Linnæus University
School of Computer Science, Physics and Mathematics

Degree project

A STUDY OF MULTIPATH WAVE PROPAGATION USING NERO2D AND FFT

Master of Science in Electrical Engineering with Specialization in
Signal Processing & Wave Propagation.
Supervisor: Sven-Erik Sandström

Author: Mayss Al-qaissi
Date: 2014-04-09
Subject: Electrical Engineering
Level: Master
Course code: 5ED06E

1. Abstract

In this report, the Fast Fourier Transform is described briefly. An implementation, in the form of the Fortran code four1, is tested to verify the accuracy. A two-ray model for wave propagation above a flat earth is discussed. The case with AM modulation is implemented in a Mathematica script. Calculations of the surface current density, with the program NERO, are made to test the accuracy. The transient scattering from a PEC cylinder is studied by means of the code run_nero that runs NERO repeatedly. From a spectrum calculated in this way, the impulse response is obtained by Fourier inversion.

Keywords: FFT, Radio channel, Impulse response, Fast multipole method

2. Acknowledgement

The author would like to acknowledge the supervisor Sven-Erik Sandström for his many helpful comments that led to a substantial improvement of this report. A special thanks also to my family for their unfailing support during the period of study and development of this thesis and my colleagues in the Electrical Engineering program for their motivations.

Contents

LIST OF FIGURES

3. Introduction

A very large class of important problems falls under the heading of Fourier transform methods or spectral methods. The Fourier transform and its discrete versions are efficient computational tools.

The discrete Fourier transform is the estimation of a function based on a finite number of equidistant sample points. The fast Fourier transform (FFT) is an algorithm to compute the discrete Fourier transform and its inverse.

In order to maintain reliable radio communication, it is of interest to be able to predict the performance of the radio channel based on geographical data such as topography and the properties of the ground/water. One way of doing this is to calculate the impulse response for a channel with multi-path propagation and a typical modulation. The simplest case with a two-ray model and AM modulation is coded in a Mathematica script.

A test case is that of a plane wave pulse that impinges on a perfectly conducting cylinder. The numerical solution of the integral equation, for the surface current, is obtained with the code NERO. This approach is extended to transient scattering by means of the FFT in order to compute the impulse response for the cylinder. This is a test case and a preparation for the computation of the impulse response for a realistic radio channel.

4. A BRIEF REVIEW OF THE FFT

The FFT relates to two domains, the time domain and a function h(t), and the frequency domain and a function H(f), that are linked by the relations,

$$H(f) = \int_{-\infty}^{\infty} h(t) e^{-2\pi i f t} dt \tag{1}$$

$$h(t) = \int_{-\infty}^{\infty} H(f) e^{2\pi i f t} df \tag{2}$$

In applications one may prefer to use the angular frequency $\omega = 2\pi f$:

$$H(\omega) = \int_{-\infty}^{\infty} h(t) e^{-i\omega t} dt \tag{3}$$

$$h(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} H(\omega) e^{i\omega t} d\omega \tag{4}$$

The function $h(t)$ is sampled with evenly spaced intervals Δ in time [1]. The intervals between the points define the *sampling rate*.
Suppose that we have N consecutive samples, with a sampling rate $1/\Delta$, and want to estimate the Fourier transform of h(t) based on these samples:

$$h_k = h(t_k), \qquad t_k = k\Delta, \qquad k = 0, 1, 2, ..., N-1 \tag{5}$$

The discrete Fourier transform of h_k for N points is denoted by H_n

$$H_n = \sum_{k=0}^{N-1} h_k e^{2\pi i k n/N} \tag{6}$$

With the complex number $W = e^{2\pi i/N}$ one obtains,

$$H_n = \sum_{k=0}^{N-1} W^{nk} h_k \tag{7}$$

The samples h_k are multiplied by powers of W in order to produce the H_n. This matrix multiplication requires N^2 complex multiplications, plus a smaller number of operations to generate the required powers of W.
The discrete Fourier transform can be computed in $N \log_2 N$ operations with an algorithm called *The Fast Fourier Transform* or the FFT [1].
The subroutine **four1** is a Fortran code to compute the FFT which is written by N.M. Brenner. The input is an array containing the samples h_k or H_n and a parameter that

specifies if it is the transform or the inverse that is to be computed. There are nn complex data points stored in the real array data. The parameter isign is set to either $+1$ or -1. When isign is set to -1, the routine calculates the inverse transform. The integer nn is the number of complex data points [1]. The actual length of the real array (data) is 2 times nn, with real and imaginary parts occupying consecutive locations.

The real and imaginary parts of the zero frequency component F_0 are in data(1) and data(2); the smallest nonzero positive frequency has real and imaginary parts in data(3) and data(4). In this manner the first nn positions in data are filled with the nonnegative part of the spectrum. The negative part is then stored in the remaining nn positions of data, in reverse order, so that the smallest negative frequency occupies data(nn+1) and data(nn+2). The largest negative frequency then occupies data(2nn-1) and data(2nn).

I have used the routine **four1** to compute the spectrum of a time function, and the time function from its spectrum. I did some simple tests for the pulse, the sine function and the Gaussian pulse that have known transforms.

The first example was a simple pulse function specified by the constant c:

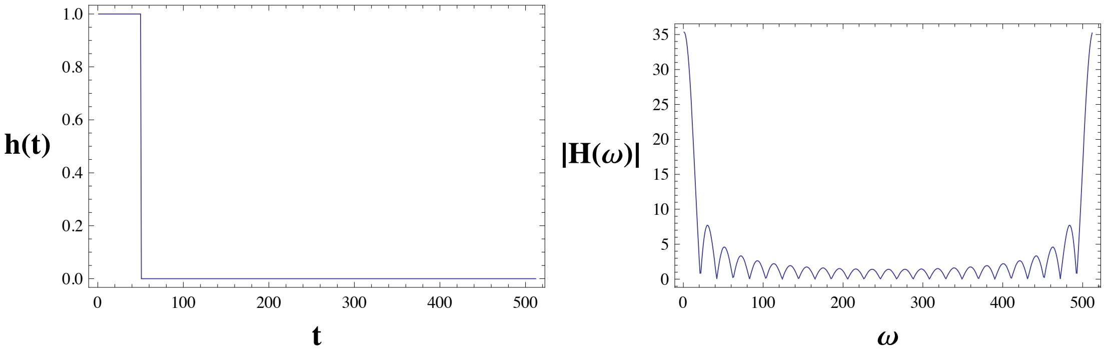
$$h(t) = \begin{cases} 1 & \text{for} \quad |t| \leq c \\ 0 & \text{elsewhere} \end{cases} \tag{8}$$

From Eq. 3 one obtains,

$$H(\omega) = \frac{1}{2\pi} \int_{-c}^{c} e^{i\omega t} dt \tag{9}$$

$$H(\omega) = \frac{c}{\pi} \frac{\sin(\omega c)}{\omega c} \tag{10}$$

So the Fourier transform of a pulse function is a sinc function. Now by using the subroutine **four1** we should get the same result. With nn= 256 one obtains Fig. 1.

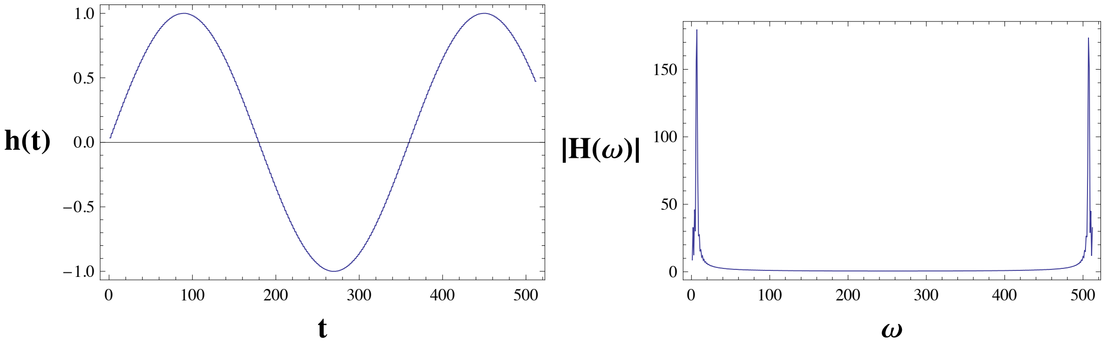


(A) A pulse function in the time domain.

(B) The coresponding Fourier transform in the frequency domain.

FIGURE 1. A Fourier pair with the conventional arrangement of the spectrum.

The second example deals with the sine function and by repeating the same procedure, one obtains a complex spectrum.

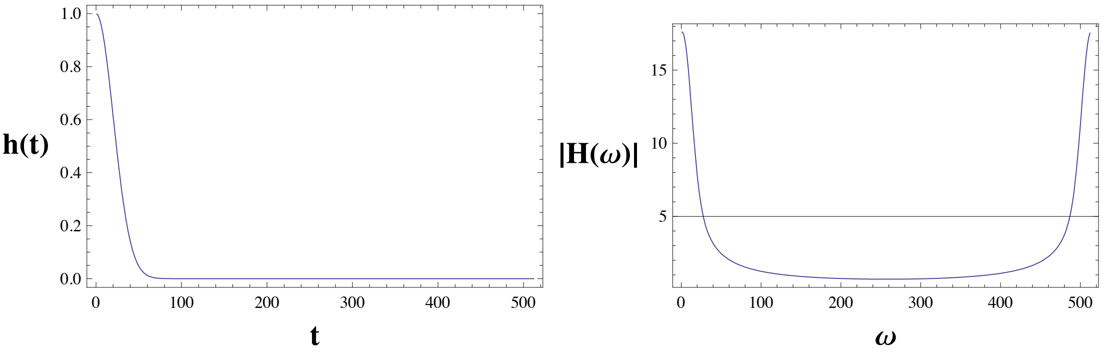


(A) A sine function in the time domain.

(B) The spectrum of the sine function.

FIGURE 2. A Fourier pair for the sine function.

Another example is the e^{-t^2} functions shown in Fig. 3.

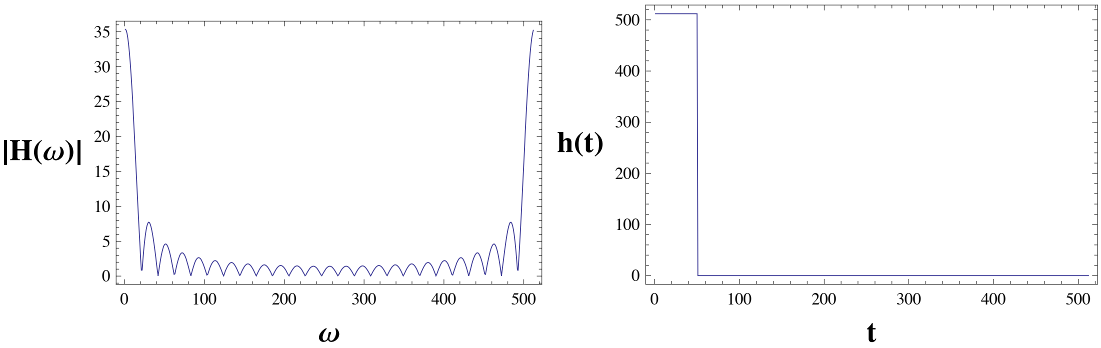


(A) e^{-t^2} in the time domain.

(B) The Fourier transform of e^{-t^2} in the frequency domain.

FIGURE 3. A Fourier pair for e^{-t^2}.

In order to obtain the inverse of the result in Fig. 1 one calls the routine four1, with isign = -1, and obtains Fig. 4:

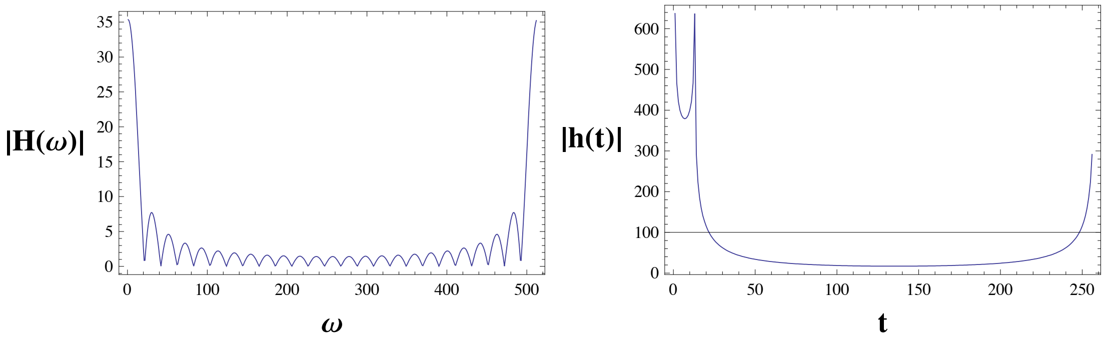


(A) The spectrum of the pulse function.

(B) The inverse of $H(\omega)$.

FIGURE 4. The inverse transform, applied to the spectrum of the sinc function, with 512 samples.

In order to reproduce the original function, a sufficient number of samples is needed, as illustrated by Fig. 4. The effect of reducing the number of samples is shown in Fig. 5.



(A) The spectrum of the pulse function.

(B) The inverse of $H(\omega)$.

FIGURE 5. The inverse transform for the sinc function computed with 256 samples .

It is clear that the pulse function is reproduced with 512 samples and also that 256 samples is insufficient.

5. OVERVIEW OF THE PROGRAM RUN_NERO

One can describe a full wave electromagnetic simulator in terms of pre-processing, processing and post-processing [2]. run_nero is a program which is written in fortran and uses the subrourine **four1**. In the pre-processing part one specifies the geometry, the excitation, the operational frequency and the output [2]. By using the graphics toolbox **wxGBTool** which is matched to the program **NERO** one can specify the geometry using predefined shapes such that circles or polygons. There is also a possibility to define geometries by introducing a finite number of points in 2D Cartesian coordinates in a counterclockwise fashion [2]. One can specify objects as PEC (perfect electrical conductor) bodies or as permeable objects with complex permittivity and permeability. In **NERO** one has three source types: plane waves, point sources or gaussian beams. Plane waves can be either TM or TE in relation to the object. It is possible to specify a number of sources (an array) for a given geometry. The operating frequency and the complex permittivity and permeability of the surrounding medium are also given [2]. In **wxGBTool** one also specifies the output from the **NERO** solver; each output type contains many parameters [3]:

1. Line output: has three parameters, point1 (x;y) as starting point of the line, point2 (x;y) as the end point of the line and the number of output points along the line.
2. Circle output: has also three parameters, the centre of the circle (x;y), the radius of the circle (R) and the number of points on the circle.
3. Bitmap output: has three parameters, lower left point of the bitmap (x;y), upper right point of the bitmap (x;y) and the resolution X which is the number of the equidistant points along the x-direction [3].

After specifying the input geometry, the excitation method and the choice of the output type, one creates two files with **wxGBTool**. The first one is the input geometry file (.igf) which contains the geometrical layout of the scene or the tested objects, the illumination and the required output. The bitmap file created by **wxGBTool** has the extension (.bdf) [3]. The next step is to run the program run_nero by using a file (test.igf) to run nero2d repeatedly for a set of frequencies in order to obtain a doublesided spectrum. This spectrum can then be inverted by means of four1. The results are plotted with Mathematica scripts.

6. Two-ray model for propagation with AM modulation above a flat earth

The two ray model has two antennas above ground and there is a reflected ray from the ground. The two ray model assumes that the transmitted wave reaches the (non-moving) receiver directly through a line-of-site path, and indirectly by perfect reflection from a flat ground surface [4]. If the links are short we may neglect the earth's curvature, so the figure below portrays the geometry involved: the transmitting antenna, located at the base station, is shown radiating from a height h_t above a perfectly reflecting, flat ground surface. The receiving antenna, a free-space distance d [m] away, is shown situated at a height h_r above the ground [4].

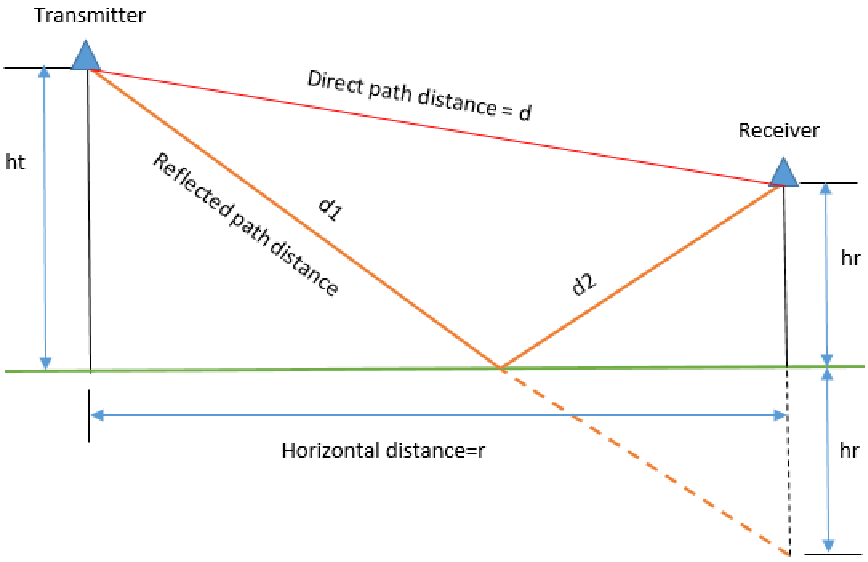


Figure 6. Two ray propagation model.

From the figure above, it is clear that:

$$d_1 + d_2 = \sqrt{r^2 + (h_t + h_r)^2} \tag{11}$$

$$d^2 = r^2 + (h_t - h_r)^2 \tag{12}$$

Then we can rewrite these two equations as the following:

$$d_1 + d_2 = \sqrt{d^2 + 4h_t h_r} \tag{13}$$

$$d_1 + d_2 = d\sqrt{1 + \frac{4h_t h_r}{d^2}} \tag{14}$$

Then since $h_r h_t \ll d^2$ and by using approximations we obtain:

$$d_1 + d_2 = d + \frac{2h_t h_r}{d} \tag{15}$$

$$\Delta d = d_1 + d_2 - d = \frac{2h_t h_r}{d} \tag{16}$$

Where Δd is the difference between the direct and indirect rays. The typical electric field appearing at the transmitter is a far-field sinewave at frequency f_c with amplitude E_T. In complex notation it is written in the usual form:

$$\tilde{E} = E_T e^{j\omega_c t} \tag{17}$$

Now by considering the direct wave impinging on the receiving antenna with complex form is given by:

$$\tilde{E}_{R,D} = E_T e^{j\omega_c (t - \frac{d}{c})} \tag{18}$$

Where c is the velocity of the light. The indirect wave, assuming perfect reflection at the ground, appears in a similar form, except that its total distance traveled is $d_1 + d_2$, while with perfect reflection, it undergoes an added π radians phase change[4]. It is thus written in complex form as:

$$\tilde{E}_{R,I} = -E_T e^{j\omega_c \left(t - \frac{d_1 + d_2}{c}\right)} \tag{19}$$

The total received field is the sum of direct and indirect field:

$$\tilde{E}_R = E_T e^{j\omega_c (t - \frac{d}{c})} \left[1 - e^{-j\omega_c \left(\frac{d_1 + d_2 - d}{c}\right)}\right] \tag{20}$$

$$\tilde{E}_R = E_T e^{j\omega_c \left(t - \frac{d}{c}\right)} \left[1 - e^{-j\omega_c \left(\frac{\Delta d}{c}\right)}\right] \tag{21}$$

The received indirect wave is similar to the direct wave but with a phase shift and a time shift $\frac{\Delta d}{c}$.

These formulas are coded in a Mathematica script (Appendix B). Since one is interested in the impulse response, one uses the input signal in Figure 7 as an approximate impulse:

$$x(t) = \frac{e^{-\left(\frac{t}{a}\right)^2}}{a\sqrt{\pi}} \tag{22}$$

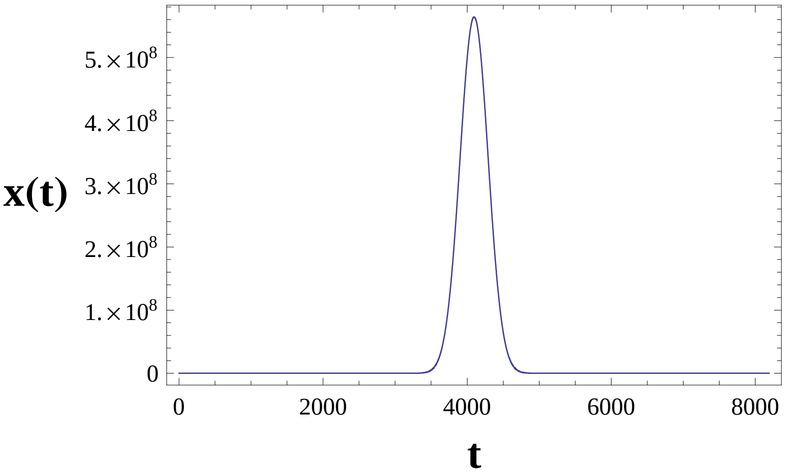


FIGURE 7. The input signal x(t).

To carry out the amplitude modulation for the input signal, one could multiply the input signal with the carrier signal $\cos(\omega_c t)$,

$$x_c(t) = x(t)\cos(\omega_c t), \tag{23}$$

as shown in Fig. 8.

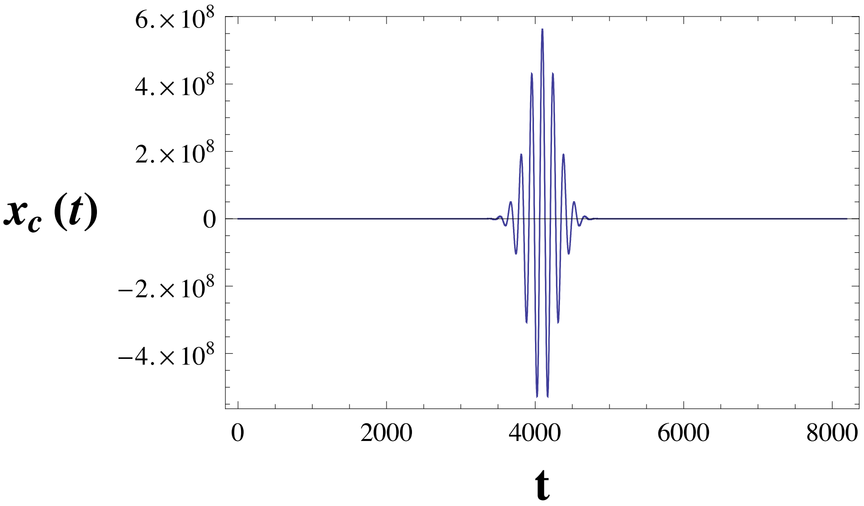


FIGURE 8. The amplitude modulated signal $x_c(t)$.

The reflected signal from the ground/water can be represented by the same input signal but with a time lag $\frac{\Delta d}{c}$,

$$x_d = x(t - \frac{\Delta d}{c}) = \frac{e^{-\left(\frac{t-\frac{\Delta d}{c}}{a}\right)^2}}{a\sqrt{\pi}} \tag{24}$$

The Fourier transform of the sum of the direct signal x(t) and the reflected signal, multipied with a reflection factor Γ, is shown in Fig. 9. Two sidebands appear in Fig. 9. X_s is given by Eq. 25.

$$X_s = F\left[x(t) + \Gamma x_d(t)\right] \tag{25}$$

A typical value of Γ is -0.2. The spectrum is shown without the reordering used in Fig. 1.

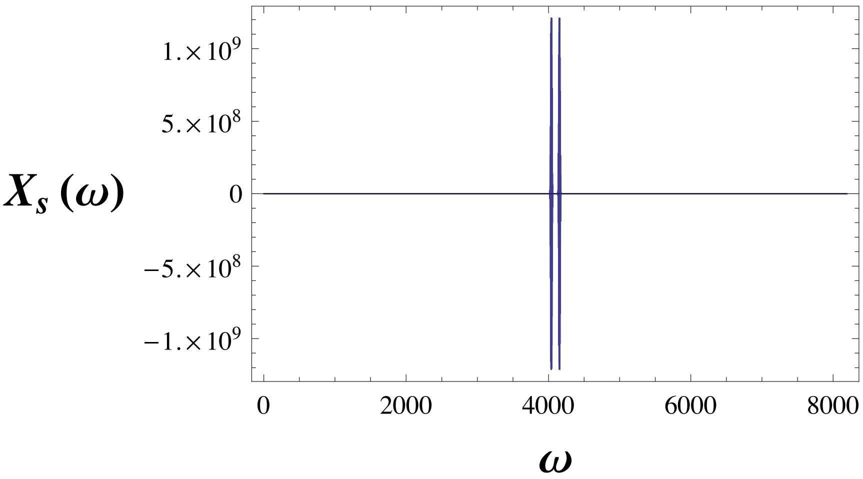


FIGURE 9. $X_s(\omega)$ is the Fourier transform of the sum of the input signal $x(t)$ and the reflected signal $x_d(t)$.

The function $h(t) = t\,e^{\frac{-t}{\tau}}$ is the assumed impulse response for the channel and its Fourier transform $H(\omega)$ is the transfer function of the channel.

The output signal in the frequency domain is $Y_s(\omega)$ and the spectrum is shown in Fig. 10. The spectrum now has three main contributions.

$$Y_s(\omega) = H(\omega)\,X_s(\omega) \tag{26}$$

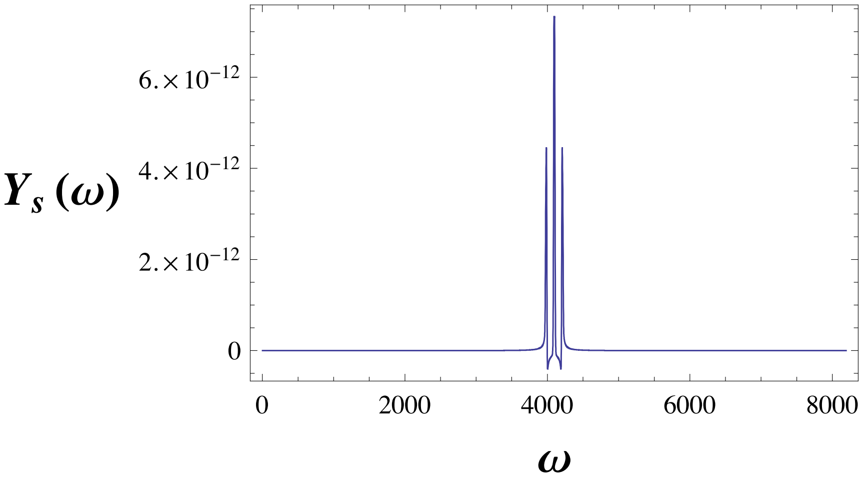


FIGURE 10. The output spectrum $Y_s(\omega)$ at the receiver.

Coherent demodulation is obtained by means of multiplication with $cos(\omega_c t)$ in the time domain. In order to extract the baseband, the signal is transformed and LP-filtered. The spectrum is multiplied by a window function that extracts the LF-part. A very narrow band appears in Fig. 11.

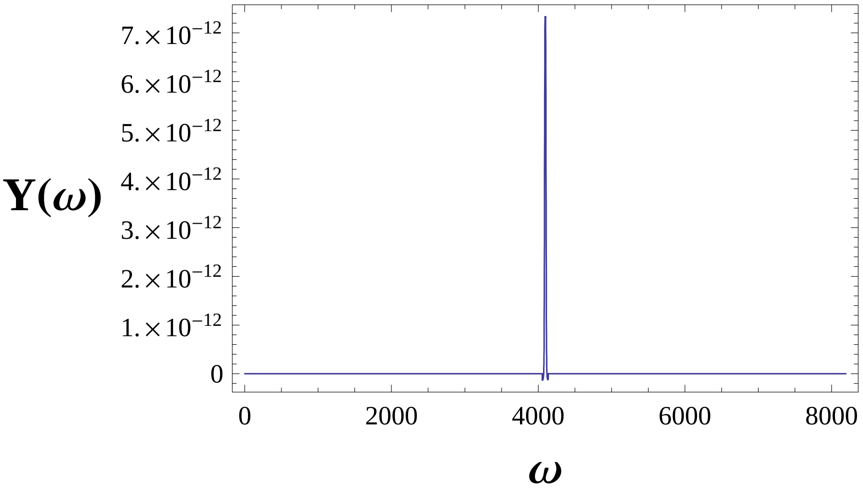


FIGURE 11. The output spectrum after using a LP-filter.

Finally, the output signal can be obtained by taking the inverse Fourier transform of the output spectrum after using a LP-filter. This signal represents the output signal at the receiving antenna and differs from the input signal mainly because of the ground reflection.

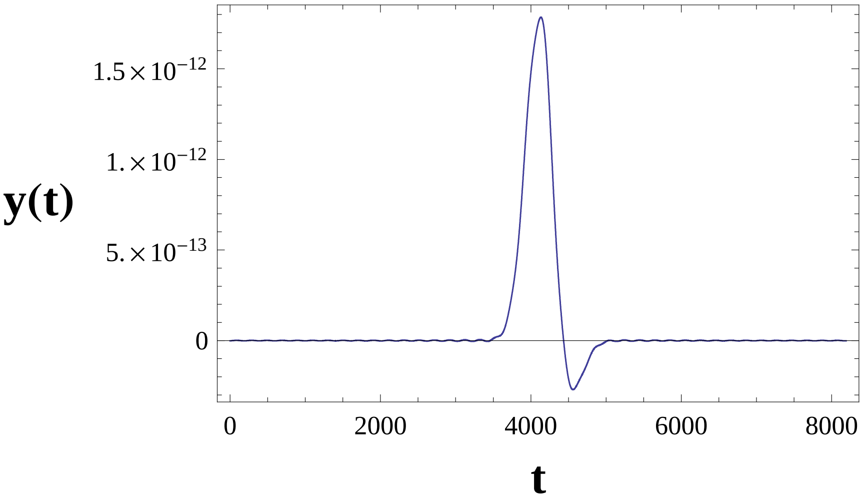


FIGURE 12. The output signal detected at the receiving antenna.

The simple two-ray model for a radio channel is coded in a Mathematica script (see appendix B).

7. Plane wave incident on a cylinder

In order to verify the accuracy of NERO, the computed surface current could be compared to that obtained with the series solutions for the TM and TE cases.

$$K_z = \frac{2}{\pi x \eta} \sum_{m=0}^{\infty} \frac{\cos m\theta}{H_m(x)} 2\, e^{im\pi/2} \tag{27}$$

$$K_\theta = \frac{2i}{\pi x} \sum_{m=0}^{\infty} \frac{\cos m\theta}{H'_m(x)} 2\, e^{im\pi/2} \tag{28}$$

Here, x=ka, with the radius a=1 and the wavenumber k. η is the free space impedance. The series solution for the TM and TE case at 1 GHz for a PEC cylinder, produce the surface currents in Fig. 13.

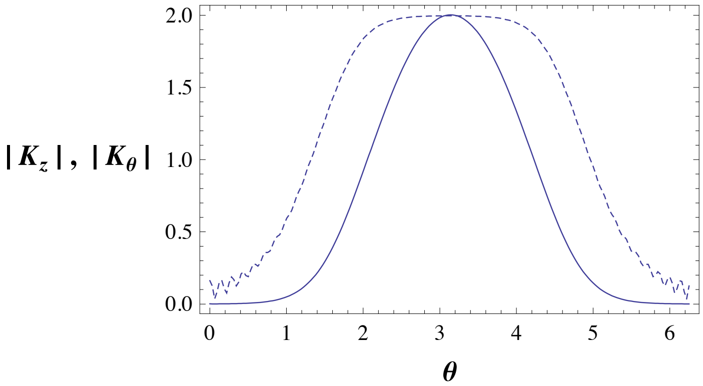
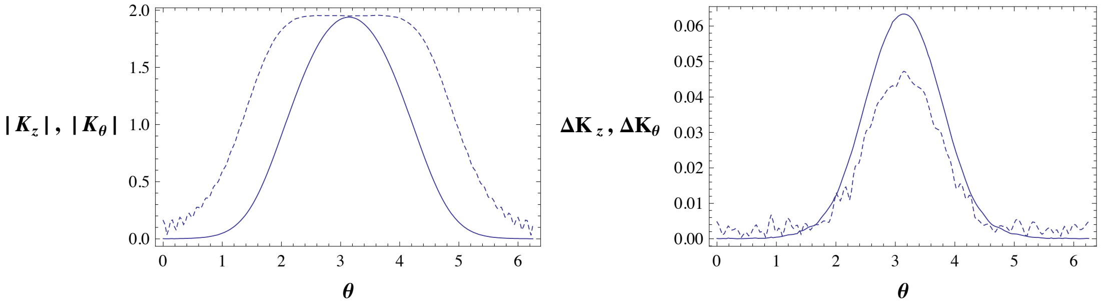


FIGURE 13. $|K_z|$ in solid line (TM) and $|K_\theta|$ in dashed line (TE), f=1 GHz.

As mentioned in Section 5, NERO is linked to a graphical tool and one can select a cylinder with unit radius and plane wave incidence. In order to obtain the surface current, the H-field close to the surface is computed. The distance to the surface and the segment length (the number of basis function per wavelength) affects the accuracy.

Starting with a frequency of 1 GHz, a radius of observation $R = 1.01$, and a segment length of 0.05, one obtains the currents in Fig. 14.A and the errors in Fig. 14.B.



(A) $|K_z|$ in solid line (TM) and $|K_\theta|$ in dashed line (TE).

(B) The error $|K_z - K_{zs}|$ in solid line, and $|K_\theta - K_{\theta s}|$ in dashed line.

FIGURE 14. The NERO solution and the errors for f=1 GHz, R=1.01.

When the frequency is increased to 10 GHz the series solution produces the result shown in Fig. 15.

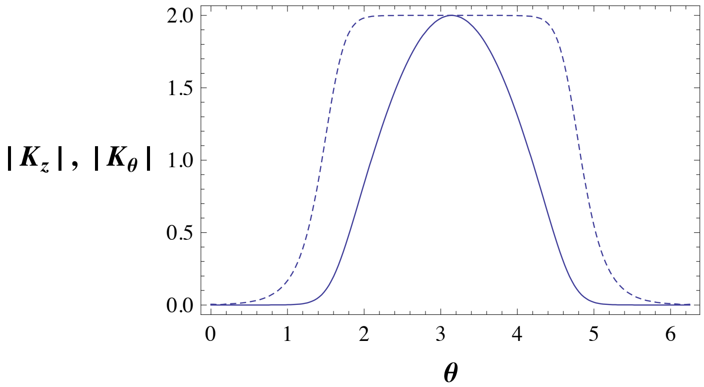
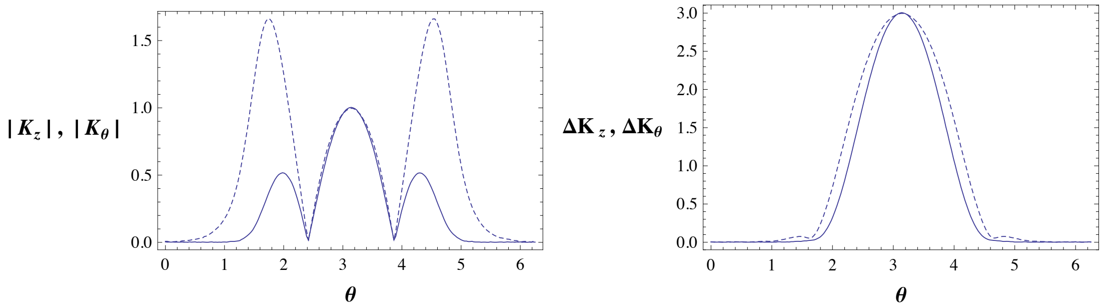


FIGURE 15. $|K_z|$ in solid line (TM) and $|K_\theta|$ in dashed line (TE), f=10 GHz .

The corresponding NERO result (R=1.01), in Fig. 16, has very poor accuracy.

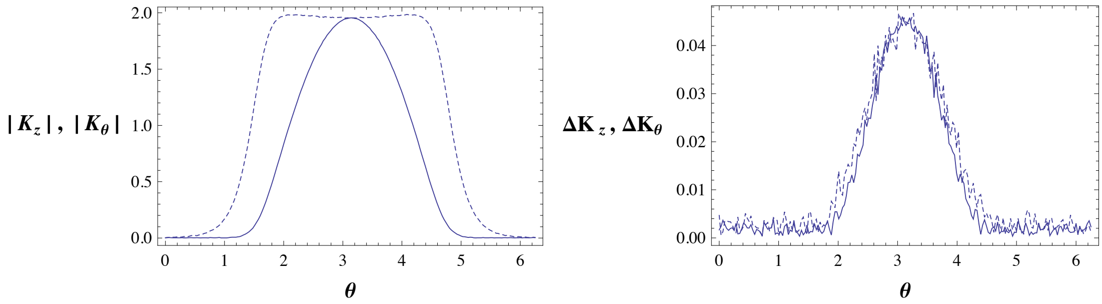


(A) $|K_z|$ (solid) and $|K_\theta|$ (dashed).

(B) The corresponding errors.

FIGURE 16. The NERO solution and the errors for f=10 GHz, R=1.01.

If the radius of observation is reduced to $R = 1.001$ one obtains Fig. 17.

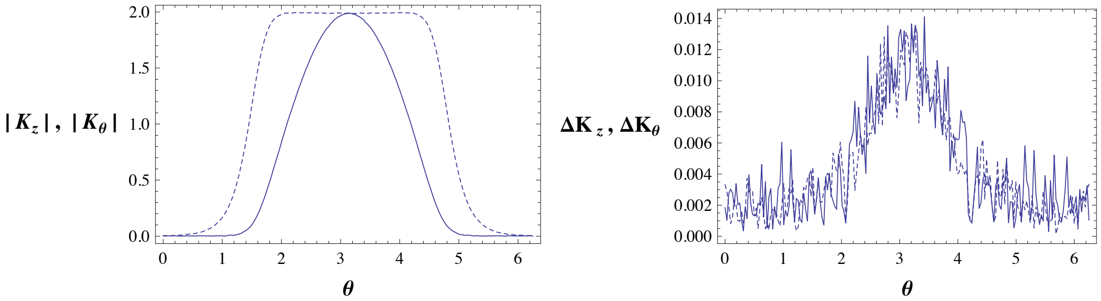


(A) $|K_z|$ (solid) and $|K_\theta|$ (dashed).

(B) The corresponding errors.

FIGURE 17. The NERO solution and the errors for f=10 GHz, R=1.001.

If the radius is reduced further to $R = 1.0005$ one obtains Fig. 18.

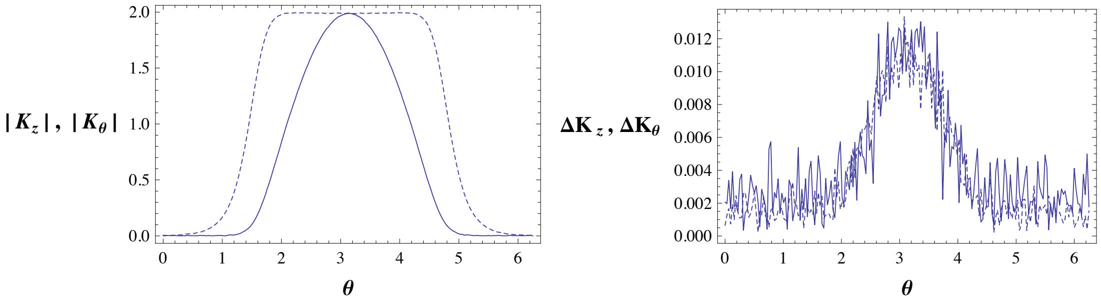


(A) $|K_z|$ (solid) and $|K_\theta|$ (dashed).

(B) The corresponding errors.

FIGURE 18. The NERO solution and the errors for f=10 GHz, R=1.0005.

Finally, the segment length is reduced to 0.01 and the result is shown in Fig.19.



(A) $|K_z|$ (solid) and $|K_\theta|$ (dashed), segment length=0.01.

(B) The corresponding errors, with segment length=0.01.

FIGURE 19. The NERO solution and the errors for f=10 GHz, R=1.0005, segment length=0.01.

In summary, Figure 13 shows a combination of parameters that leads to a mediocre accuracy. When the frequency is increased (Fig. 14) the accuracy is lost completely. Reducing the radius, as in Fig. 17, restores accuracy, since a higher frequency requires that the H-field is calculated closer to the surface. Further reduction of the radius is beneficial, as shown in Fig. 18. Figure 19 confirms that an increase in the number of basis functions gives a small improvement.

8. Pulse incident on a cylinder

The electromagnetic scattering problem can be interpreted as a linear system with one input (the incident field) and many outputs (the scattered field at all points in space) [5]. In this section we will study the far field of the scattered field at one point. An incident plane wave with Gaussian dependence x(t) is assumed to produce an output y(t).

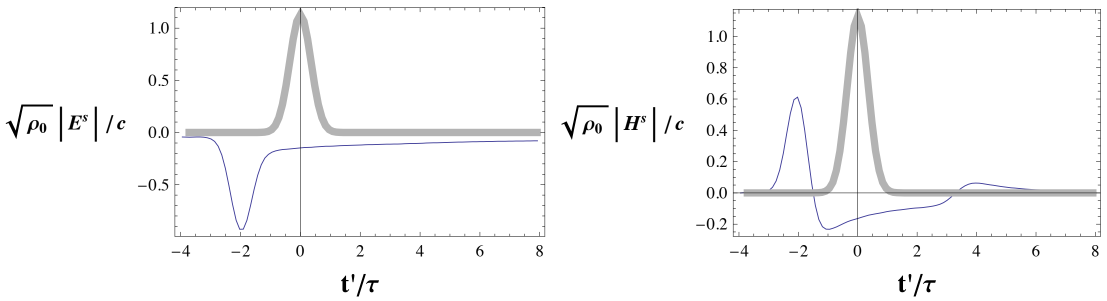
$$x(t) = (n/\pi)e^{(-n^2t^2)} \tag{29}$$

The frequency response $H(\omega)$ for this linear system is simply the ratio of the Fourier transform of the output to the Fourier transform of the input, i.e.,

$$H(\omega) = e^{(\omega/2n)^2}F\{y(t)\} \tag{30}$$

where $F\{y(t)\}$ represents the Fourier transform of the output [5]. The NERO code was used for most of the spectrum in order to compute the output signal for a number of equidistant frequency points, i.e. the spectrum Y(f). For low frequencies the series is used to avoid the low frequency breakdown of the FMM method. The impulse response y(t) was then obtained by means of the inverse Fourier transform. This result was compared to that obtained with the series solution and the results in [5].

Fig. 20A shows the incident and the backscattered fields for the TM case and Fig. 20B shows the incident and the backscattered fields for the TE case. In Fig. (20) the time $t' = 0$ corresponds to the time when the peak of the incident pulse would reach an observer at a distance ρ_0, if the incident pulse were reflected from the center of the cylinder. For this calculation the diameter of the cylinder was taken to be the width of the incident Gaussian pulse, i.e., n in Eq. 29 takes the value of $2/\tau$, where τ is the time required for a wave to travel one cylinder radius. Since a pulse that is reflected back from the cylinder travels a shorter distance than a hypothetical pulse reflected from the center, it is expected to arrive at $t = -2\tau$. In the TM case the expected impulse response has no indication of a creeping wave contribution. In the TE case, the initial pulse due to specular reflection is followed by a contribution that could indicate surface waves.



(A) The incident pulse x(t) (gray) and the backscattered pulse y(t) (blue) for TM case.

(B) The incident pulse x(t) (gray) and the backscattered pulse y(t) (blue) for TE case.

FIGURE 20. The incident and the backscattered fields.

The frequency response obtained in [5] used the Fourier transform of the approximate impulse response which had computed numerically and then used Eq. (30) to calculate the frequency response.

The TM case is formulated in terms of an E-field and the TE case in terms of an H-field. For the TM case, the total electrical field obtained when a linearly polarized electromagnetic wave is incident upon a perfectly conducting circular cylinder is the sum of the incident and the scattered wave.

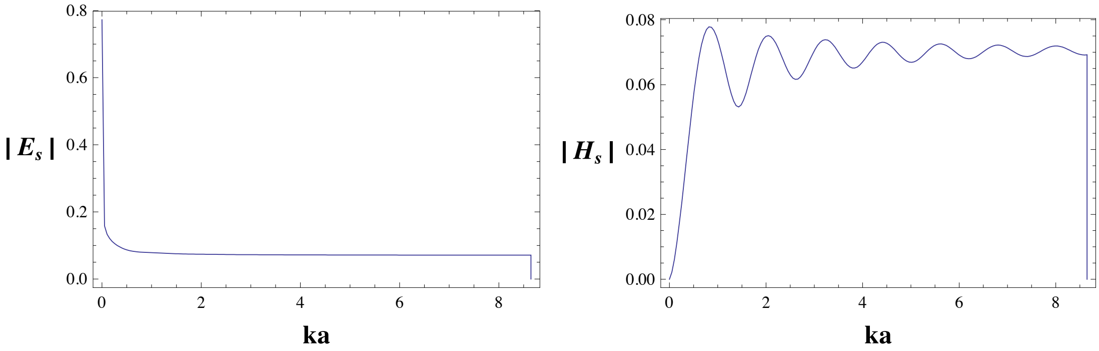
$$E_z = e^{ik\rho\cos\theta} - \sum_{m=-\infty}^{\infty} i^m \frac{J_m(ka)}{H_m(ka)} H_m(k\rho)e^{im\theta} \tag{31}$$

For the TE case one uses the H_z field.

$$H_z = e^{ik\rho\cos\theta} - \sum_{m=-\infty}^{\infty} i^m \frac{J_m'(ka)}{H_m'(ka)} H_m(k\rho)e^{im\theta} \tag{32}$$

$J_m(ka)$ is a Bessel function of order m, and $H_m(ka)$ is a Hankel function of the first kind and order m.

To compute the scattered field one uses the series part in Eqs. 31 and 32 for each frequency to compare with the results from the NERO code. Fig. (21) shows the frequency response obtained from the series solution for backscattering for both TM and TE. The formulation in [6] obtains the H-field all the time while we obtain the E-field for the TM case and the H-field for the TE case.



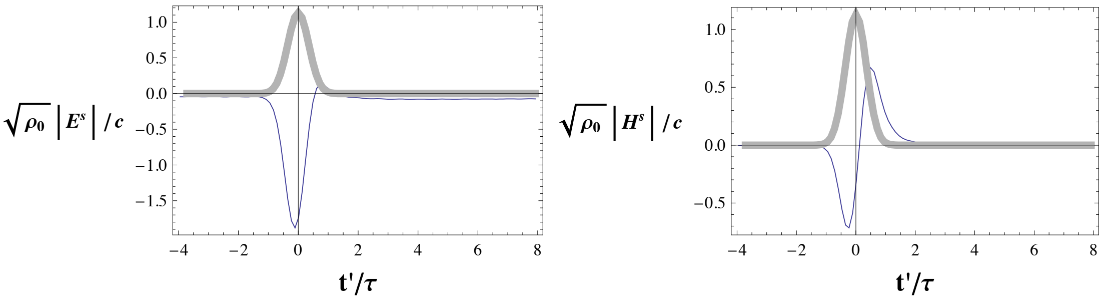
(A) $|E_s|$ is the frequency response obtained from the series solution for TM case.

(B) $|H_s|$ is the frequency response obtained from the series solution for TE case.

FIGURE 21. The backscattered frequency response obtained from the series solution.

Fig. (21) is compared with the results in [5] and gives good agreement for the backscattered frequency response.

Fig. 22A shows the incident and scattered fields for TM case and Fig. 22B shows the incident and scattered fields for TE case.



(A) The incident pulse x(t) (gray) and the forward scattered pulse y(t) (blue) for TM case.

(B) The incident pulse x(t) (gray) and the forward scattered pulse y(t) (blue) for TE case.

FIGURE 22. The incident and the forward scattered fields.

The conclusion from the tests with the NERO code is that the low frequencies must be handled with the series solution because of the low frequency breakdown of the FMM method. Below a certain frequency, Eqs. 31 and 32 are used instead of NERO to obtain the scattered fields.

9. Further work

An obvious extension is to use a modulated carrier instead of just a pulse. This would correspond to a realistic radio channel and one would also avoid the low frequency breakdown of NERO. A next step would be to apply the method to the setting in section 6. Eventually one would apply NERO to a realistic radio channel based on a terrain model.

References

[1] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in Fortran*. New York: Cambridge University press, 1992.

[2] N. Sherkat, "Approximation of antenna patterns with gaussian beams in wave propagation models.," Master's thesis, Linnaeus University, School of Computer Science, Physics and Mathematics, http://urn.kb.se/resolve?urn=urn:nbn:se:lnu:diva-14437, 2011.

[3] A. Haydar and I. Akeab, "Simulation of wave propagation in terrain using the fmm code nero2d," Master's thesis, Linnaeus University, School of Computer Science, Physics and Mathematics, http://urn.kb.se/resolve?urn=urn:nbn:se:lnu:diva-8767, 2010.

[4] M. Schwartz, *Mobile Wireless Communications*. New York: University of Cambridge, 2005.

[5] L. Bennett Jr. and W. L. Weeks, "Transient scattering from conducting cylinders," *IEEE AP-18*, pp. 627–633, 1970.

[6] A. Ludwig and Y. Leviatan, "Time-domain analysis of two-dimensional scattering problems by use of a sourse-model technique," *Proceedings of the URSI General Assembly, New Delhi, October*, 2005.

[7] J. Fostier and F. Olyslager, "An open-source implementation for full-wave 2D scattering by million-wavelength-size objects," *IEEE Trans. Antennas. Propag. Mag.*, vol. 52, pp. 23–34, 2010.

APPENDIX A. THE PROGRAM RUN_NERO

```
c------------------------------------------------------------------
 program run_nero                              !2014-09-02
c make run_nero
c ./run_nero
  integer i,j,i1,i2,i3,i4,nstep,igf_max
  real*8 f_,f,Fi1,Fi2,Fs1,Fs2,neta
  integer nn,isign,nnm,i_f,i_ff,nn0,i_dim
  parameter(i_dim= 100000)
  real*8 data_in(2*i_dim),data(2*i_dim),re,im,Pi,t,Dt,w,Dw
  real*8 n,zp,th,ka,rho,c,tau,tp,theta_n,theta_s
  real*8 t_min,t_max,f_lim
  complex*16 Fc(6),c_data_0,c_data,H,HV(i_dim),Ci,Ezser,Hzser,Ez,Hz
  complex*16 H0
  parameter(Pi= 3.141592653589793D0, Ci= (0.d0,1.d0),c= 2.9979245d8)
  character(80) string
  logical TM,Source,Line,IEQ,BACK_SC
  external Ezser,Hzser

  neta= 376.7303                              ! free space impedance
  open(40,file='spec_i.dat')
  open(41,file='spec_s.dat')
  open(42,file='spec_d.dat')
  open(51,file='spectrum_i.dat')
  open(53,file='spectrum_p.dat')
  open(55,file='time.dat')
  open(56,file='time_i.dat')

  zp= 2**4                                    ! zero padding > 2*4
  nn0= 2**6                                   ! time samples used
  nnm= nn0/2
  nn= nn0*zp                                  ! after zero padding
  if(nn .GT. i_dim) stop 'nn < i_dim'
  write(6,*)'nn=',nn
  tau= 1/c                                    ! time to travel unit radius
  n= 2/tau                                    ! sharpness of pulse
  Dt= 4.d-10                                  ! time step < limit
  Dw= 2*Pi/(Dt*nn)                            ! frequency step
  t_min= -4.; t_max= 8.                       ! plot window
  f_lim= 1.d-1
  IEQ=.True.                                  ! integral equation or series
  write(6,*)'IEQ=',IEQ
  do i= 1,nn
     t= (i-nnm)*Dt
     if(i .LE. nn0) then
       data_in(2*i-1)= n/sqrt(Pi)*exp(-n**2*t**2)
                                              ! gaussian pulse at t=0
 else
      data_in(2*i-1)= 0.d0                      ! zero padding
   endif
   data_in(2*i  )= 0.d0
   re= data_in(2*i-1) ; im= data_in(2*i)
   if(t/tau.GT.t_min .AND. t/tau.LE.t_max) then
      write(55,*) sngl(t/tau),sngl(re/c)
   endif
enddo
isign= 1
call four1(data_in,nn,isign)
call system("cp circle_TM.igf test.igf")

do i= 1,nn
   if(i .LE. nn/2) then
      i_f= i-1                                  ! index prop. to pos. freq.
                                                ! (0,fs/2)
else
      i_f= i-1-nn                               ! index prop. to neg. freq.
                                                ! (-fs/2,-DF)
   endif
   w= i_f*Dw
   if(IEQ .AND. i.LE.nn/2) then                 ! use integral equation
      if(i .LE. nn/6) then                      ! null elements in spectrum?
      open(1,file=’test.igf’)
      open(2,file=’circle_TM.igf’)
      igf_max= 1000
      do j=1,igf_max                            ! lines in .igf file(16 assumed)
         if(j .EQ. 2) then
            read( 1,*)f_,i1,i2,i3,i4
            f= max(abs(w/(2*Pi)),f_lim)         ! avoid zero frequency
            write(2,*)sngl(f),i1,i2,i3,i4       ! write modification
         elseif(j .EQ. 12) then
            read( 1,*)i1,i2,theta_n             ! read theta_n
            write(2,*)i1,i2,int(theta_n)        ! write back
            theta_n= theta_n*Pi/180.d0
            if(theta_n .GT. 1.d0) then
               BACK_SC=.True.
               theta_s= Pi
            else
               BACK_SC=.False.
               theta_s= 0.d0
            endif
         elseif(j .EQ. 15) then
            read( 1,*)i1,i2,rho,i4             ! read rho
            write(2,*)i1,i2,sngl(rho),i4       ! write back
         else
            read( 1,1,end= 3)string
            write(2,*)string                   ! write back the same line
```

```
           endif
        enddo
 3      continue
        close(1); close(2)                   ! input file edited
        call system("nero2d circle_TM.igf")  ! run nero
        open(26, file='freq.dat')            ! see program con.f
        read(26,*)f_ ; read(26,*)TM          ! extract polarization
        read(26,*)Source; read(26,*) Line    ! extract observation type
        close(26)
        if(TM) then ; write(6,*)'Polarization TM'
        else        ; write(6,*)'Polarization TE'
        endif
        if(Line) then ; write(6,*)'Line'
        else          ; write(6,*)'Circle'
        endif

        open(11,file='line0_in.bdf')         ! incoming fields
        open(12,file='line0_sc.bdf')         ! scattered fields
        open(13,file='circle0_in.bdf')
        open(14,file='circle0_sc.bdf')
        do j= 1,6                            ! read field at first observation point
           if(Line)then
              read(11,*)Fi1,Fi2
              read(12,*)Fs1,Fs2
           else
              read(13,*)Fi1,Fi2
              read(14,*)Fs1,Fs2
           endif
           Fc(j)= 0*dcmplx(Fi1,Fi2)+ 1*dcmplx(Fs1,Fs2)
                                             ! scattered field
        enddo
        close(11); close(12); close(13); close(14)
                                             ! spectrum point extracted
        if(TM) then                          ! store spectrum point
           H= conjg(Fc(3))                   ! Ez - physics convention
           if(f .LE. 1.d8) then              ! use series
              ka= max(abs(w),2*Pi*f_lim)/c   ! handle low frequency limit
              th= theta_s
              H0= Ezser(ka,ka*rho,th)
              write(41,*)sngl(f),sngl(abs(H0))
              write(42,*)sngl(f),sngl(abs(H-H0))
              H= H0
           endif
        else
c           H= 1/neta*Fc(2)                  ! Ey
           H= conjg(Fc(6))                   ! Hz - physics convention
           if(f .LE. 1.d8) then
              ka= max(abs(w),2*Pi*f_lim)/c
              th= theta_s
```

```
            H0= Hzser(ka,ka*rho,th)
            write(41,*)sngl(f),sngl(abs(H0))
            write(42,*)sngl(f),sngl(abs(H-H0))
            H= H0
         endif
      endif
      else
         H= (0.d0,0.d0)                       ! null in spectrum
      endif

      write(40,*)sngl(f),sngl(abs(H))
      HV(i)= H
   elseif(i .LE. nn/2) then                   ! non-negative frequencies
      th= theta_s                             ! th=Pi backcattering, (TM,Ez),
                                              ! (TE,Hz)

      if(i .EQ. 1) write(6,*)'th=',sngl(th)
      rho= 100.d0
      ka= max(abs(w),2*Pi*f_lim)/c            ! avoid zero argument
      if(TM) then                             ! series solution for cylinder
         H= Ezser(ka,ka*rho,th)
      else
         H= Hzser(ka,ka*rho,th)
      endif
      write(41,*)sngl(ka*c/(2*Pi)),sngl(abs(H))
      HV(i)= H
    endif

   if(i .LE. nn/2) then
      c_data= HV(i)*dcmplx(data_in(2*i-1), data_in(2*i))*Dt
   else
      i_f= nn- i+ 2
      H= HV(i_f)
      c_data= conjg(H)*dcmplx(data_in(2*i_f-1),-data_in(2*i_f))*Dt
   endif
   data(2*i-1)= real( c_data)
   data(2*i  )= aimag(c_data)
   re= data(2*i-1) ; im= data(2*i)
   write(51,*) i,sqrt(re**2 + im**2)
   write(53,*) i,abs(H)*exp(-0.25*(w/n)**2)   ! explicit Fourier transform
enddo
isign= -1
call four1(data,nn,isign)
do i= 1,nn
   re= data(2*i-1)/(Dt*nn) ; im= data(2*i)/(Dt*nn)
   t= (i-nnm)*Dt
   tp= t-rho*tau                              ! shifted time
   if(tp/tau.GT.t_min .AND. tp/tau.LE.t_max) then
                                              ! the output is normalized with
```

```
                                                   ! 1/c
       write(56,*) sngl(tp/tau),sngl(re*sqrt(rho)/c)
     endif
  enddo
  if(BACK_SC) then ; write(6,*)'Backscattered pulse'
  else             ; write(6,*)'Forwardscattered pulse'
  endif
  write(6,*)'pls,plu,plc,plx'
                                                   ! plot files for spectrum, output
                                                   ! spectrum, diff. spectrum

 1format(A80)
  stop
  end

c-------------------------------------------------------------------
```

Appendix B. Mathematica script that calculates the impulse response for the two ray channel and simple AM modulation

```
fact= 2^38
nmax= 2^(-26)*fact
Print["nmax= ",nmax]
tf[t_]:= t/fact
fc= 2.*10^9
wc= 6*fc

(* time delay of reflected wave *)
h1= 10
h2= 20
d= 1*10^3
c= 3*10^8
td= N[2*h1*h2/(c*d)]
Print["td= ",td]

(* create input function x(t) (impulse) *)
a= 1.*10^-9
Print["a= ",a]
dirac[t_]:= Exp[-(t/(fact*a))^2]/(a*Sqrt[Pi])
m= Table[dirac[t],{t,-nmax,nmax-1}]
mp= ListPlot[m, PlotRange->All, PlotJoined->True]
Export["TIMEDOMAIN/m.pdf",mp,"PDF"]

(* For AM-DSBSC *)
co= Table[Cos[wc*t/fact],{t,-nmax,nmax-1}]
x= m*co
```

```
xp= ListPlot[x, PlotRange->All, PlotJoined->True]
Export["TIMEDOMAIN/x.pdf",xp,"PDF"]
(* signal reflected from ground/water *)
xd= Table[dirac[t-td*fact],{t,-nmax,nmax-1}]*co

(* Fourier transform X(w) *)
Gam= -0.2+ I*0.00
Print["Gam= ",Gam]
X= Fourier[x+Gam*xd]
(* Plot with standard ordering Xso(w) *)
Xmp= Join[Take[X,-nmax],Take[X, nmax]]
Xso= ListPlot[Re[Xmp], PlotJoined->True,PlotRange->All]
Export["TIMEDOMAIN/Xso.pdf",Xso,"PDF"]

(* create impulse response h(t)*)
u[t_]:= (1+Sign[t])/2
tau= 1.*10^-9
Print["tau= ",tau]
h= Table[u[t]*tf[t]*Exp[-tf[t]/tau],{t,-nmax,nmax-1}]
hp= ListPlot[h, PlotRange->All, PlotJoined->True]
Export["TIMEDOMAIN/h.pdf",hp,"PDF"]
(* Fourier transform H(w) *)
H= Fourier[h]
Hmp= Join[Take[H,-nmax],Take[H, nmax]]
Hso= ListPlot[Re[Hmp], PlotJoined->True,PlotRange->All]
Export["TIMEDOMAIN/Hso.pdf",Hso,"PDF"]


(* output signal- demodulation *)
Y=  Sqrt[2*nmax]/fact*H*X
(* Demodulate *)
y= InverseFourier[Y]*co
(* LP-filter for demodulation *)
null= Table[If[Abs[w]<nmax*0.99,0,1],{w,-nmax,nmax-1}]
Yt0=  Fourier[y]
Ytmp= Join[Take[Yt0,-nmax], Take[Yt0,nmax]]
Ytso= ListPlot[Re[Ytmp], PlotJoined->True,PlotRange->All]
Export["TIMEDOMAIN/Ytso0.pdf",Ytso,"PDF"]
Yt=  Fourier[y]*null

(* Output spectrum *)
CC= (1.+1.*I)*10^(-20)
Ytmp= Join[Take[Yt,-nmax], Take[Yt,nmax]]
Put[Ytmp[[1]]+CC,"TIMEDOMAIN/four.dat"]
Do[
PutAppend[Ytmp[[i]]+CC,"TIMEDOMAIN/four.dat"],
{i,2,2*nmax}]
Ytmp=ReadList["TIMEDOMAIN/four.dat"]
Ytso= ListPlot[Re[Ytmp], PlotJoined->True,PlotRange->All]
```

```
Export["TIMEDOMAIN/Ytso.pdf",Ytso,"PDF"]

(* Output signal *)
yt= InverseFourier[Yt]
ytmp= Join[Take[yt,-nmax], Take[yt,nmax]]
yts= ListPlot[Re[ytmp], PlotJoined->True,PlotRange->All]
Export["TIMEDOMAIN/yts.pdf",yts,"PDF"]
   ClearAll
```

Linnæus University
School of Computer Science, Physics and Mathematics

SE-391 82 Kalmar / SE-351 95 Växjö
Tel +46 (0)772-28 80 00
dfm@lnu.se
Lnu.se/dfm