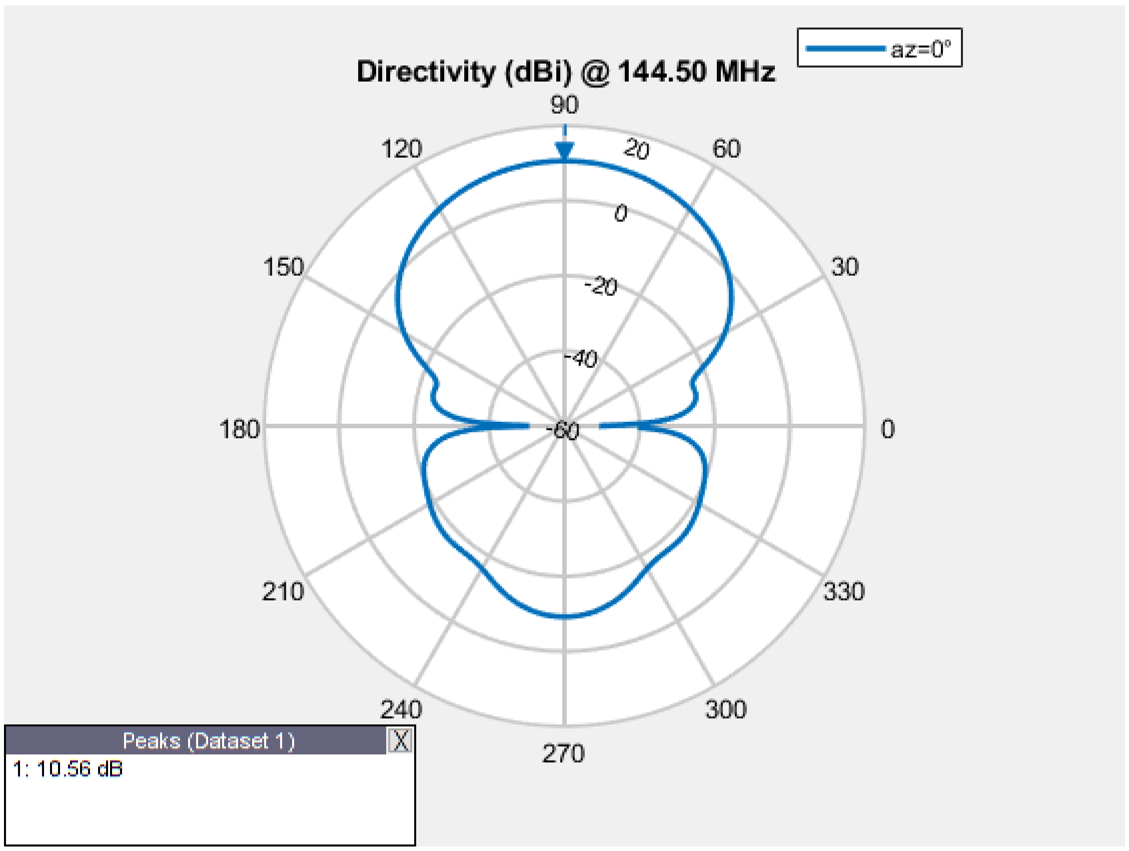
```
% yagidesign.ReflectorLength = optimdesign(1);
% yagidesign.DirectorLength = optimdesign(2:5);
% yagidesign.ReflectorSpacing = optimdesign(6);
% yagidesign.DirectorSpacing = optimdesign(7:10);
% yagidesign.Exciter.Length = optimdesign(11);
% yagidesign.Exciter.Spacing = optimdesign(12);
yagidesign.ReflectorLength = optimdesign(1);
yagidesign.ReflectorSpacing = optimdesign(2);
yagidesign.DirectorSpacing = optimdesign(3:6);
fig2 = figure;
pattern(yagidesign,fc)
```


```
Output : Directivity
Frequency : 144.5 MHz
Max value : 10.6 dBi
Min value : -50.8 dBi
Azimuth : [-180° , 180°]
Elevation : [-90° , 90°]
```

E and H-Plane Cuts of Pattern

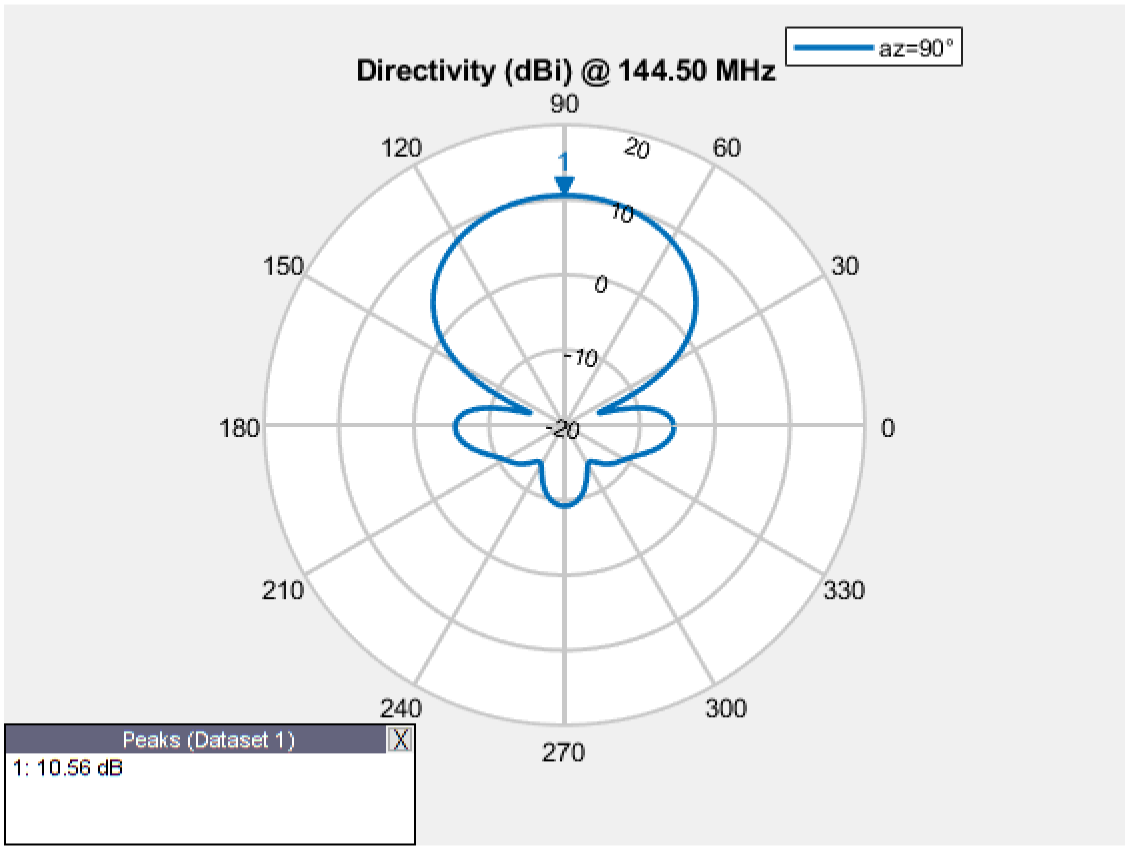
To obtain a better insight into the behavior in the two orthogonal planes, plot the normalized magnitude of the electric field in the E and H-planes, i.e. azimuth = 0 and 90 deg respectively. Enable the antenna metrics on the polar pattern plots to establish the directivity at zenith, Front-to-Back ratio, and the beamwidth in E and H-planes.

```
fig3 = figure;
patternElevation(yagidesign,fc,0,'Elevation',0:1:359);
```



```
pE = polarpattern('gco');
```

```
fig4 = figure;
patternElevation(yagidesign,fc,90,'Elevation',0:1:359);
```



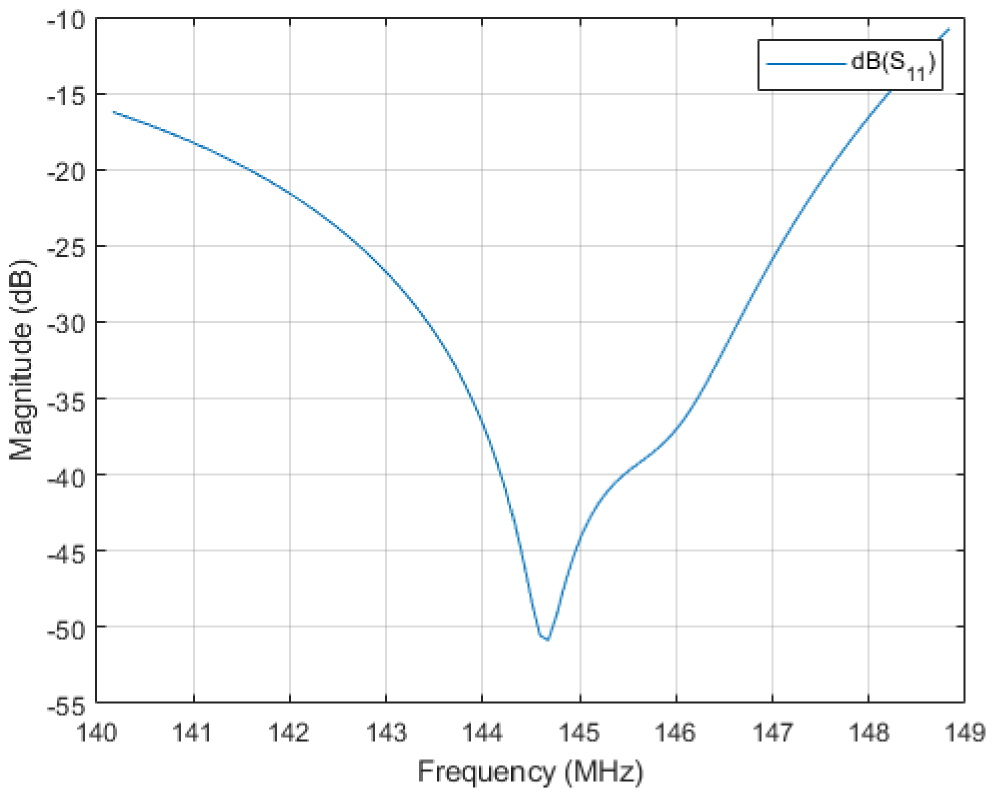
```
pH = polarpattern('gco');
```

The optimized design shows a significant improvement in the radiation pattern. There is higher directivity achieved in the desired direction toward zenith. The back lobe is small resulting in a good front to back ratio for this antenna.

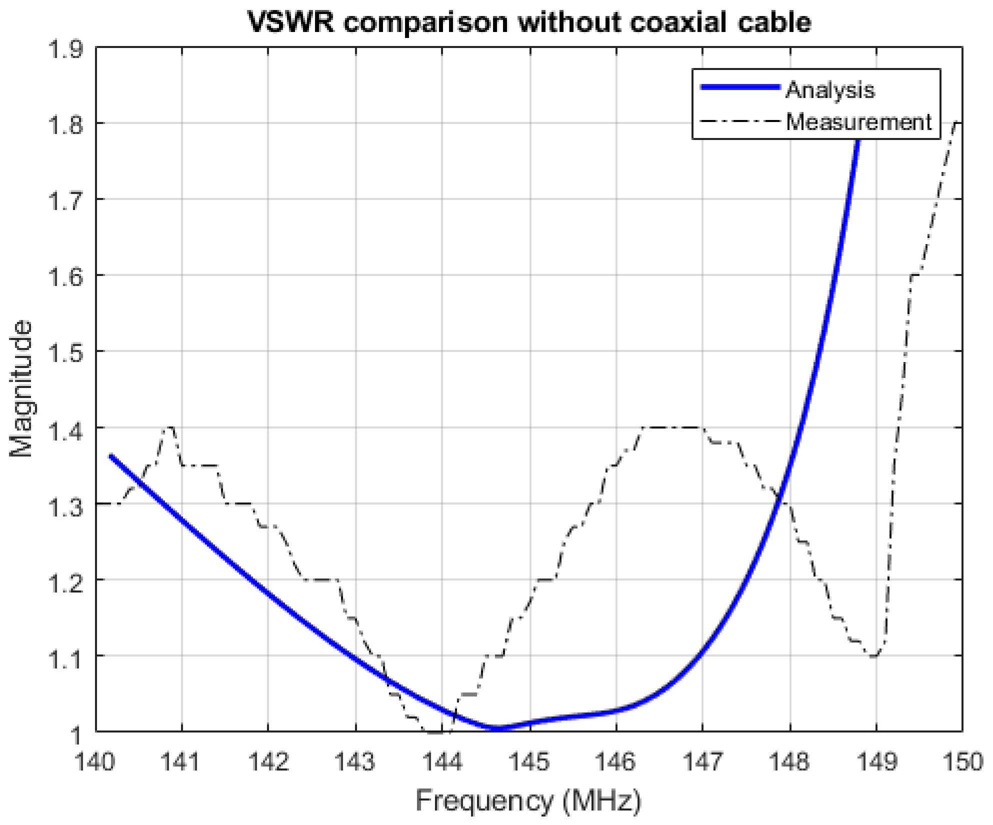
Input Reflection Coefficient of Optimized Antenna

The input reflection coefficient for the optimized Yagi-Uda antenna is computed and plotted relative to the reference impedance of 50Ω. A value of -10 dB or lower is considered as a good impedance match as per the simulation the reflection coefficient is around -37 dB throghout the VHF band and -50.7 dB at 145.5MHz.

```
s = sparameters(yagidesign,freq,Z0);
fig5 = figure;
rfplot(s);
```



```
vswr_measured = csvread('SWR_Values_Sep_15.csv',1,0);
figure
vswr(yagidesign,freq,Z0)
hold on
plot(vswr_measured(:,1),vswr_measured(:,2),'k-.')
legend('Analysis','Measurement')
title('VSWR comparison without coaxial cable')
ylabel('Magnitude')
```



Model the Effect of Coaxial Cable

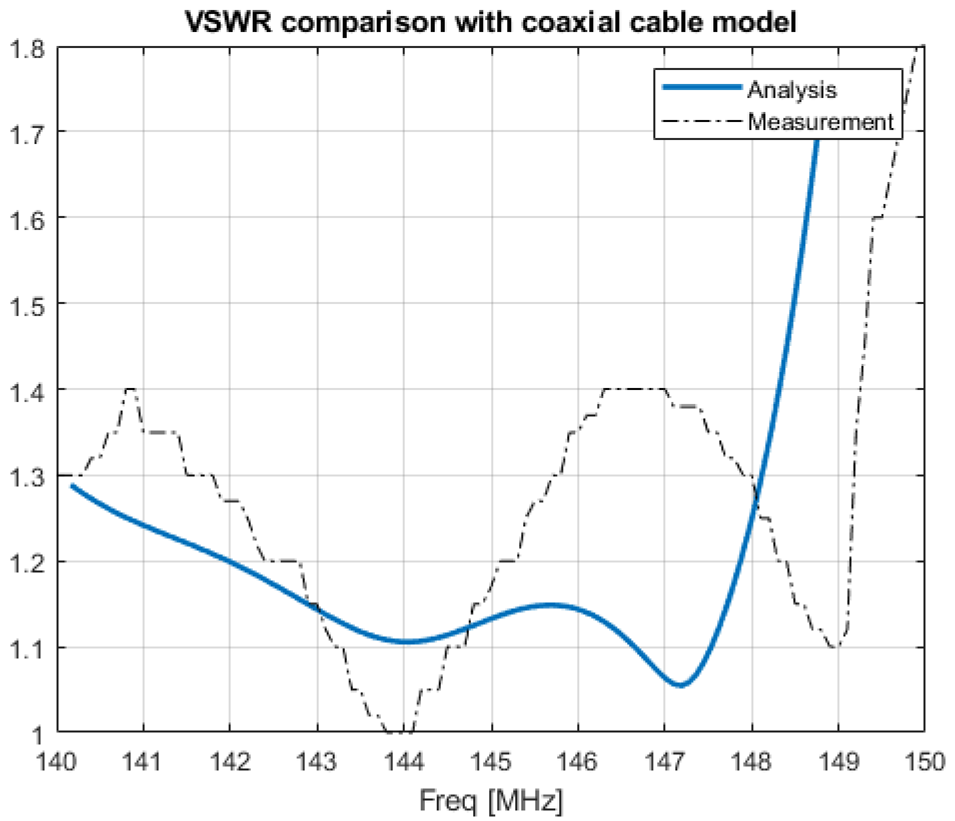
The coaxial cable connected to the yagi is a RG-58/U with a characteristic impedance of 50Ω with a velocity factor of 0.66. The losses introduced by the coaxial cable will also contribute the measured SWR value. Since the measurements measurements were made at the shack, there will be small diiference between the simulated and measured values. Factors like the dielectric constant of the antenna boom and the additional metal structures like the nuts and bolts used to fasten the elements to the boom are not considered during simulation.

```
out_radius = 3.51e-3;
in_radius = 0.91e-3;
eps_r = 2.95;
line_length = 5.05*lambda;
coax_cable = rfckt.coaxial;
coax_cable.OuterRadius = out_radius;
coax_cable.InnerRadius = in_radius;
coax_cable.EpsilonR = eps_r;
coax_cable.LossTangent = 2e-4;
coax_cable.LineLength = line_length;
```

Cascade the sparameters for the designed yagi with the coaxial cable

```
Zyagi =impedance(yagidesign,freq);
analyze(coax_cable,freq,Zyagi);
figure
hline = plot(coax_cable,'VSWRin','None');
hline = 
  Line (VSWR_{in}) with properties:

              Color: [0 0.4470 0.7410]
          LineStyle: '-'
          LineWidth: 0.5000
             Marker: 'none'
         MarkerSize: 6
    MarkerFaceColor: 'none'
              XData: [1×101 double]
              YData: [1×101 double]
              ZData: [1×0 double]

  Show all properties
```

```
hline.LineWidth = 2;
hold on
plot(vswr_measured(:,1),vswr_measured(:,2),'k-.')
legend('Analysis','Measurement')
title('VSWR comparison with coaxial cable model')
```



Tabulating Initial and Optimized Design

Tabulate the initial design guesses and the final optimized design values.

```
yagiparam=  {'Reflector Length';'Reflector Spacing';'Director Spacing - 1';
             'Director Spacing - 2';'Director Spacing - 3';
             'Director Spacing - 4'};
```

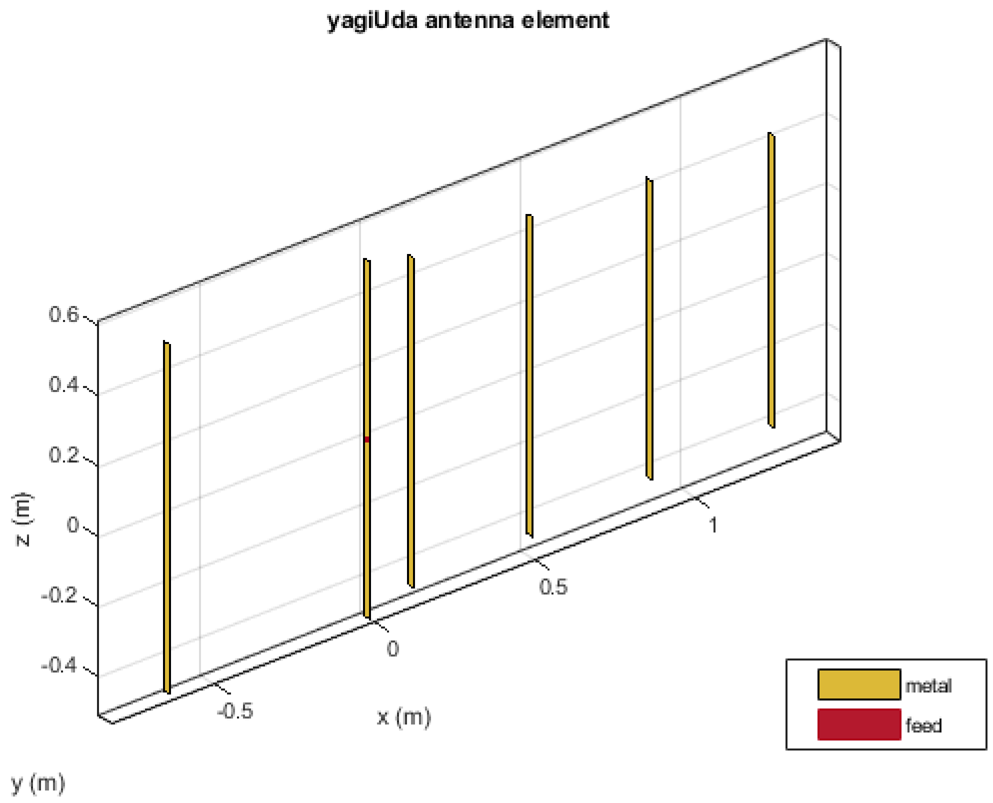
```
initialdesign = initialdesign';
optimdesign = optimdesign';
Tgeometry = table(initialdesign,optimdesign,'RowNames',yagiparam)
```

```
Tgeometry = 6×2 table
```

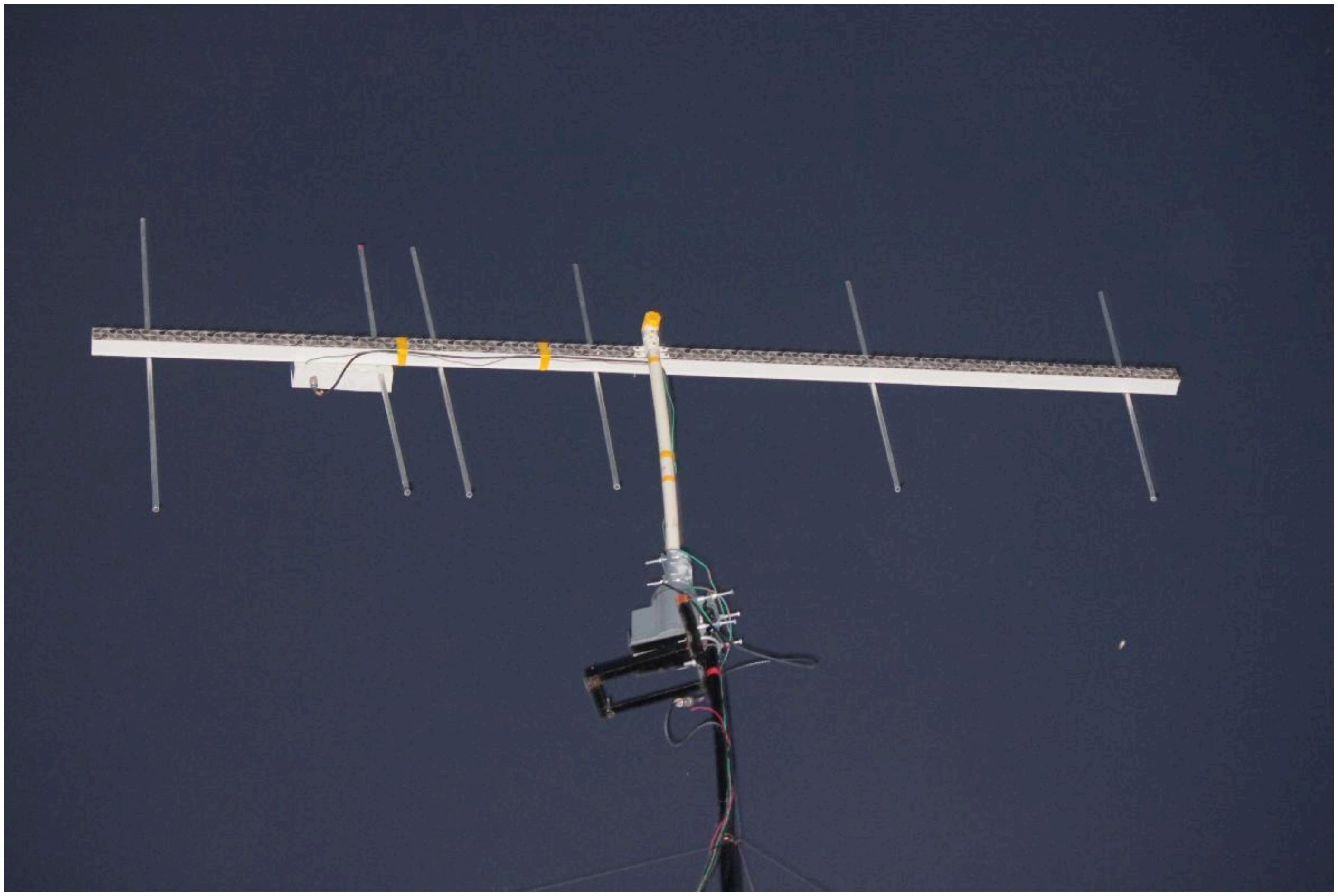
	initialdesign	optimdesign
1 Reflector Length	0.5187	0.9952
2 Reflector Spacing	0.7261	0.6310
3 Director Spacing - 1	0.3112	0.1379
4 Director Spacing - 2	0.4149	0.3700
5 Director Spacing - 3	0.6224	0.3742
6 Director Spacing - 4	0.6224	0.3849

Designed and Fabricated Antennas

```
figure
yagidesign.Tilt = 90;
yagidesign.TiltAxis = [0 1 0];
show(yagidesign)
```



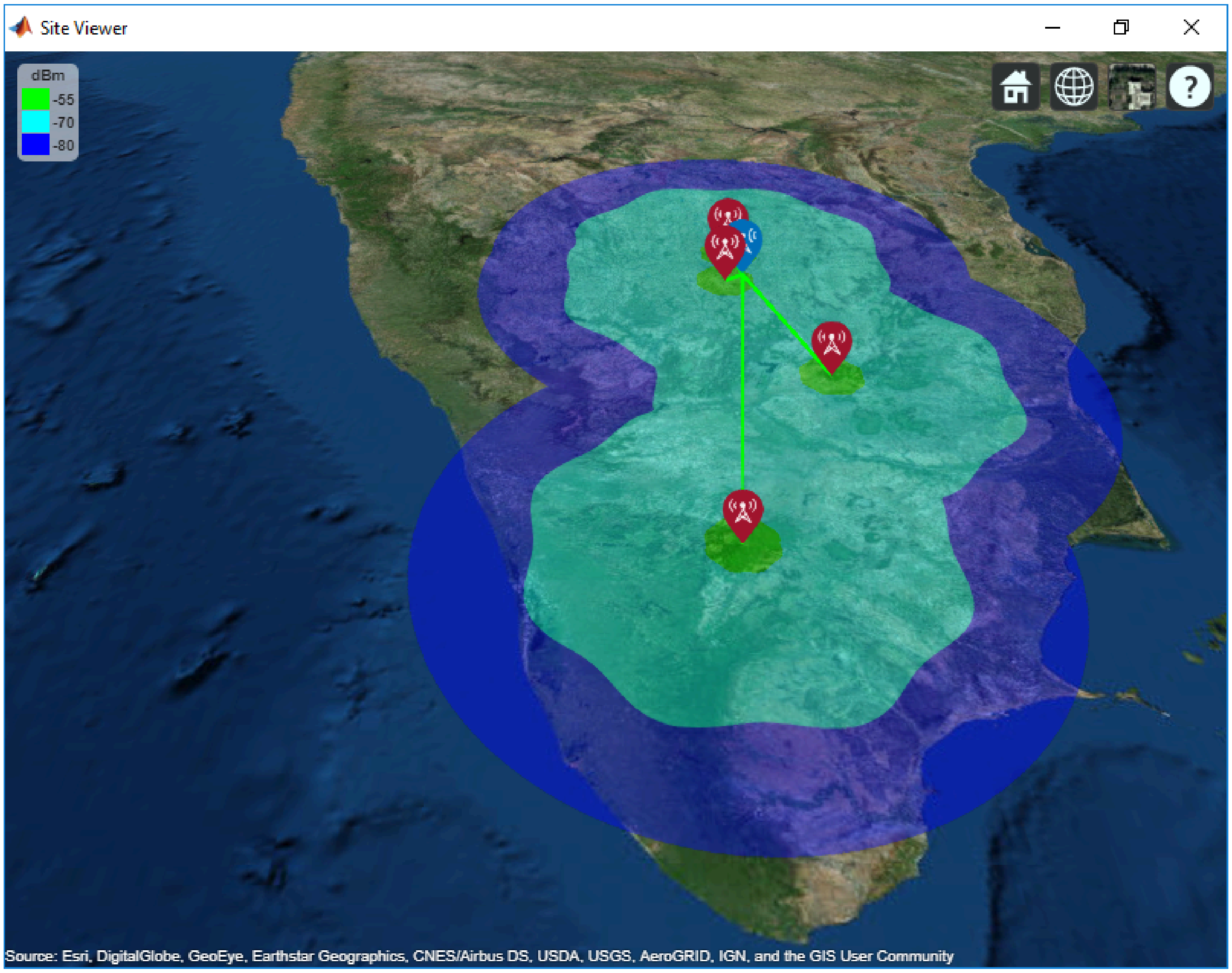
The below picture shows the antenna fabricated and mounted on a antenna rotator on top of a 10 meter high mast.



Propagation

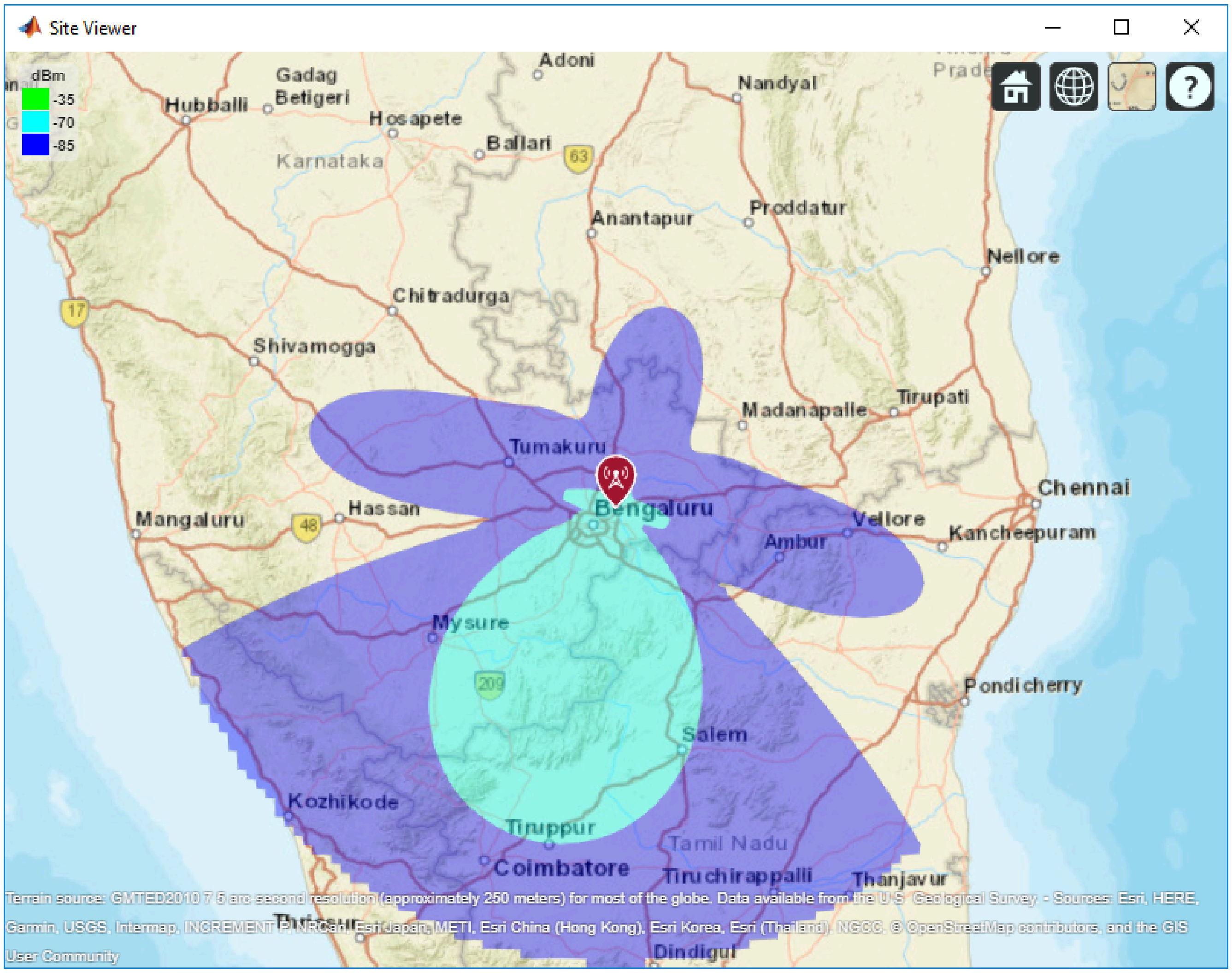
The 4 commonly accesed reapeters along with their simulated coverage area is shown below. The 'Freescape' propagation model is considered in this simulation, for a more detailed model other propagation model like 'Longley-rice' model can be considered that can also consider the impact of temperature, humidity and conductivity of ground. The repeaters are modeled with an omni directional antenna, while the shack is modled with YAgii optimized earlier.

```
% VHF Coverage

names = ["VU2TCD","VU2RSB","VU2TWO","VU2KOD"];
lats = [11.779957,13.386614,12.974499,10.238974];
lons = [78.217233,77.667272,77.609646,77.488515 ];
TransFreq = 145e6;
TransPower = 25;
strongSignal = -35;
strongSignalColor = "green";
moderateSignal = -70;
moderateSignalColor = "cyan";
weakSignal = -85;
weakSignalColor = "blue";
repeater_sites = txsite('Name',names,'Latitude',lats,'Longitude',lons,...
    'Antenna',monopole,'TransmitterFrequency',TransFreq,'TransmitterPower',TransPower)
repeater_sites.show
repeater_sites.coverage('PropagationModel','freespace','SignalStrengths',...
    [strongSignal,moderateSignal,weakSignal],'Colors', [strongSignalColor,...
    moderateSignalColor,weakSignalColor],'MaxRange',200e3,'ReceiverGain',2)
```


Site Viewer
dBm
🟩 -55
🟦 -70
🟦 -80
Source: Esri, DigitalGlobe, GeoEye, Earthstar Geographics, CNES/Airbus DS, USDA, USGS, AeroGRID, IGN, and the GIS User Community

The antenna is oriented in sucha way that it would cover all the repeaters of interest with rotating the antenna.

```
shack_site = txsite('Name',names,'Latitude',13.064183,'Longitude',77.747958,...
    'Antenna',yagidesign,'TransmitterFrequency',145e6,'TransmitterPower',10,...
    'AntennaAngle',-105);
shack_site.coverage('PropagationModel','freespace','SignalStrengths',...
    [strongSignal,moderateSignal,weakSignal],...
    'Colors', [strongSignalColor,moderateSignalColor,weakSignalColor],...
    'MaxRange',300e3,'ReceiverGain',2);
```


Site Viewer
dBm
-35
-70
-85
Bengaluru
Chennai
Mysure
Salem
Coimbatore
Terrain source: GMTED2010 7.5 arc-second resolution (approximately 250 meters) for most of the globe. Data available from the U.S. Geological Survey - Sources: Esri, HERE, Garmin, USGS, Intermap, INCREMENT P, NRCan, Esri Japan, METI, Esri China (Hong Kong), Esri Korea, Esri (Thailand), NGCC, © OpenStreetMap contributors, and the GIS User Community

Conclusion

This article capture how an antenna can be designed and optimized based on specifc constraints like impedance, boom length and pattern. This article also discuss how the effects of coaxial cable and other losses can be considered along with various propagation models to estimate coverage. Propagation models like 'Longley-rice' can provide more realisitic coverge estmate considering the terrain and ground conditions.

Reference

[1] C. A. Balanis, Antenna Theory. Analysis and Design, p. 514, Wiley, New York, 3rd Edition, 2005