PART II. THE TRANSFER FUNCTION PROGRAM

5. TECHNICAL ASPECTS OF THE PROGRAM

This section discusses special techniques used in the design of the transfer function program. The larger data structures and the more complex or detailed routines are explained or justified. More details can be found in the documentation section, Section 7.

5.1. Random Number Generation

The assumption that long-term channel-fading characteristics follow a Rayleigh distribution requires two separate sequences from independent, normal distributions that have common zero mean and common unit variance. The polar method is used to generate two such sequences, see Law and Kelton [10, p. 491]. In turn, the polar method requires two separate sequences from independent uniform distributions on the interval (0,1) as input. To ensure independence, this program includes two different pseudorandom number generators that are each composites of several different linear congruential uniform random number generators (LCG).

The difficulty with a single LCG, which sometimes is used as input for the polar method, is that the normal distributions obtained may not be independent, see Brately, Fox, and Schrage [21, p. 223]. Example output sequences from the polar method, when plotted against one another, fall on a spiral! Clearly, these are not independent number streams.

5.2. Complex Number Arrays

An array of complex numbers is represented by an array of real numbers, that is, an array of floating point numbers. The even indices of the array, beginning with 0, indicate the real part of the complex number, while the odd indices indicate the imaginary part. The array elements with indices 0 and 1 are the real and imaginary part of the first complex number, respectively; the array elements with indices 2 and 3 are the second complex number, etc. The size of these arrays is driven by the channel simulation hardware requirements.

The random number input to the computation of the impulse response is held in a dynamically allocated array of floating point numbers. Stepping through this array is accomplished by pointer arithmetic. One section of this array is maintained for each ionospheric path. Dynamic memory is used since the program will support from one to three skywave paths. Hence, up to three separate segments of the array will be necessary. The array is used as both input and output for a digital filter.

5.3. Fast Fourier Transform

The fast Fourier transform (FFT) algorithm used returns the complex Fourier coefficients into the input array, hence only one array is necessary. The length of the input array is doubled by zero padding. This padding provides interpolation and helps avoid “wrap-around” effects.

5.4. Computation of τ_l and α

The parameters τ_l and α must be calculated iteratively since the equation for the gamma distribution has no closed form unless α is an integer. The computation of τ_l is accomplished by a bisection algorithm since the function being evaluated is not continuous. The presence of a natural logarithm in the function can produce complex results, see Section 7.4.4. The bisection algorithm is used since two estimates that lead to a result with the desired tolerance can be quickly found while avoiding the discontinuities. The shape parameter α is calculated directly from

$$\alpha = \frac{\ln s_v}{\ln Z_L + 1 - Z_L}, \tag{22}$$

where s_v is given by (13) and

$$Z_L = \frac{\tau_L - \tau_l}{\tau_c - \tau_l}, \tag{23}$$

see Vogler and Hoffmeyer [5, p. 32].

5.5. Data Structures

The first of the major data structures is a single element array containing one structure that holds all input and computed parameters that apply to all the paths. There is also an array of structures that holds all the input and calculated parameters particular to each sky wave path.

6. USER'S GUIDE

This section provides instructions to use the program. Several topics are discussed including input and output, and running the program under various environments.

6.1. The Input File

An input file contains a variable number of parameters depending on the number of ionospheric paths in the HF channel situation under consideration. The parameters are listed in order and separated by delimiters such as carriage return and line feed (CR-LF). The input files are ASCII files that consist of two types of information: data used by the program in the computing run and data descriptive of each path. The first five items in an input file are the "computing" data and are read into a single element array of type **compute** (see the documentation in Section 7). The italics here indicate a variable in the program while bold indicates a structure. Where a real number is indicated, a decimal point is required, for example, 326.0. The first five input elements are:

slices - an integer that indicates the number of time slices or samples in the time direction. The program computes the Fourier coefficients for a complete impulse response for each sample point.

delta_t - a real number that indicates the length of the sampling interval. This is given in microseconds for consistency with the units of the other time values in the input file. In Vogler and Hoffmeyer [5], this is indicated by the symbol t_m.

afl- a real number that indicates the receiver threshold. This is generally taken to be the half amplitude point (the 3-dB point).

paths - an integer that indicates the number of ionospheric paths modeled. The upper limit is three paths.

seed - an integer in the range 1 - 30,268, inclusive, that is used to initialize the random number seeds for the two uniform random number generators.

The parameters that follow serve to characterize the individual skywave paths. These parameters are all elements of the structure **ray_path**. For each path or layer, there are eleven parameters:

path_Distance - a real number that indicates the point-to-point distance, in kilometers, along the great circle between sender and receiver.

center_freq - a real number that indicates the center frequency, in megahertz, of the transmitting radio system.

penetrate_freq - a real number that indicates the penetration frequency, in megahertz, of the reflecting layer. Above this frequency there is no ionospheric reflection possible for this layer. The penetration frequency is generally higher than the maximum useable frequency (MUF) for the most highly ionized layer.

thick_scale - a real number that indicates the thickness, in kilometers, of the reflecting layer.

maxD_hgt - a real number indicating the height, in kilometers, of the maximum electron density of the layer.

peak_amplitude - a real number that indicates the maximum power A of the signal.

sigma_tau - a real number equal to the delay spread, in microseconds, at the half power point (3dB) A_{fl}.

sigma_c - a real number that indicates the rise time, in microseconds, of the impulse response with respect to the lower end of the delay spread τ_L .

sigma_D - a real number that gives the Doppler spread, in Hertz, at the half power point (3dB) A_{fl}.

fds - a real number that gives the Doppler frequency, in Hertz, indicating the Doppler shift at the mean delay τ_c.

fdl - a real number that equals the Doppler frequency, in Hertz, indicating the Doppler shift at the lower end of the delay spread τ_L.

If the last two parameters are equal, *fdl* = *fds,* then the transfer function will not be characteristic of the slant phenomenon. Slant characterizes Doppler frequency dependence on delay. This aspect of the channel is modeled as a linear relationship, see section 2.1.3. pp. 15-16 in Vogler and Hoffmeyer [4].

The input files can be generated with any editor that can save files in ASCII format. Examples of input files are given in Table 3. File 1 is an example input file for a situation with one layer. File 2 shows an example input file for two layers.

Table 3. Example Input Files

File 1	**Parameter**	**File 2**
512	*slices*	512
1,000,000	*delta_t*	500,000
0.5	*Afl*	0.5
1	*paths*	2
1	*seed*	1
126.0	*path_Distance*	88.0
5.5	*center_freq*	5.3
13.0	*penetrate_freq*	7.07
30.0	*thick_scale*	20.0
265.0	*maxD_hgt*	225.0
1.0	*peak_amplitude*	0.8
70.0	*sigma_tau*	80.0
35.0	*sigma_c*	39.2
0.05	*sigma_D*	0.12
0.2	*fds*	-0.05
0.1	*fdl*	-0.05
	path_Distance	88.0
	center_freq	5.3
	penetrate_freq	5.56
	thick_scale	26.0
	maxD_hgt	228.0
	peak_amplitude	1.0
	sigma_tau	240.0
	sigma_c	117.8
	sigma_D	0.15
	fds	-0.05
	fdl	-0.05

6.2. The Output Files

6.2.1. The Transfer Function

This output file contains the complex Fourier coefficients that define the transfer function. The coefficients are written to the file as real floating point numbers without exponent and with a space between each field. There are 8,192 coefficients written for each time slice. Between the transfer functions for each time slice, a line feed is included; however, this should generally be seen as white space. The 8,192 coefficients represent 4,096 complex numbers with the real and imaginary parts alternating. The first number in the list is the real part of the first complex coefficient and the second number is the corresponding imaginary part, and so forth.

The present version of the HF channel hardware simulator expects coefficients for 299 time slices. Since the output file will be large, it is recommended that the transfer function software be run on the same machine driving the hardware simulator. Alternately, a network supporting large file transfers is recommended.

6.2.2. The Parameter Listing

This output file lists the parameters used by the program. All the input file parameters, as well as the important derived parameters are written to this file. The derived parameters are listed here in terms of their variable name in the program.

The computing parameters contained in the structure **compute** and derived by the program are as follows:

delta_tau - the delay interval in microseconds.

big_el - the earliest of the *tau_l* values for the layers in microseconds. If that value is negative, then *big_el* is set to 0.0 μs.

The following are the parameters computed for each layer contained in structure **ray_path** which are derived by the program:

*tau*_c - the mean delay associated with the carrier frequency, in microseconds.

sigma_f - value used to determine the Doppler spread shape.

sip - the slope of the linear relationship between the delay and the Doppler shift. The units are Hertz / microseconds.

tau_L - left end of the delay spread, in microseconds.

tau_U - right end of the delay spread, in microseconds.

tau_l - location parameter for the delay function in microseconds. This is not necessarily the same for every layer.

alpha - shape parameter for the delay power profile.

sigma_l - the rise time of the impulse response with respect to *tau_l* in microseconds.

lambda - exponential autocorrelation factor through time, for random input stream construction.

6.3. Running the Program

This section describes how to run the program in Windows 95 or Windows 3.0 / 3.1. It is assumed that the executable file, LEWS.EXE, is in the directory to which the output files will be written. The program writes the transfer function output file by appending to the file given on the command line. Hence, if the file name listed on the command line already exists, then the transfer function coefficients will be appended to that file. The program opens a new file for the parameter listing output file. Therefore, if the file name listed on the command line already exists, that file will be overwritten and all contents lost. The command line to start the program consists of the executing file, the input file, and the two output files:

```
< LEWS.EXE INPUT_FILENAME FIRST_OUTPUT_FILENAME SECOND_OUTPUT_FILENAME >
```

where FIRST_OUTPUT_FILENAME indicates the parameter listing output file and < > encloses the characters to'be typed.

6.3.1. Windows 95

The executing program and the input file should be available in the same directory to which the output files will be written. Click the start button on the task bar. Click on Run from the start menu. Type the full command line, including the path to the directory (the browse capability may also be utilized). For example, type:

```
< D:\BORLANDC\BIN\LEWS.EXE INPUT.931 OUTPUTA.931 OUTPUTB1.931 >.
```

Click OK or hit the return key.

6.3.2. Windows 3.1 /3.0

The executing program and the input file should be available in the same directory to which the output files will be written. From the program manager, click on File (or type Alt-F). Then click Run (or type R). Type the full command line, including the path to the directory (the browse capability may also be used). For example, type:

```
< C:\BORLANDC\BIN\LEWS.EXE INPUT.931 OUTPUTA.931 OUTPUTB1.931 >.
```

Click OK or hit the return key.

6.4. Modifying the Program

The program may also be run directly from Borland C++ 3.1 using the standard procedures in the environment. This procedure is recommended if changes to the source code are made. Other C compiler/environments may also be used.

The program is written with the following constraints, limitations, and options:

- Target is a Windows 3.0 and above executable file.
- Large memory model.
- Full floating point is enabled.
- Source code is Borland C++ without objects.