VISUALIZING THE 3D POLAR POWER PATTERNS AND EXCITATIONS OF PLANAR ARRAYS WITH MATLAB

J. C. BRÉGAINS, F. ARES, AND E. MORENO

Radiating Systems Group, Department of Applied Physics,
15782 Campus Sur, Univ. of Santiago de Compostela (Spain)
e-mails: (Ares) faares@usc.es - (Brégains) fajulio@usc.es – (Moreno) famoreno@usc.es

ABSTRACT

A few lines of MATLAB M-code suffice to create 3D polar plots of the power patterns of planar arrays together with 3D plots of the amplitudes and phases of their excitations.

1. INTRODUCTION

Many antenna designers are familiar with MATLAB [1] and use it in their everyday work. However, many also shy from attempting to exploit its graphical capabilities to the full. Here we present some quite simple M code that straightforwardly performs a task that is often desirable but is often left undone or is performed less efficiently by exporting numerical results to an external graphical program [2, 3]: production of the 3D polar plot of the power pattern of a planar or linear antenna array of identical radiating elements (or of a single element) given only their positions and excitations and the corresponding element factor. For good measure, the program also produces 3D plots of the amplitudes and phases of the element excitations. The code presented has been neither optimized nor worked up as a stand-alone application or function (we keep it always to hand in the MATLAB working directory as an M-file named planar_pow3d.m, modifying certain parameters in the code itself whenever necessary), and nor does it include exhaustive measures to prevent undesired results when given illegitimate data such as negative excitation amplitudes; but anyone familiar with MATLAB will easily be able to adapt it to their own requirements.

2. THE CODE

In this section we run through the M code for Figs.1-3, which show power and excitation plots for a 20 x 20 planar array of isotropic elements located 0.5λ apart, the excitations of which are the result of multiplying (i.e., by using separable distributions [4]) those of a -20 dB Chebyshev linear array [4] with those of a linear array generating a flat-topped pattern with a half-power beamwidth of 43.2º and a maximum side lobe level of -20 dB [5]. To facilitate commentary, line numbers have been added to the code.

Lines 1 and 2 clear memory and read the positions and excitations of the array elements from a text file that in this example is named pos_ excit_planar.txt, in which the data for each radiating element have previously been introduced in the format

x_coord TAB y_coord TAB amplitude TAB phase \n

(except for the last element, for which \n should be omitted). Here TAB and \n indicate the tabulation and newline characters, the x and y coordinates are given in units of one wavelength (the array is assumed to lie in the x-y plane with its centre at the origin), and phases are in radians.

```
1.  clear;
2.  [X,Y,AMP,PHA]=textread('pos_excit_planar.txt','%f %f %f %f');
```

The number of elements in the array does not have to be specified by the user, but is determined by the program itself:

```
3.  nelem=size(AMP,1);
```

In order to be able to fix the limits of the graphs of excitation amplitudes and phases, we now determine the maximum amplitude and the maximum distance of any element from the array centre, and we normalize the amplitudes:

```
4.  dist=sqrt(X.*X+Y.*Y); dmax=max(max(dist)); if dmax==0 dmax=1; end;
    max_amp=max(max(AMP));
5.  AMP=AMP/max_amp;
```

If dmax is zero (i.e. if there is just a single element at the origin, a set-up that can be used to plot an element factor), then dmax is set to unity to prevent a run-time error when plotting the excitation.

The first graph drawn is that of the excitation amplitudes (Fig.2):

```
6.  fig_amp=figure('Name','Array amplitude distribution','NumberTitle', 'off');
7.  colordef(fig_amp,'white');
8.  plot3(X,Y,AMP,'bo','MarkerFaceColor','white','MarkerSize',12);
9.  axis([-dmax dmax -dmax dmax 0 1]);
10. tick_pos=(0:0.2:1);
11. set(gca,'ZTick',tick_pos,'ZTickLabel', tick_pos);
12. xlabel('x/\lambda','FontSize',12,'position',[0 1.4*dmax 0],
    'HorizontalAlignment', 'center');
```

```
13. ylabel('y/\lambda','FontSize',12,'position',[1.4*dmax 0 0],
    'HorizontalAlignment', 'center');
14. zlabel('Normalized Amplitude','FontSize',12);
15. grid on; axis square; box on;
16. camproj('perspective'); view(140,20);
```

Lines 6-8 open a graphics window for an unnumbered figure called "Array amplitude distribution", assign its identifying "handle" to the variable fig_amp, set its background colour to white, and plot the element amplitudes, as a function of the element positions, as 12-point blue beads with white faces; lines 9-14 set the axis ranges, styles and labels (in line 11, the function gca returns the handle of the current axes); and lines 15 and 16 enclose the graph in a cubic box with three ruled faces, and specify that this box be viewed in perspective from an elevation of 20º at 140º from the negative y axis.

The excitation phases are plotted similarly (but in red) following their conversion to degrees in line 17 (Fig.3)[1].

```
17. PHADEG=180*PHA/pi;
18. fig_pha=figure('Name','Array phase distribution','NumberTitle', 'off');
19. colordef(fig_pha,'white');
20. plot3(X,Y,PHADEG,'ro','MarkerFaceColor','white','MarkerSize',12);
21. axis([-dmax dmax -dmax dmax 0 360]);
22. tick_pos=(0:60:360);
23. set(gca,'ZTick',tick_pos,'ZTickLabel', tick_pos);
24. xlabel('x/\lambda','FontSize',12,'position',[0 1.4*dmax 0],
    'HorizontalAlignment', 'center');
25. ylabel('y/\lambda','FontSize',12,'position',[1.4*dmax 0 0],
    'HorizontalAlignment', 'center');
26. zlabel('Phase (degrees)','FontSize',12);
27. grid on; axis square; box on;
28. camproj('perspective'); view(140,20);
```

Now the main job begins: the 3D polar plot of the power pattern in dB relative to peak (Fig.1). Lines 29-33 set the power cut-off level in dB (the absolute value is given, i.e. 40 instead of -40); specify the resolution and the initial and final values of the azimuthal and polar angle variables (the user is given a choice between plotting the pattern over the whole sphere or only over the northern

[1] It is intended that the phases of the elements have been previously constrained to a [0, 2π] range.

hemisphere); and use these specifications to set up vectors of the values of θ and ϕ defining the vertices (θ,ϕ) of the plot.

```
29.  level_db= 40; n_points_theta=151; n_points_phi=151; itheta=0; iphi=0;
     fphi=2*pi;
30.  whole=input('Hemisphere (0) or whole space (1)?:\n');
31.  if whole==0; ftheta=pi/2; else ftheta=pi; end;
32.  THETA=linspace(itheta,ftheta,n_points_theta)';
33.  PHI=linspace(iphi,fphi,n_points_phi);
```

Lines 34-43 calculate the field at each of the plot vertices using the standard expression

$$F(\theta,\phi)=f_e(\theta,\phi)\sum_{n=1}^{N}A_n e^{i[\Phi_n+k\sin\theta(x_n\cos\phi+y_n\sin\phi)]} \qquad (1)$$

and store the field amplitudes in the matrix RADIUS. In eq. (1), f_e is the scalar element factor, k is the wavenumber, and A_n, Φ_n, x_n and y_n are the amplitude, phase, x coordinate and y coordinate of the n-th element of the array. Note that $k = 2\pi$ identically (because the unit of length is the wavelength λ); and that the element factor is obtained from a function, felem, that is assumed to have been placed previously in the working directory (though if the elements are isotropic line 40 can simply be omitted).

```
34.  for m=1:n_points_theta,
35.     for n=1:n_points_phi,
36.        temp_field=0;
37.        for r=1:nelem,
38.              temp_field=temp_field+AMP(r)*exp(i*(PHA(r)+2*pi*(sin(THETA(m))*
                 (X(r)*cos(PHI(n))+Y(r)*sin(PHI(n))))));
39.         end;
40.         temp_field=temp_field*felem(THETA(m),PHI(n));
41.         RADIUS(m,n)=abs(temp_field);
42.     end;
43.  end;
```

The field amplitudes are then normalized relative to peak (line 44) and converted to dB in such a way that sub-cutoff values disappear and the peak value (0 dB) will be plotted with the largest radius in the polar plot (lines 45-51).

```
44.  RADIUS=RADIUS/max(max(RADIUS));
```

```
45. for m=1:n_points_theta,
46.     for n=1:n_points_phi,
47.         if(RADIUS(m,n)<10^(-level_db/20)) RADIUS(m,n)=0;
48.         else RADIUS(m,n)=20*log10(RADIUS(m,n))+level_db;
49.         end;
50.     end;
51. end;
```

Finally, the (θ,ϕ) coordinates and the corresponding power values (RADIUS) are converted to Cartesian coordinates (lines 52-54), and these are plotted (lines 55-64) as an illuminated bluish surface –setting Facecolor with the values given within the corresponding brackets, in which [Red Green Blue] levels are specified– with a transparency parameter value of 0.6.

```
52. XX=sin(THETA)*cos(PHI).*RADIUS;
53. YY=sin(THETA)*sin(PHI).*RADIUS;
54. ZZ=cos(THETA)*ones(1,n_points_phi).*RADIUS;
55. fig_power=figure('Name','3D Polar Power Pattern in dBs','NumberTitle',
    'off');
56. ps=surf(XX,YY,ZZ, 'EdgeColor', 'blue', 'EdgeAlpha', 0.60, 'LineWidth',
    1.00);
57. set(ps,'FaceColor',[0.169 0.502 1.000]); alpha(0.60); title('Normalized
    Power Pattern (dB)');
58. tick_pos=(-level_db:10:level_db); ticks=abs(tick_pos)-level_db;
59. set(gca,'XTick',tick_pos,'XTickLabel',ticks);
60. set(gca,'YTick',tick_pos,'YTickLabel',ticks);
61. set(gca,'ZTick',tick_pos,'ZTickLabel',ticks);
62. set(gca,'XColor', 'blue','YColor', 'blue','ZColor', 'blue');
63. axis equal; axis auto; box on;
64. camlight; lightangle(0,45); lighting gouraud; camproj('perspective');
65. clear;
```

3. FURTHER EXAMPLES

Fig.4 shows an example of a pattern plotted in the whole of space. It is in fact the flat-topped pattern of one of the linear arrays that are multiplied together to form the planar array of Figs.2 and 3; its positions and excitations are listed in Table 1. The striations on the face corresponding to the roll-off from the main beam are oscillations that could be avoided by increasing the number of vertices of

the plot. Fig.5 shows the pattern of an array of 20 uniformly excited antennas with element factor fe = $\cos^2\theta$ that are equispaced on a circle of radius 1.76λ.

4. AND NOW THE BAD NEWS

The reader will already have noticed that in spite of Figs. 1, 4 and 5 being polar plots, axes and grids are drawn as though for Cartesian plots. The power levels indicated on the edges of the box frames enclosing the patterns actually refer to points on axes through the origin (the centre of the array) that are parallel to these edges, and the grids are included merely to help guide the eye to these undrawn axes. We do not claim that this is the perfect solution to the thorny problem of indicating magnitude, i.e. radius, in 3D polar plots; other designers may prefer other solutions, like color–dependence, for example.

Another issue in everyday work is, of course, speed. While the screen plot of Fig.4 only took 5 seconds using 91 θ values and 91 ϕ values, plotting Fig.1 with 151x151 = 22,801 vertices took 2 minutes 23 seconds (in a PC running at 1.8 GHz). The actual plotting could of course be speeded up enormously by translating to c-code (and compiling it, by means of mcc function) instead of using the MATLAB interpreter, but for most patterns this gain in speed will not compensate for the time spent doing the compiling.

5. CONCLUDING REMARKS

3D polar plots of radiation power patterns are an effective means of presenting the results of the antenna designer's work, and can also serve as useful diagnostic orientation. They can be created within MATLAB by making simple modifications to the code presented here to suit each particular case. The code can be downloaded as a file called Visualizing.zip from the following internet address:

http://www.ece.osu.edu/~volakis

For those interested in checking the performance of the code without having to create their own data files, the data files for the arrays producing the patterns of Figs. 1, 4 and 5 (pos_ excit_planar.txt, pos_excit_linear.txt and pos_ excit_circular.txt, respectively) were included in the named zip file. More details about the graphic functions can be obtained from [6].

6. REFERENCES

[1] S. N. Makarov: “Antenna and EM Modeling with MATLAB” Wiely-Interscience, New York, 2002.

[2] Michael Carr: “Visualization with OpenGL: 3D Made Easy”, IEEE Antennas and Propagation Magazine, Vol. 39, No. 4, pp. 116-120, 1997.

[3] T. J. Peters: “A Fast Algorithm for Plotting and Contour Filling Radiation Patterns in Three Dimensions”, IEEE Transactions on Antennas and Propagation, Vol. 40, No. 4, pp. 453-456, 1992.

[4] R. S. Elliott: “Antenna Theory and Design”, IEEE-Press, Wiley-Interscience, Revised Edition, 2003.

[5] H. J. Orchard, R. S. Elliott, and G. J. Stern: “Optimizing the Synthesis of Shaped Beam Antenna Patterns”, IEE Proc. H., Vol. 132, pp. 63-68, 1985.

[6] The MATLAB Group, Inc.: “Using MATLAB Graphics: 3-D Visualization”, pdf document (printable version taken from Help menu in MATLAB), 2002.

LEGENDS FOR FIGURES AND TABLES

Table 1. Element positions and excitations of the linear array producing the flat-topped power pattern of Fig.4. It is intended that the program will normalize the amplitudes to the maximum value.

Figure 1. 3D polar plot of the power pattern of a 20 x 20 planar array of isotropic elements located 0.5λ apart, the excitations of which are the result of multiplying those of a -20 dB Chebyshev linear array with those of a linear array generating a flat-topped pattern with a half-power beamwidth of $43.2°$ and a maximum side lobe level of -20 dB.

Figure 2. The excitation amplitudes of the array generating the pattern of Fig.1.

Figure 3. The excitation phases of the array generating the pattern of Fig.1.

Figure 4. 3D polar plot of the flat-topped power pattern of a 20 element linear array of isotropic elements located 0.5λ apart (one of the linear factor arrays of the array of Fig.1). Half-power beamwidth, $43.2°$; maximum side lobe level, -20 dB.

Figure 5. 3D polar plot of the power pattern of a circular array of 20 uniformly excited antennas with element factor $f_e = \cos^2\theta$ that are equispaced on a circle of radius 1.76λ.

x coordinate (λ)	y coordinate (λ)	Relative Amplitude	Phase (radians)
0.0000	-4.7500	1.0000	0.0000
0.0000	-4.2500	1.7973	0.0000
0.0000	-3.7500	2.1522	0.0000
0.0000	-3.2500	1.5574	0.0000
0.0000	-2.7500	0.3439	0.0000
0.0000	-2.2500	0.6243	3.1416
0.0000	-1.7500	0.7248	3.1416
0.0000	-1.2500	0.1324	3.1416
0.0000	-0.7500	0.4209	0.0000
0.0000	-0.2500	0.4095	0.0000
0.0000	0.2500	0.0199	3.1416
0.0000	0.7500	0.3279	3.1416
0.0000	1.2500	0.2170	3.1416
0.0000	1.7500	0.1027	0.0000
0.0000	2.2500	0.2411	0.0000
0.0000	2.7500	0.0877	0.0000
0.0000	3.2500	0.1269	3.1416
0.0000	3.7500	0.1624	3.1416
0.0000	4.2500	0.0421	3.1416
0.0000	4.7500	0.3198	0.0000

TABLE 1

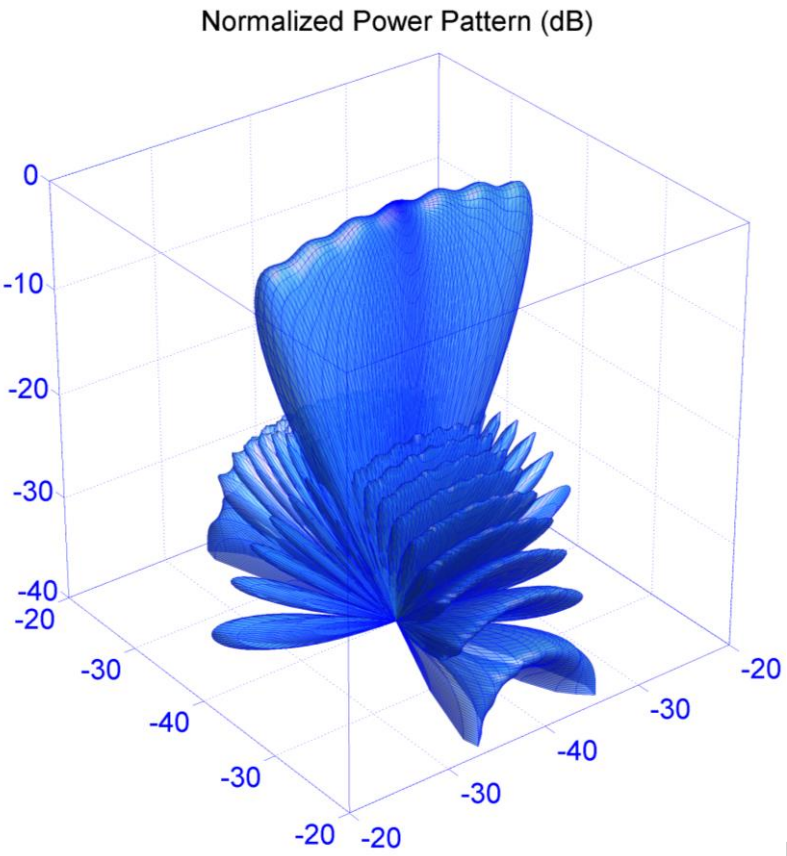

FIGURE 1.

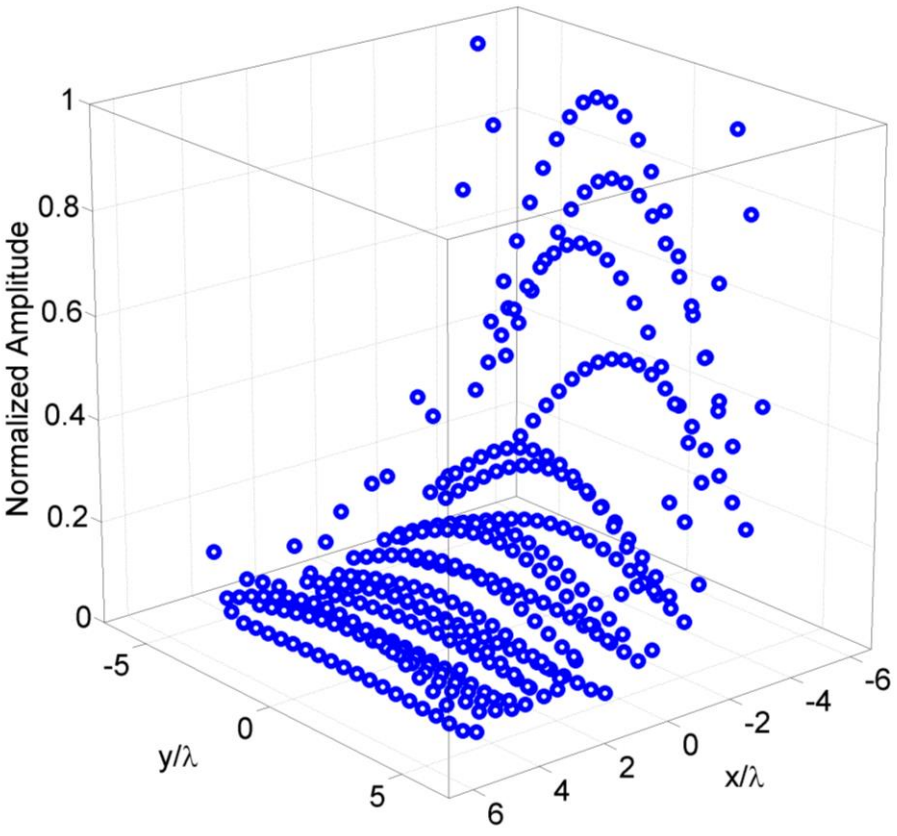

FIGURE 2.

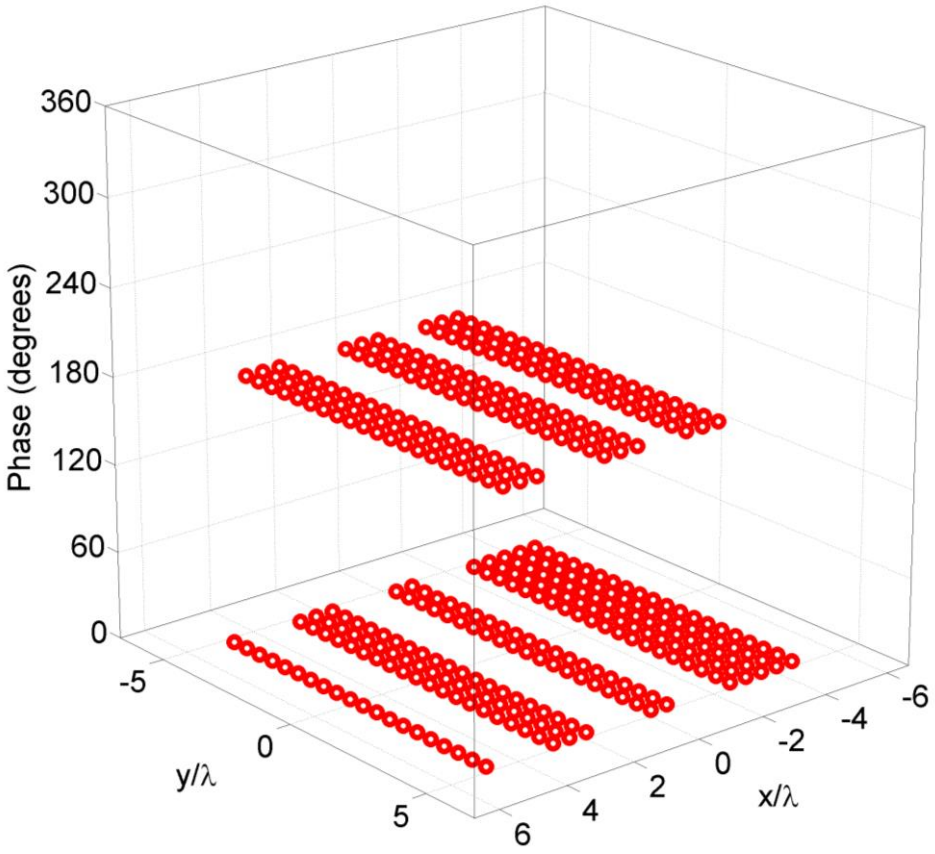

FIGURE 3.

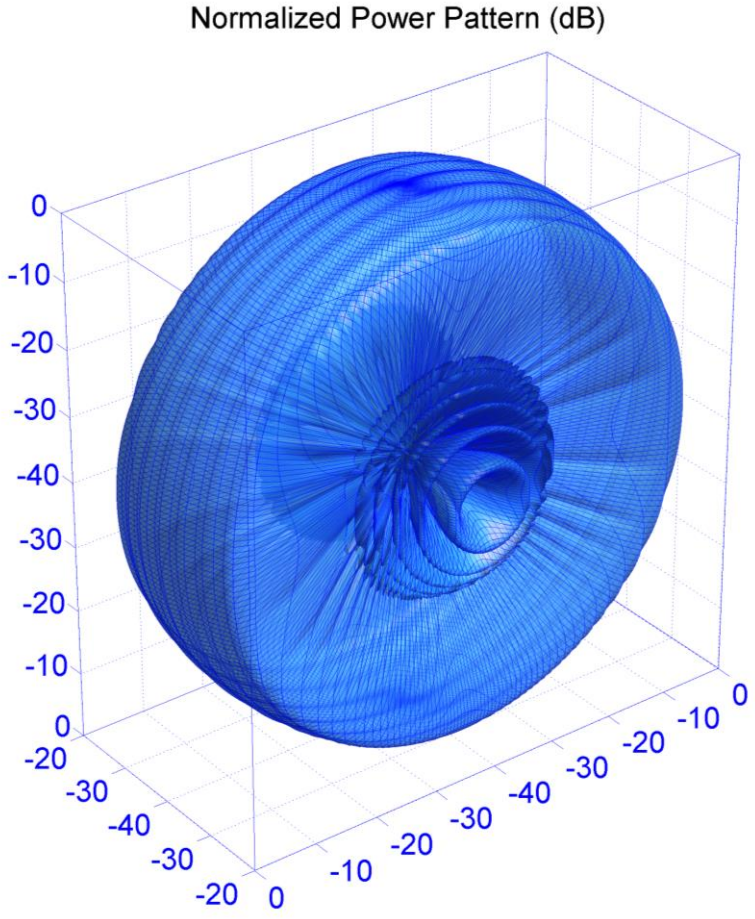

FIGURE 4.

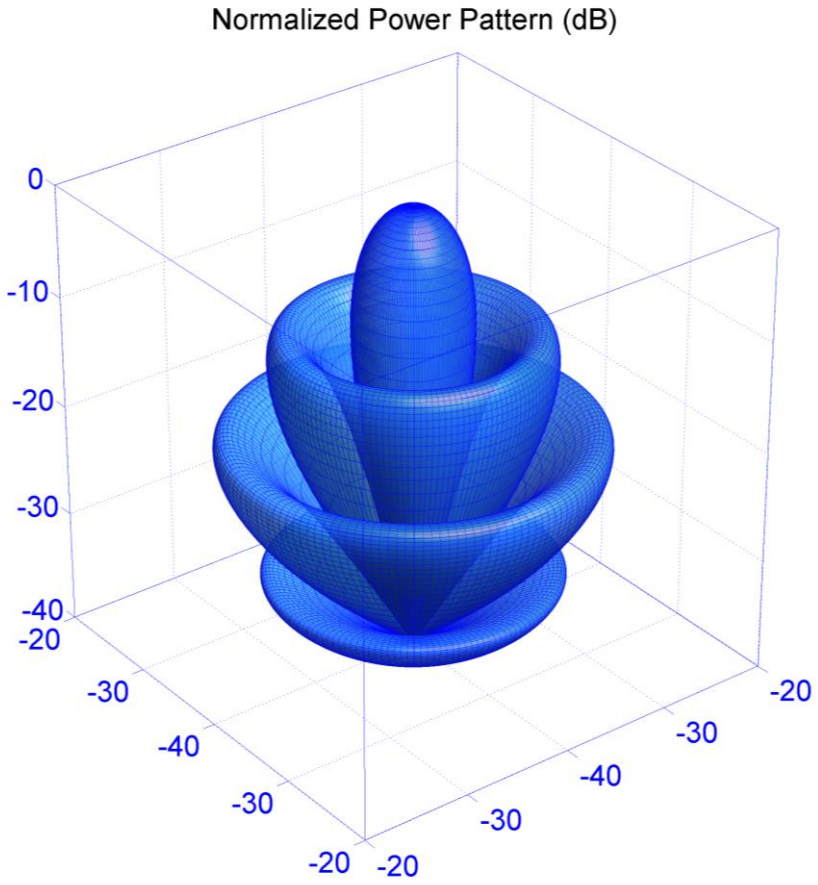
Normalized Power Pattern (dB)
0
-10
-20
-30
-40
-20
-30
-40
-30
-20
-20
-30
-40
-30
-20
-20

FIGURE 5.