Channel Modeling of a Mobile Repeater for the Shinkansen Bullet Train

Brad Schafer, ‘Minister of Engineering’ SchaferIsTheMan Engineering Inc.

I. Introduction

This paper presents the results of the examination of the signal received by two monopole antennas attached to the Shinkansen high speed bullet train as the train moves through different terrains and receives signals from different angles. The results show the effect of small-scale fading on the received signal and provide a justification for the design of an adequate power sampling system to implement selection diversity between the two received signals.

II. System Model

Multipath shape factors were used to model the system and determine the effects of small-scale fading on the received signal. The principal shape factors: the angular spread Λ, the angular constriction γ, and the azimuthal direction of maximum fading Θ_{max} are directly related to the average rate at which the signal fades [1]. The shape factors can be used to determine the level-crossing rate and the average fade duration.

The system will be modeled and examined in four operating environments consisting of two terrain conditions and two directional conditions. The first terrain type is a rural terrain in which there is relatively minimal multipath. The second type of terrain is either mountainous or a high-urban region where multipath effects are more considerable. As the train moves forward the orientation of the train with respect to the base station will change. To account for this variation the angle-of-arrival will be considered with the base station is directly in front of the train and orthogonal to the path of the train. A train speed of 260 kilometers per hour will be used when modeling the system. The four operating environments are summarized in Appendix 1.

III. System Modeling – Shape Factors

The shape factors are functions of the complex Fourier coefficients

$$F_n = \int p(\theta_x)\exp(j\cdot n\cdot\theta_x)\cdot d\theta_x \qquad (1)$$

The Fourier coefficients were calculated for the four system conditions. The Matlab code used to calculate the coefficients is included in Appendix 2. The first three coefficients for each system condition is shown in Table 1.

	F_0	F_1	F_2
Rural, Θ_0 = 0	0.10472	0.10443	0.1036
Rural, Θ_0 = 90	0.10472	0.10443i	-0.1036
Urban, Θ_0 = 0	2.2313	1.1411	0.1203
Urban, Θ_0 = 90	2.2313	1.1411i	-0.1203

Table 1. Fourier coefficients for model conditions.

Using the Fourier coefficients the angular spread Λ, angular constriction γ, and azimuthal direction of maximum fading Θ_{max} were calculated using the appropriate coefficient equations. [1] The Matlab code used to generate these shape factors is included in Appendix 3. The results for each system condition are shown in Table 2.

$$\Lambda = \sqrt{1 - \frac{|F_1|^2}{F_0^2}} \qquad (2)$$

$$\gamma = \frac{|F_0F_2 - F_1^2|}{F_0^2 - |F_1|^2} \qquad (3)$$

$$\theta_{\max} = \frac{1}{2}\cdot\arg\{F_0F_2 - F_1^2\} \qquad (4)$$

	Λ	γ	Θ_{max} (deg)
Rural, Θ_0 = 0	0.0744	0.9682	90
Rural, Θ_0 = 90	0.0744	0.9682	0
Urban, Θ_0 = 0	0.8593	0.2812	90
Urban, Θ_0 = 90	0.8593	0.2812	0

Table 2. Shape Factors for model conditions

The shape factors were used to determine the level-crossing rate in a Rayleigh fading channel using

$$N_R = \frac{\sqrt{2\cdot\pi}v\Lambda\rho}{\lambda}\sqrt{1+\gamma\cos(2[\theta-\theta_{MAX}])}e^{(-\rho^2)} \qquad (5)$$

The average fade duration was calculated using using

$$\bar{\tau} = \frac{\lambda[\exp(\rho^2)-1]}{\sqrt{2\pi}v\rho\Lambda\sqrt{1+\gamma\cos([2(\theta-\theta_{MAX})])}} \qquad (5)$$

Figure 1 shows the resulting level crossing rate for the for model conditions. The Matlab code used to generate this graph is included in Appendix 4. As expected when the threshold level is near the normalized threshold the small scale fading signal produces the maximum level crossing rate. The level crossing rate is higher in the urban environment rather than the rural environment due to an increase in the multipath waves being received. The level crossing rate was also higher when the train traveled inline with the base station signal rather than orthogonally. Traveling inline produced a greater relative velocity between the base station and the train receiver and therefore increased the frequency of the signal fade.

The average fade duration versus normalized threshold level is shown in Figure 1. The Matlab code used to generate this plots are shown in Appendix 5. As the threshold level is increased more the received signal is received below the threshold value and the average fade duration increases.

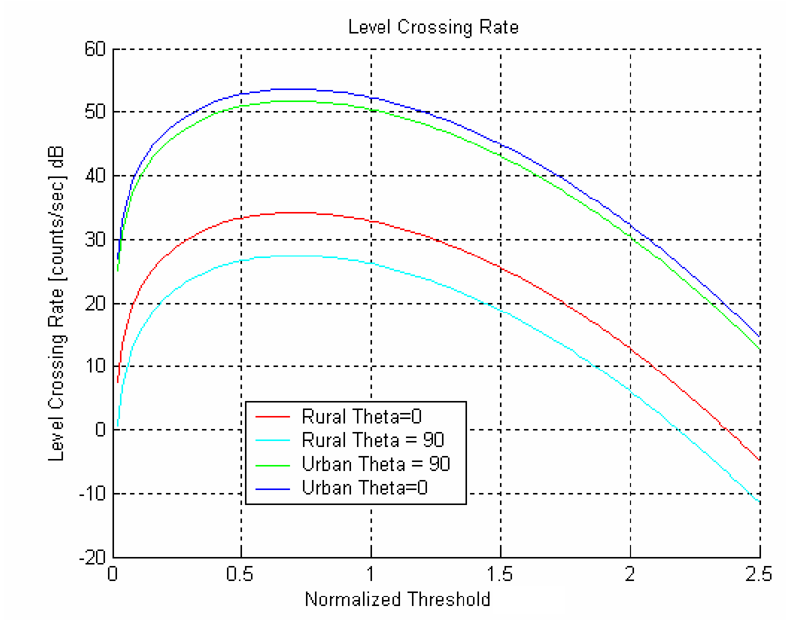


Figure 1. Level Crossing Rate vs. Normalized Threshold Level

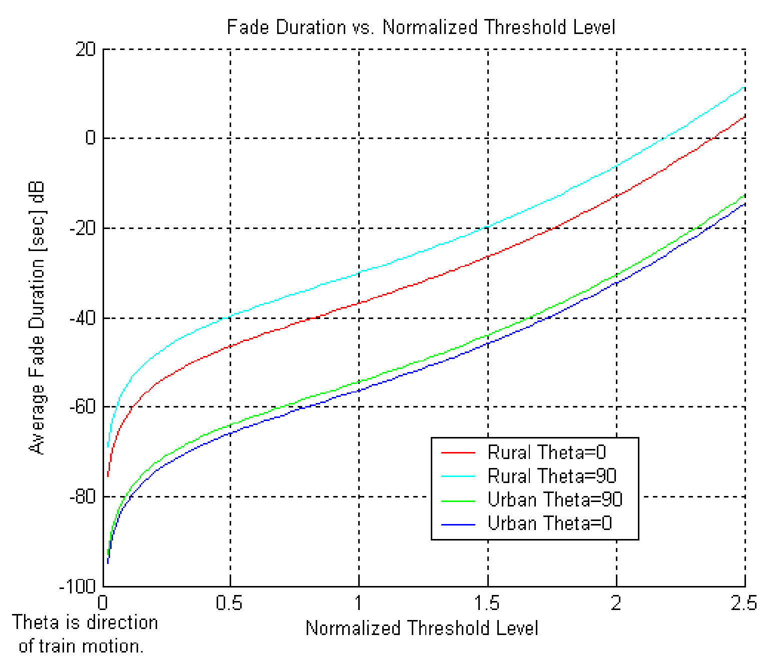


Figure 2. Average Fade Duration vs. Normalized Threshold Level

IV. System Modeling – Modeled Signal

The signal power received at each antenna on the train is a function of the carrier signal frequency, the direction of the train, the angle of arrival from the base station, and the received multipath signals. A matlab script was used to model each of the model conditions and calculate the signal power received by each antenna during a one millisecond time frame. Next, the threshold selection output was determined from the two antenna signals. Figures 3 through 6 show the modeled reception signal and corresponding threshold selection output for each system condition. Figures 3 through 6 are also included as full page figures in Appendix 6. The Matlab code used to generate the power signals is included in Appendix 7.

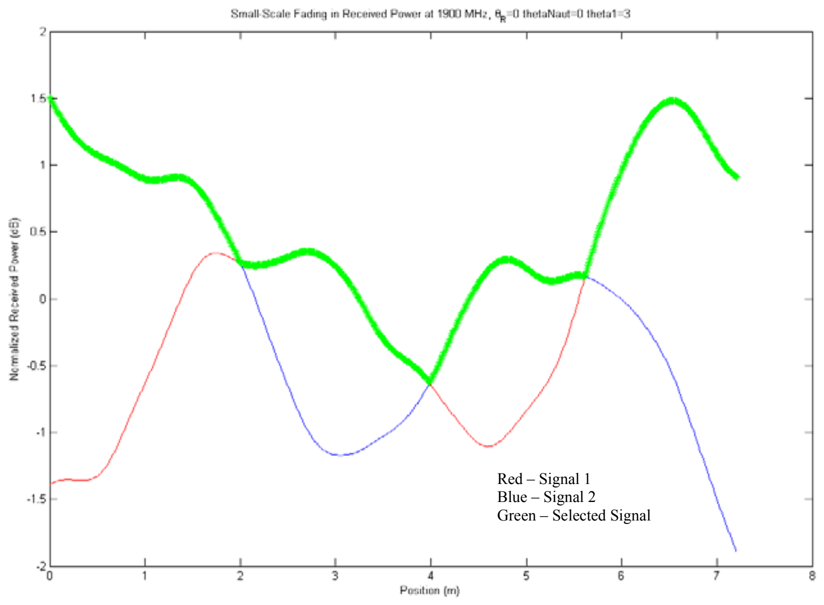


Figure 3. Small Scale Fading Received Power for Model 1

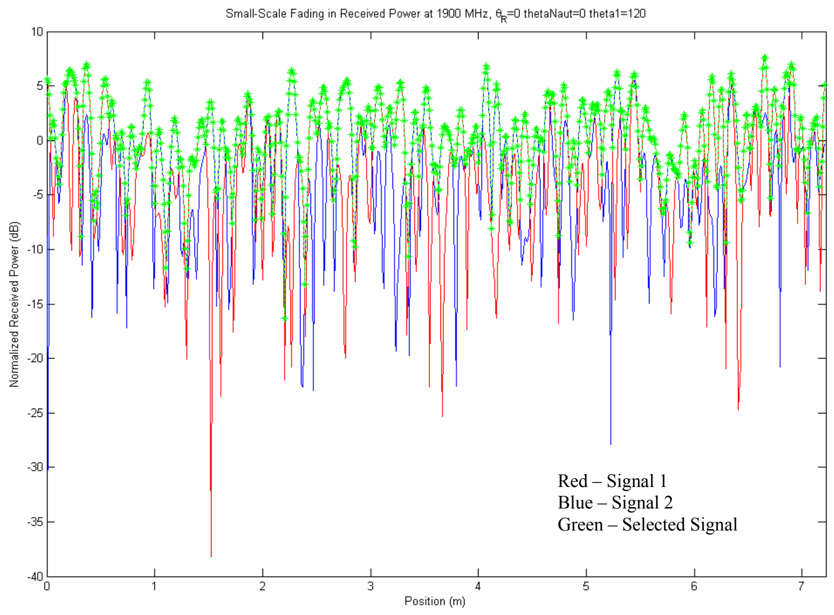


Figure 4. Small Scale Fading Received Power for Model 2

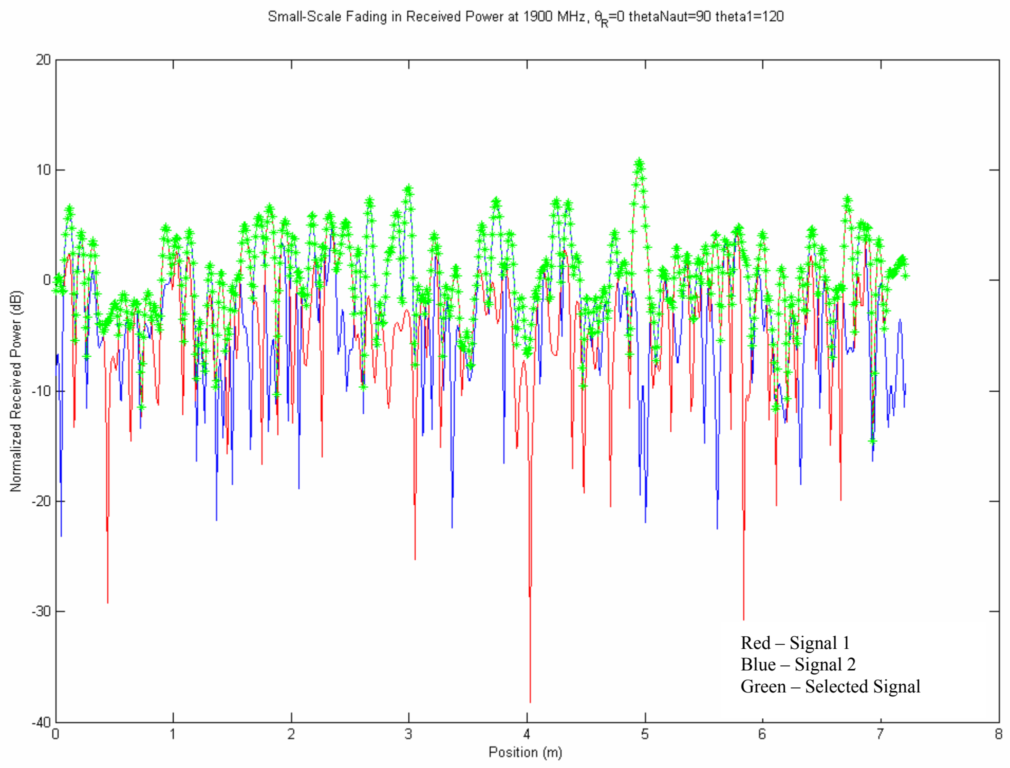


Figure 5. Small Scale Fading Received Power for Model 3

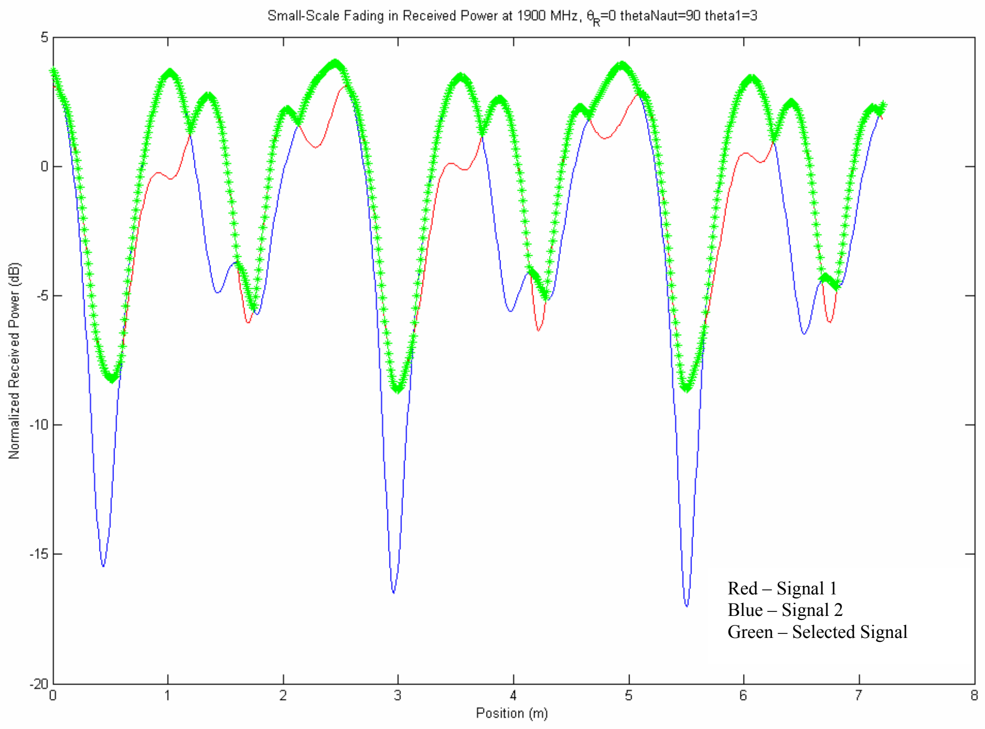


Figure 6. Small Scale Fading Received Power for Model 4

IV. Conclusions

From inspection the simulated small scale fading signals indicate that the worst case reception will occur in the urban environment when the train is moving inline with the base station. Under these conditions the level crossing rate is the highest of the possible conditions and the threshold selection diversity output supplies the lowest signal power values.

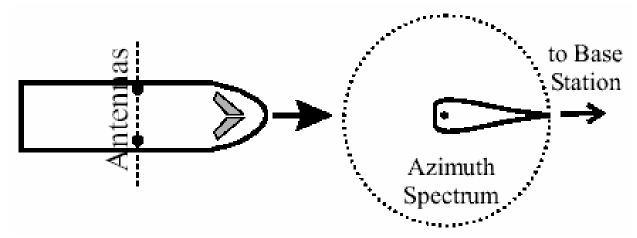
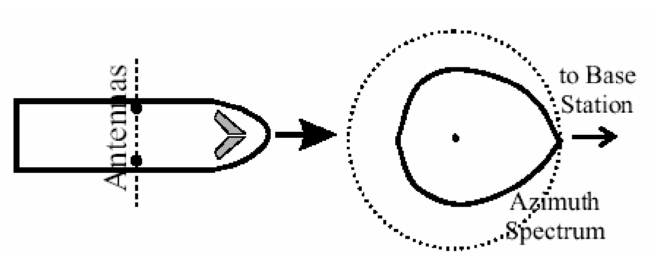
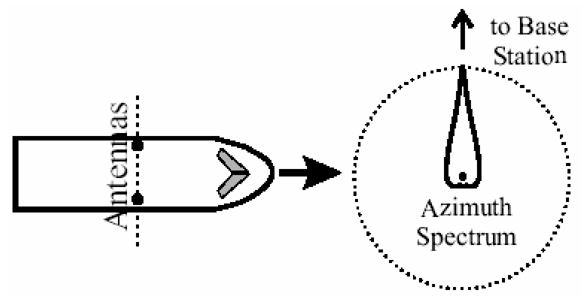
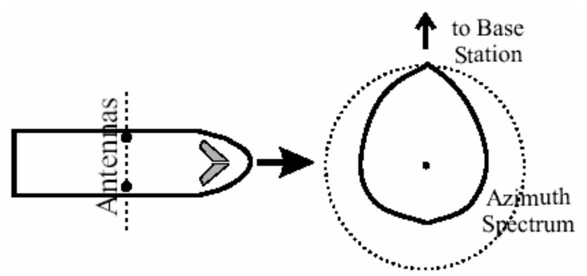
The information obtained visually through the small signal graphs agrees with and confirms the information obtained using the shape factors. In order to switch input signals the selection diversity algorithm must sample the power from the two signals faster than the highest level crossing rates. The maximum level crossing rate from Figure 1 is 51.7 dB which converts to 385 samples per second. The maximum level crossing rate is a minimum power sampling rate and in order to reduce delay in switching a power sampling rate of roughly 10 times the minimum value should provide for ample sampling speed. Therefore the recommended power sampling frequency is 4 kilohertz.

References

[1] G. D. Durgin and T. S. Rappaport, "Theory of Multipath Shape Factors for Small-Scale Fading Wireless Channels," in *IEEE* Trans. Antennas Propagat., vol. 48. May 2000.

Bradley J. Schafer was born in Kenosha, WI on April 22 1978. He received a triple degree in Mechanical Engineering, Electrical Engineering and underwater basket weaving from the Georgia Institute of Technology. Brad has been quoted saying, 'Laugh now, but this basket weaving stuff is going to pay off big time… . . . . big time.

Appendix 1: System Model Parameters

Condition 1: ThetaNaut =0 Rural Landscape Theta1=3 degrees 	**Condition 2**: ThetaNaut =0 Urban Landscape Theta1=120 degrees 
Condition 3: ThetaNaut =90 Rural Landscape Theta1=3 degrees 	**Condition 4**: ThetaNaut =90 Urban Landscape Theta1=120 degrees 

Appendix2: Calculation of the Complex Courier Coefficients

```
% FUNCTION:  function [Fn] = FourCoeff( theta1_deg, thetaNaut_deg, n)
%
% Calculates the Fourier coefficient
%
% Author:  Brad Schafer, 3/21/2004
%
% Inputs:
%  theta1_deg - angle of arrival from base station. Azimuth direction of
%              peak arrival
%  thetaNaut_deg - Thickness of distribution
%  n              nth coefficient
%
% Outputs:
%  Fn - value of Fourier Coefficient
%
% Sample Usage:
% To calculate the n=0 fourier coefficent of a spectrum with a direction of arrival at 90
%   degrees and a theta1 value of 120 degrees...
% >> [Fn] = FourCoeff( 120, 90,0);


function [Fn] = FourCoeff( theta1_deg, thetaNaut_deg, n)
    theta1 = theta1_deg/180*pi;
    thetaNaut = thetaNaut_deg/180*pi;
    A = 1;
    Fn = (2 * A * theta1 * exp(j * n * thetaNaut) ) / (n^2 * theta1^2 + 1) *  (1-(-1)^n *
theta1 * exp(-pi/theta1));
return ;
```

Appendix3: Calculation of Shape Factors

```
% FUNCTION:  function [angS] = angSpread( Fzero, Fone)
%
% Calculates the Angular Spread
%
% Author:  Brad Schafer, 3/21/2004
%
% Inputs:
%   Fzero - first fourier coefficient
%   Fone -  second fourier coefficient
%
% Outputs:
%  angS - angular spread. Should be a number between 0 and 1
%
% Sample Usage:
% To calculate the angluar spread with a couple sample courier
% >> function [angS] = angSpread( 0.1047,0.1044)


function [angS] = angSpread( Fzero, Fone)
   angS = sqrt(1-((abs(Fone))^2) / (Fzero^2))
return ;
```

Appendix4: Calculation of Lever Crossing Rates

```
%function [p,LCR_array] = levelCrossingRate( angSprd, angConst, maxFade, theta1,
thetaNaut)

hold on
[p,y]=levelCrossingRate( .0744, .9682, 0, 3, 90);
plot(p,20*log10(y),'-r')
%plot(p,y)
[p,y]=levelCrossingRate( .0744, .9682, 90, 3, 0);
plot(p,20*log10(y),'-c')
%plot(p,y)
[p,y]=levelCrossingRate( .8593, .2812, 90, 120, 0);
max(y)
plot(p,20*log10(y),'-g')
%plot(p,y)
[p,y]=levelCrossingRate( .8593, .2812, 0, 120, 90);
plot(p,20*log10(y),'-b')
%plot(p,y)

hold off
```

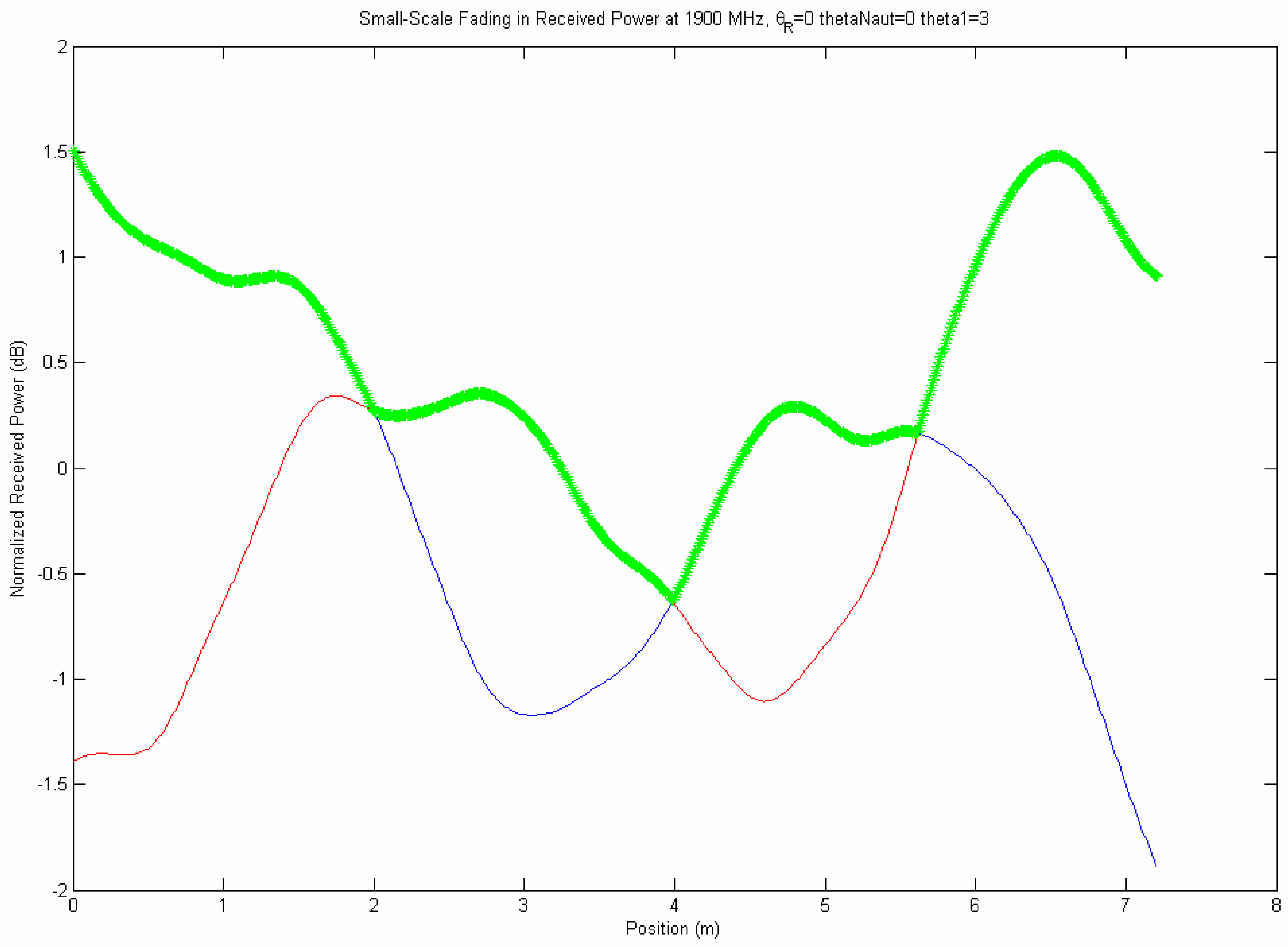
```
function [p,LCR_array] = levelCrossingRate( angSprd, angConst, maxFade, theta1,
thetaNaut)
   v = 72.22;  %speed of train
   maxFade_rad = maxFade/180*pi;
%    approachAng_rad = approachAng/180*pi;
   f = 1900e6;
   wavelength = 3e8 / f;
   %[output,theta] = Azimuth( theta1, thetaNaut);
   %a1 = polar(theta,output);
   %pTheta = output;
   p = .02:.02:2.5;

   LCR_array = zeros(1,length(p));
   for n = 1:length(p)
      LCR1 = sqrt(2*pi) * v * angSprd * p(n) / wavelength;
      LCR2 = sqrt( 1 + angConst * cos(2*(0-maxFade)));
      LCR3 = exp( -(p(n))^2 );
      LCR_array(n) = LCR1 * LCR2 * LCR3;
   end

   %pLCR = plot(pTheta, LCR)
return
```

Appendix5: Calculation of Average Fade Duration

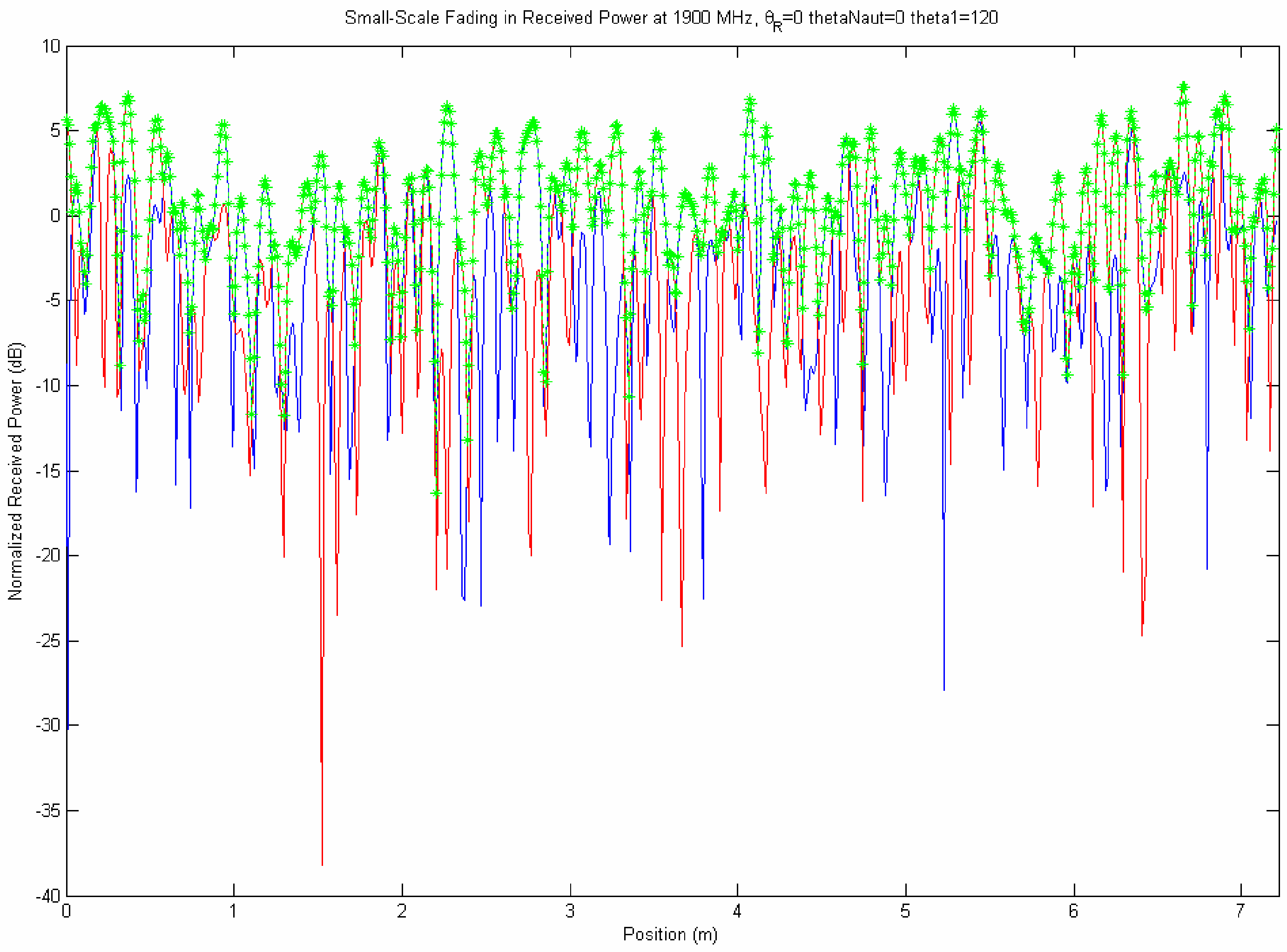
```
function [p,FD_array] = fadeDuration( angSprd, angConst, maxFade, theta1, thetaNaut)
   v = 72.22;  %speed of train
   maxFade_rad = maxFade/180*pi;
%    approachAng_rad = approachAng/180*pi;
   f = 1900e6;
   wavelength = 3e8 / f;
   %[output,theta] = Azimuth( theta1, thetaNaut);
   %a1 = polar(theta,output);
   %pTheta = output;

   p = .02:.02:2.5;

   FD_array = zeros(1,length(p));
   for n = 1:length(p)
      FD1 = wavelength * ( exp( (p(n))^2 ) - 1 );
      FD2 = sqrt(2*pi)*v*p(n)*angSprd;
      FD3 = sqrt(1 +angConst*cos(2*(0-maxFade)));
      FD_array(n) = FD1 / FD2 / FD3;
   end
```

```
%function [p,LCR_array] = levelCrossingRate( angSprd, angConst, maxFade, theta1,
thetaNaut)

hold on
[p,y]=fadeDuration( .0744, .9682, 0, 3, 90);
plot(p,20*log10(y),'-r')
title('Fade Duration vs. Normalized Threshold Level')

%plot(p,y)
[p,y]=fadeDuration( .0744, .9682, 90, 3, 0);
plot(p,20*log10(y),'-c')
xlabel('Normalized Threshold Level')
ylabel('Average Fade Duration [sec] dB')
%plot(p,y)
[p,y]=fadeDuration( .8593, .2812, 90, 120, 0);
plot(p,20*log10(y),'-g')
%plot(p,y)
[p,y]=fadeDuration( .8593, .2812, 0, 120, 90);
plot(p,20*log10(y),'-b')
%plot(p,y)

hold off
```

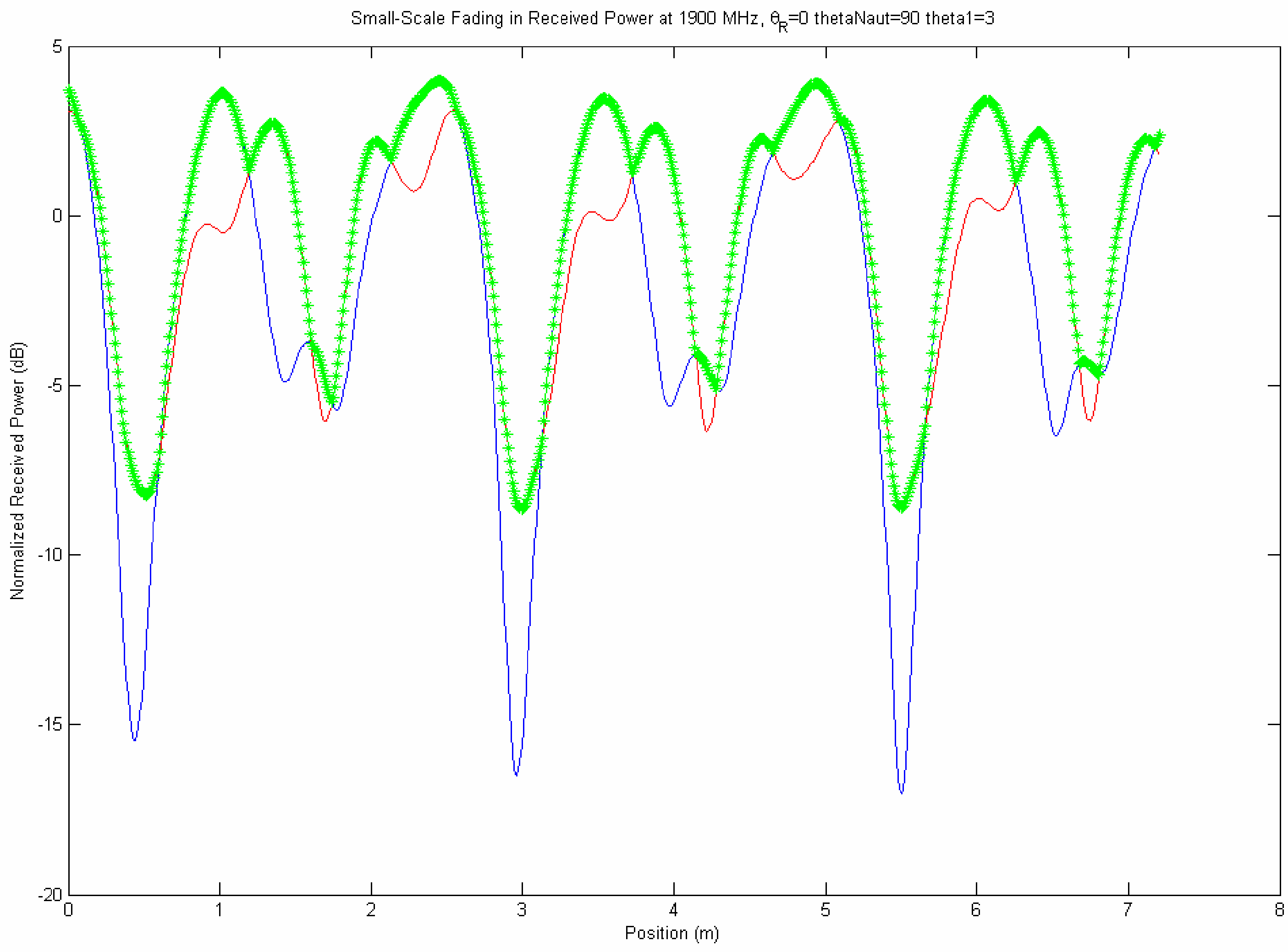
Appendix 6a. Small Scale Fading Received Power for Model 1.



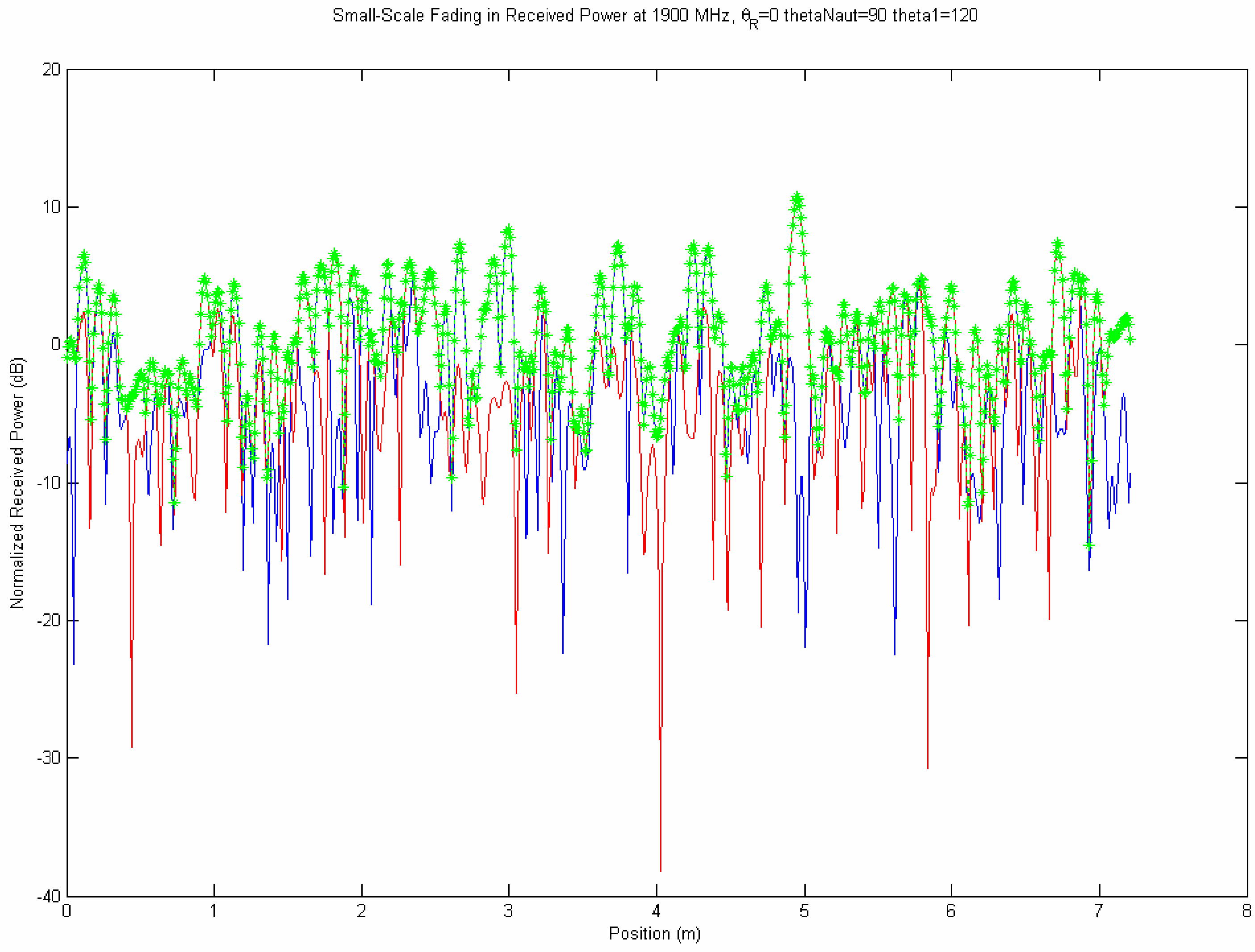
Appendix 6b. Small Scale Fading Received Power for Model 2.



Appendix 6c. Small Scale Fading Received Power for Model 3.



Appendix 6d. Small Scale Fading Received Power for Model 4.



Appendix7: Small Scale Fading Received Power Matlab Code

```
% FUNCTION:  [Ez,r] = MakeWaves( L, f, thetaR, thetaNaut, theta1, N )
%
% Makes sample fading plots as a function of space.
%
% Author:  Greg Durgin, 3/13/2004
%          modified by Brad Schafer 4/21/04
%
% Inputs:
%  L - the total distance (in meters) to plot
%  f - carrier frequency (in Hz)
%  thetaR - azimuth orientation of movement in space (in degrees)
%  thetaNaut - angle of signal approach
%  theta1 - angle associated with thickness of received beam

%  N - total number of plane waves (default is 100)
%
% Outputs:
%  Ez - vector of total z-component of E-field as function of space
%  r - vector of corresponding positions (same size as Ez)
%
% Sample Usage:
% To make a fading profile over 3 meters in space at a carrier frequency of
% 2.0 GHz as observed with a 45-degree orientation in space:
% >> [Ez,r] = MakeWaves( 3, 2.4e9, 45 );
%
% Sample Usage New:
% To make a fading profile over 3 meters in space at a carrier frequency of
% 2.0 GHz as observed with a 45-degree orientation in space:
% >> [Ez,r] = MakeWaves4( 3, 2.4e9, 45, 0, 120 );

% Notes:
% The code provides a useful example of how to simulate the small-scale
% fading that occurs when many plane waves superimpose in space.  Feel free
% to adapt this code for your ECE3065 project.  The output of this function
% is the z-component of electric field (in complex phasor form) as a
% function of space.  The function automatically graphs Ez as a function of
% r on a dB-scale graph at the end of the routine, so that you can see the
% types of outputs required for the project.
%
% The function "p" at the end of this file is your azimuth spectrum of
% multipath power.  The default spectrum is omnidirectional (so the graphs
% will not depend on thetaR at all!) but this can be changed fairly
% quickly.
%Cont.
```

```
function [Ez,r] = MakeWaves4( L, f, thetaR, thetaNaut, theta1, N )

% Set a default value of 100 for N
if ~exist('N')
   N = 100;
end;

% Initialize Variables
lambda = 3e8/f;                   % wavelength
M = 16*L/lambda;                    % good number of space samples
k = 2*pi/lambda;                   % freespace wavenumber
%r = linspace(0,L*(1-1/M),M);        % make an array of sequential positions
x = linspace(0,L*(1-1/M),M);        % make an array of sequential positions
Ez = zeros(size(x));                % initialize Ez component to zero
Ez2 = zeros(size(x));                % initialize Ez component to zero
theta = linspace(0,2*pi*(1-1/N),N); % vector azimuth angles
thetaR_rad = thetaR/180*pi;
thetaNaut_rad = thetaNaut/180*pi;
theta1_rad = theta1/180*pi;

% Step through N azimuth angles and add z-components of electric
% field from each direction.  The expression below that is added to Ez each
% loop is a single plane wave.  Its amplitude is related to the multipath
% angle spectrum and its phase is random, but tapers "k" radians/meter in
% the direction of wave travel.  By the end of this loop, Ez will be the
% sum of N different plane waves, creating the strange
% constructive-destructive interference pattern as a function of space.

for n=theta,
   %for first one y's are all zeros
   Ez = Ez + (p(n,thetaNaut_rad,theta1_rad))^.5 *   exp( j*( 2*pi*rand(1) + k*x*cos(n-
thetaR_rad)                          ) );
   %for second attenna y is -2
   Ez2 = Ez2 + (p(n,thetaNaut_rad,theta1_rad))^.5 * exp( j*( 2*pi*rand(1) + k*x*cos(n-
thetaR_rad) - k*2*sin(n-thetaR_rad) ) );
end;

% Normalize Ez so that average power is 1
Erms = (sum(abs(Ez).^2)/length(Ez))^.5;     % root-mean-square
Ez = Ez/Erms;                             % normalize to average power 1

Erms2 = (sum(abs(Ez2).^2)/length(Ez2))^.5;     % root-mean-square
Ez2 = Ez2/Erms2;                             % normalize to average power 1

% Plot a nice graph of power in dB as a function of space
figure(1);
```

```
highest = max(Ez,Ez2);
plot(x,20*log10(abs(Ez)),'-b',x,20*log10(abs(Ez2)),'r-',x,20*log10(abs(highest)),'g*:');
% plot dB-scale Ez as a function of r
title(sprintf('Small-Scale Fading in Received Power at %1.0f MHz,
\\theta_R=%i',f/1e6,thetaR*180/pi));
xlabel( 'Position (m)' );
ylabel( 'Normalized Received Power (dB)' );

return; % finish


% Here is the function defining the azimuth spectrum, the distribution of
% multipath power as a function of angle-of-arrival.
%
% The example below is for an omnidirectional spectrum (a constant).
% Change it to see other types of angle spectra.
%
function P = p(theta_p,thetaNaut_p,theta1_p)
   P = 1 * exp(-abs((theta_p-thetaNaut_p)/theta1_p));
% Here is another example, the cosine-squared spectrum
% P = cos(theta)^2;
return;
```