User's Manual for the Scanning Fast Field Program (SCAFFIP) General Version 1.0

by John M. Noble
Dave Marlin
Battlefield Environment Directorate

ARL-TR-545
January 1995
NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

The citation of trade names and names of manufacturers in this report is not to be construed as official Government indorsement or approval of commercial products or services referenced herein.

Destruction Notice

When this document is no longer needed, destroy it by any method that will prevent disclosure of its contents or reconstruction of the document.
The Scanning Fast Field Program (SCAFFIP) is an atmospheric acoustic propagation model incorporating many of the effects on the environment on the sound field such as geometrical spreading, refraction, diffraction, molecular absorption, and complex ground impedance. SCAFFIP provides attenuation levels with range and azimuth or sound pressure levels in dB (re: 20 µPa) with range and azimuth for a given geometry, frequency, and meteorological profile. The meteorological profile and the geometry provides the model the ability to calculate the sound speed profile. The geometry profile is required because the angular dependence of the sound speed on the wind direction is relative to the direction of propagation. This model works over a flat-earth and a non-turbulent atmosphere. Even with these restrictions, the model performs very well for many scenarios. The model contains a friendly user interface requiring a minimum amount of information to run the model. There are also flags that can be set to obtain more detailed information.
Figures

1. Diagram of geometry definition ................................................. 8
2. Layering of the atmosphere by the FFP ...................................... 10
3. Transmission line analog used by the FFP .................................. 12
4. Log-log plot of sound-absorption coefficient versus frequency for sound in air at 20 °C at 1 atm pressure and with a water-vapor fraction h of $4.676 \times 10^{-3}$ (Rh = 20 %) ................................................. 19

Tables

1. Transmission line analog relationships to acoustic fields .................. 13
2. Estimation of $\Omega$ and $\sigma$ for various ground surfaces given $n' = 0.750$ and $S_r = 0.875$ ................................................. 22
1. Introduction

The Scanning Fast Field Program (SCAFFIP) is based on the Fast Field Program (FFP) with the added ability to scan multiple azimuths to predict the propagation conditions about the location of a sensor. SCAFFIP makes a prediction of the acoustic propagation conditions based on spherical spreading, molecular absorption, refraction, acoustically complex ground impedance, and diffraction over benign terrain.

The FFP is a one-way solution to the acoustic-wave equation originally developed for underwater sound propagation predictions. [1,2] Raspet et al. [3] and Lee et al. [4] adapted the FFP to propagation in the atmosphere. The FFP developed by Raspet et al. used a propagation matrix formulation. If each layer in the atmosphere is viewed as an optical device, a matrix for each layer in the atmosphere can be constructed. Multiplying each matrix together results in a new matrix representing how an acoustic signal will be affected as it propagates through the atmosphere. Next, a Bessel Function Transform of the problem is taken with respect to range. After the solution is calculated, an inverse transform is performed to arrive at the final solution.

The software package that comes with SCAFFIP is an integrated set of algorithms for running the acoustic propagation model on any computer platform containing an ANSI version of a C compiler. There is an option of using the nongraphical user interface that comes with the package for extracting the propagation model and incorporating it into the user interface environment (see appendix D). Example test cases (in appendix B) show the structure of the input data files used by SCAFFIP and give a basis to work with to investigate the installation of the files. The software package is available on a variety of computer media. A list of possible media is 4-mm Digital Audio Tape (TAR or CPIO format), 0.25-in. cartridge tape (TAR or CPIO format), 8-mm Exabyte tape (TAR or CPIO format), MS-DOS 3.5-in. high- or low-density diskettes, or 0.5-in. 9-track magnetic tape (TAR or CPIO format).
2. Theory

2.1 Speed of Sound

Meteorological phenomena can have a significant effect on the received sound field. Some of the meteorological variables affecting the speed of sound in air are: pressure, temperature, vector wind speed, and humidity. To observe the effect of each meteorological variable, consider each one independently and examine the equation for the speed of sound in air. The value of \( c \), according to Laplace’s adiabatic assumption for air as an ideal gas, [5] is

\[
c(T) = \sqrt{\frac{\gamma RT}{M}}
\]  

(1)

where

\[
\gamma = \text{the ratio of specific heats}
\]

\[
R = \text{the universal gas constant equal to 8314.16 J/(kg K)}
\]

\[
M = \text{the molecular weight.}
\]

The presence of water molecules alters the sound speed by lowering \( \gamma \) and decreasing \( M \). The decrease in \( M \) dominates so that the overall effect of increasing humidity is an increasing sound speed. These changes can be quantified as

\[
\gamma = \frac{7 + h}{5 + h}
\]  

(2)

and

\[
M = 29 - 11h
\]  

(3)

where

\[
h = \text{the fraction of water molecules in air.}
\]
As the amount of water in the atmosphere increases, the molecular weight of a parcel of air will decrease because the molecular weight of a water molecule is less than diatomic nitrogen. This molecular weight effect will try to increase the sound speed as the fraction of water molecules in the air increases. To calculate the fraction of water molecules in air, the Goff-Gratch equation, equation (4), must be used to first calculate the partial pressure of saturated water vapor, $P_{\text{sat}}$ at temperature $T$.

$$\log_{10}\left(\frac{P_{\text{sat}}}{P_0}\right) = 10.79586 \left[ 1 - \left(\frac{T_{01}}{T}\right) \right] - 5.02808 \log_{10}\left(\frac{T}{T_{01}}\right)$$

$$+ 1.50474 \times 10^{-4} \left(1 - 10^{-8.29692 \left(\frac{T}{T_{01}}\right) - 1}\right)$$

$$+ 0.42873 \times 10^{-3} \left(10^{4.76955 \left(1 - \frac{T_{01}}{T}\right) - 1} - 1\right) - 2.2195983$$

where

- $T_{01} = 273.16$ K
- $P_0 = 1$ atm or the reference pressure.

After the value for $P_{\text{sat}}$ is determined, the fraction of water molecules in air can be calculated using the following relationship:

$$h = \frac{10^{-2} (RH) P_{\text{sat}}}{P}$$

where

- $RH = \text{the relative humidity in percent}$
- $P = \text{the pressure in atmospheres}$.

The magnitude of the dependence of the sound speed on humidity is not obvious. To understand the degree of the effect of humidity on sound speed, consider a particular case. At 20 °C, the difference in sound speed between 0 and 100 percent humidity is 2 m/s. A fluctuation in the humidity of this amount is very unlikely. If the variation in humidity is reduced to a change of 50 to 100 percent, the change in the sound speed is only 1 m/s. Therefore, the
variation of sound speed caused by changes in humidity should always be much less than 1 m/s. Generally, humidity fluctuations can be ignored.

The effect of the wind speed on the speed of sound is a vector relation. The effective sound speed is calculated using

\[ c_{\text{eff}} = c(T) + u \cdot \cos(\theta_w - \pi - \theta_R) \]  

(6)

where

- \( c(T) \) = the speed of sound in the absence of wind at temperature T
- \( u \) = the magnitude of the horizontal wind speed
- \( \theta_R \) = the bearing of the receiver from the source
- \( \theta_w \) = the direction from which the wind blows
- \( \theta_w - \pi \) = the direction the wind is blowing (figure 1).

All directions are relative to north.

The sound speed will also vary with height because the sound speed is a function of temperature and vector wind speed. This variation will cause the acoustic wave to be refracted as it propagates through the atmosphere. The degree of refraction the acoustic wave undergoes is related to the sound speed gradient present in the atmosphere. If the sound speed increases with height, the acoustic wave will be refracted downwards. If the sound speed decreases with height, the acoustic wave will be refracted upwards.
2.2 FFP

The propagation of sound from a point source located at the origin is given by the classical wave equation

\[ \nabla^2 p - \frac{1}{c^2} \frac{\partial^2 p}{\partial t^2} = -4\pi \delta(x,y,z) \]  
(7)

where \( \delta \) represents a delta function source of unit strength. For simple harmonic motion, equation (7) becomes the Helmholtz equation

\[ \nabla^2 p + k^2 p = -4\pi \delta(x,y,z) \]  
(8)
where

\[ k = \text{is the wavenumber, } \omega/c, \text{ in which } c = \text{the sound speed, } \omega = \text{the angular frequency}. \]

For the FFP, \( k \) and \( c \) are restricted to vary only in the \( z \)-direction.

Transforming equation (8) into cylindrical coordinates and assuming azimuthal symmetry, the Helmholtz equation becomes

\[
\frac{\partial^2 p}{\partial r^2} + \frac{1}{r} \frac{\partial p}{\partial r} + \rho \frac{\partial}{\partial z} \left( \frac{1}{\rho} \frac{\partial p}{\partial z} \right) + k^2 p = -\frac{2}{r} \delta(r) \delta(z-z_s)
\]  

where the source is located at \( r = 0 \) and \( z = z_s \) and \( \rho \) is the density of the medium.

The atmosphere is viewed as a series of constant sound speed layers for the FFP (as shown in figure 2). The layers in the atmosphere are bounded on top and bottom by complex impedance surfaces. The top boundary is typically modeled as an infinite half-space with constant parameters. At the bottom boundary, the atmospheric layer adjoins a partially absorbing surface that can be represented by the complex acoustical impedance of the ground.

To reduce the dimensionality of equation (9), a zero-order Hankel transform is applied with respect to the range variable \( r \). This gives the transform pair:

\[
\tilde{p}(\kappa, z) = \int_0^\infty p(r, z) J_0(\kappa r) r \, dr
\]

\[
p(r, z) = \int_0^\infty \tilde{p}(\kappa, z) J_0(\kappa r) \kappa \, d\kappa.
\]
Applying the first transform to equation (9) results in

$$\frac{d^2 \hat{p}}{dz^2} + \left[ k^2(z) - \kappa^2 \right] \hat{p} = -2 \delta(z-z_f).$$

(11)

<table>
<thead>
<tr>
<th>Layer</th>
<th>Impedance $Z_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$Z_0$</td>
</tr>
<tr>
<td>1</td>
<td>$Z_1$</td>
</tr>
<tr>
<td>2</td>
<td>$Z_2$</td>
</tr>
<tr>
<td>...</td>
<td>$Z_R$</td>
</tr>
<tr>
<td>R</td>
<td>Receiver</td>
</tr>
<tr>
<td>R+1</td>
<td>$Z_{R+1}$</td>
</tr>
<tr>
<td>...</td>
<td>$Z_s$</td>
</tr>
<tr>
<td>S</td>
<td>Source</td>
</tr>
<tr>
<td>S+1</td>
<td>$Z_{S+1}$</td>
</tr>
<tr>
<td>...</td>
<td>$Z_{N-1}$</td>
</tr>
<tr>
<td>N-1</td>
<td>$Z_{N-1}$</td>
</tr>
</tbody>
</table>

Figure 2. Layering of the atmosphere by the FFP.
This equation can be decomposed into

\[ \ddot{u}_z = \frac{i}{\omega \rho_o} \frac{d\ddot{p}}{dz} \]  \hspace{1cm} (12)

and

\[ \frac{d\ddot{u}_z}{dz} = \frac{i}{\omega \rho_o} \frac{d^2\ddot{p}}{dz^2} = \frac{i}{\omega \rho_o} \left[ k^2(z) - \kappa^2 \right] \ddot{p} + \frac{2}{i \omega \rho_o} \delta(z-z_s) \]  \hspace{1cm} (13)

where

- \( \rho_o \) = the mean air density
- \( \ddot{u}_z \) = the transformed particle velocity in the z-direction.

The delta function is the source term producing a discontinuity in \( \ddot{u}_z \) at a height \( z_s \). To solve the resulting equations, Lee et al. [4] used an analogy to a transmission line, which results from the form of the transformed equation. The transformed equations have a very similar form to the telegrapher's equations of electrical transmission line theory. From the analogous telegrapher's equations,

\[ \frac{dV}{dz} = -Z I(z) \]  \hspace{1cm} (14)

and

\[ dI(z) dz = -Y(z) V(z) + I_o \delta(z-z_s) \]  \hspace{1cm} (15)

there is a similar form of the equations if the shunt admittance \( Y(z) \) is made a function of \( z \) and the series impedance \( Z \) is made a constant with a current source at \( z = z_s \). The equivalent transmission line configuration to figure 2 is shown in figure 3.
Using the Lee et al. analogy, the acoustic problem can be arranged so that a solution can be calculated. The analogy is made by representing each layer in the atmosphere by an element in a transmission line with a certain characteristic admittance and attenuation constant. The admittance is defined as one over the impedance or one over the sum of the resistance plus the reactance of the electrical element. The admittance of the element causes the voltage running through it to be attenuated and a shift in the phase of the signal. Viewing the analogy from the perspective of the acoustic wave problem, as the acoustic wave is propagated through the atmosphere, it undergoes losses and phase shifting caused by refraction and spreading of the acoustic wave. The equivalence between the transmission line and the acoustic...
wave problem is illustrated in table 1. This equivalence can be carried out for each layer of the atmosphere; thus, constructing a transmission line. The top and bottom boundaries in the atmosphere become loading admittance elements on each end of the transmission line. The problem now has been converted from determining the voltage in the transmission line to a point in the line. This is a well-known process in electrical engineering.

Table 1. Transmission line analog relationships to acoustic fields

<table>
<thead>
<tr>
<th></th>
<th>Acoustic Wave</th>
<th>Transmission Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fields</td>
<td>Pressure p(k,z)</td>
<td>Voltage V(z)</td>
</tr>
<tr>
<td></td>
<td>Velocity ( \omega (k,z) )</td>
<td>Current I(z)</td>
</tr>
<tr>
<td>Attenuation Constant</td>
<td>( \gamma = \sqrt{k^2 - (\omega/\alpha)^2} )</td>
<td>( \gamma = \sqrt{Z I} )</td>
</tr>
<tr>
<td>Characteristic Admittance</td>
<td>( Y_c = -\gamma/\omega \rho )</td>
<td>( Y_c = \sqrt{Y/Z} )</td>
</tr>
</tbody>
</table>
The zero-order Bessel function in equation (10) can be expanded using Hankel functions:

\[ J_0(\kappa r) = \frac{1}{2} \left[ H_0^{(1)}(\kappa r) + H_0^{(2)}(\kappa r) \right] \]  

(16)

The Hankel functions can be represented as an incoming and outgoing acoustic wave. The FFP is designed to model radially outgoing acoustic waves. This allows for the first Hankel function to be suppressed with the additional argument that the incoming acoustic wave will not contribute significantly to the final result. The asymptotic expansion of the second Hankel function for large arguments is as follows:

\[ H_0^{(2)}(\kappa r) \approx \frac{2}{\pi \kappa} \frac{e^{i(\kappa r - \pi/4)}}{\sqrt{r}} \quad \kappa r > 1 \]  

(17)

making the problem easier to handle. The important contributions from the integrand of the inverse transform, equation (10), comes from the area where \( \kappa \sim \kappa_\nu \). Substituting equation (17) into the inverse transform of equation (10) and taking the far-field approximation, the acoustic pressure equation can be written as

\[ p(r,z) \approx \frac{1 + i}{\sqrt{2\pi r}} \int_0^\infty \tilde{p}(\kappa,z) e^{i\kappa r} \sqrt{\kappa} \, d\kappa. \]  

(18)

To perform the calculation on a computer, the continuous integral must be replaced by a numerical integral over discrete values of \( \kappa \). Applying this to equation (18) yields

\[ p(r,z) = \frac{(1 + i)}{\sqrt{2\pi r}} \Delta \kappa \sum_{n=0}^{N-1} \tilde{p}(\kappa_n) \sqrt{\kappa_n} \ e^{-i \ (2\pi nm/N)} \]  

(19)
where

\[ \Delta \kappa = \frac{\kappa_{\text{max}}}{N - 1} \]
\[ \kappa_n = n \Delta \kappa \]
\[ m = \frac{r}{\Delta r} \]
\[ \Delta r = \frac{2 \pi}{N \Delta \kappa}. \]  \hspace{1cm} (20)

The term \( \kappa_{\text{max}} \) comes from the property of the integrand of equation (10) and only has significant contributions in a finite range of \( \kappa \), allowing the summation to be terminated at a finite number of terms.

A problem in the derivation is one of the complex numerical integrations. The function being integrated contains branch points and poles on the real axis. Because of the nature of branch points and poles, the integration being performed must not include any of these points for a correct solution. To avoid these problems, the current FFP uses what is called extra loss in the calculations. The mathematical result of using this extra loss is to move the numerical integration off the real axis. The effect of the extra loss is removed from the solution in an approximate manner by multiplying the computed pressure by the term \( \exp(\alpha r) \), where \( \alpha \) is the extra loss attenuation constant in Np/m. The proper choice of the artificial attenuation is essential if meaningful results are to be obtained from the code.

Another problem with the numerical integration is the number of points \( N \) used in the summation of equation (19). A lower bound on the number of points required in the summation is

\[ N_{\text{min}} = \frac{\kappa_{\text{max}} r}{\pi}. \] \hspace{1cm} [6] \hspace{1cm} (21)

However in most cases, this lower bound is too large to perform one single Fast Fourier Transform (FFT). The problem is if there are enough points to
sufficiently sample the wavenumber space in the numerical integration. If equation (19) is rewritten in the form

\[ S = \sum_{n=0}^{N-1} G_n e^{-i(2\pi n m/N)}, \]  

(22)

this summation [6] can be rewritten so that the single summation is rearranged to a double summation of the form

\[ S = \sum_{b=1}^{p} e^{-i(2\pi (b-1)m/p)} \sum_{n=0}^{N'-1} G_{n+(b-1)N'} e^{-i(2\pi n m/(pN'))} \]  

(23)

where

\[ N' = N/p \]
\[ p = \text{an integer larger than 1}. \]

p is the number of panels that the original FFT has been divided into. Each panel contains \( N' \) points. This technique allows the FFT to be performed in order to calculate the acoustic pressure with range at the height of the receiver.

### 2.3 Absorption of Sound in the Air

Losses in the medium are basically caused by viscosity, heat conduction, and molecular exchanges of energy. In the nineteenth century, only the mechanisms of viscosity and heat conduction were suspected of causing dissipation of sound; therefore, they are presently referred to as classical absorption.

In classical absorption, if the effect of absorption is represented by a factor \( e^{-\alpha r} \) where \( r \) is the distance of propagation, then the attenuation coefficient \( \alpha_d \) caused by viscosity and heat conduction is given by equation (17) from Physical Acoustics XVII [7]:

---

16
\[ \alpha_{cl} = 5.578 \times 10^{-9} \frac{T/T_o}{T + 110.4} \frac{f^2}{P/P_o}. \]  

The units of \( \alpha_{cl} \) in equation (24) are nepers/meter, where

- \( P_o \) = the reference pressure of \( 1.01325 \times 10^5 \) Np/m\(^2\) (1 atm)
- \( P \) = pressure in Np/m\(^2\)
- \( T_o \) = the reference temperature of 293.15 K
- \( T \) = temperature in K
- \( f \) = frequency (Hz)

In molecular absorption, energy exchanges at the molecular level include rotational and vibrational modes. Analysis of the rotational mode shows that the representative attenuation coefficient is proportional to \( \alpha_{cl} \), the classical attenuation coefficient:

\[ \frac{\alpha_{rot}}{\alpha_{cl}} = 4.16 e^{-16.8 T^{-1/3}} \]  

when 293 K < \( T \) < 690 K.

For frequencies below 10 MHz, it has been demonstrated that energy losses caused by classical and molecular absorption are additive.

\[ \alpha_{er} = \alpha_{cl} + \alpha_{rot}. \]  

A simplified empirical form of the equation can be written as

\[ \alpha_{er} = 1.83 \times 10^{-11} \frac{\sqrt{T/T_o} f^2}{P/P_o}, \]  

which is correct within 2 percent for 213 K < \( T \) < 373 K.
The vibrational mode of absorption should also be considered. Because the atmosphere is composed mostly of nitrogen and oxygen, each will contribute an attenuation coefficient, where \( j \) stands for either oxygen or nitrogen. The symbols are defined as follows:

\[
\alpha_{vib, j} = \frac{4pX_j}{35c} \left( \frac{q_j}{T} \right)^2 \frac{e^{-q_j/(Tf^2)}}{f_{r,j} + (f^2/f_{r,j})}
\]  

(28)

\( X_j \) = the mole fraction of air component considered (0.20948 for oxygen and 0.78084 for nitrogen)

\( q_j \) = the characteristic vibrational temperature (2239.1 K for oxygen and 3352.0 K for nitrogen)

\( c \) = the speed of sound at temperature \( T \) (m/s).

The \( f_{r,j} \) are the individual relaxation frequencies for oxygen and nitrogen. The computation of these frequencies depends on the relative humidity and atmospheric pressure. The relaxation frequencies are given by:

\[
f_{r,O} = \frac{P}{P_o} \left( 24 + 4.04 \times 10^4 h \frac{0.02 + h}{0.391 + h} \right)
\]

\[
f_{r,N} = \frac{P}{P_o} \sqrt{\frac{T_o}{T}} \left( 9 + 280 h e^{-4.170 (T_o/T)^{1/3} - 1} \right).
\]

The total attenuation coefficient is then the sum of \( \alpha_{cr} \) and \( \alpha_{vib,j} \). Figure 4 is a log-log plot [6] of total attenuation coefficients for \( T = 20 \) °C and Rh = 20 percent. Figure 4 shows the attenuation caused by classical absorption, vibrational relaxation of nitrogen and oxygen, and the total attenuation coefficient caused by the sum of the three attenuation mechanisms.

The attenuation coefficient \( \alpha \) is proportional to the square of the frequency. As the frequency doubles, the attenuation will quadruple. The attenuation of the sound wave caused by molecular and vibrational absorption is very important for frequencies over 250 Hz. For frequencies below 250 Hz, this attenuation does not contribute much to the total attenuation of the sound wave.
2.4 Complex Ground Impedance

There are several models available for calculating the complex ground impedance. The impedance model used in SCAFFIP is the Four Parameter Model of Attenborough. [8] In this impedance model, the complex normalized characteristic impedance of the ground is calculated using

\[
Z_c = \frac{4q^2 + i \frac{S^2 \alpha}{\omega \rho_o}}{3 \Omega k_b}
\]  

(30)
where

\[ q^2 = \Omega^\gamma' \]

- \( S_f \) = the pore shape factor ratio
- \( \Omega \) = the porosity of the ground
- \( \sigma \) = the flow resistivity of the ground (mks) rayls
- \( \omega \) = the angular frequency
- \( \rho_o \) = the density of air (1.2 kg/m\(^3\))
- \( k_b \) = the normalized wave number.

The normalized wave number \( k_b \) is computed from

\[
k_b = \sqrt{\gamma \Omega} \left[ \left( \frac{4}{3} - \frac{\gamma - 1}{\gamma N_{pr}} \right) \frac{q^2}{\Omega} + i \frac{S_f \sigma}{\omega \rho_o} \right]^{1/2}
\]  

where

\[ \gamma = \text{the ratio of specific heats (equation (2))} \]
\[ N_{pr} = \text{the Prandtl number (0.724)}. \]

The parameters \( S_f, \Omega, \rho, \) and \( n' \) are normally varied until agreement is reached between impedance measurements and the impedance model is achieved. However, this method of determining the four parameters cannot always be completed if time or resources are lacking. Table 2 provides rough estimates of the values for the four parameters, given some general descriptions of a variety of ground surfaces, to aid the user when the four parameters are unknown.

Some ground surfaces have a layered structure resulting from the gradual deposition of material over a soil base (a layer of snow over frozen ground, decomposition of organic material over clay or sandy soil, or a well plowed pasture over clay or harder packed soil.) An effective impedance \( Z(d) \) can be calculated for a semi-infinite layer of impedance \( Z_2 \) covered by a layer, depth \( d \), of another material of impedance \( Z_1 \). The effective impedance is given by
Z(d) = \left[ \frac{Z_2 - iZ_1 \tan(k_b d)}{Z_1 - iZ_2 \tan(k_b d)} \right] Z_1 \tag{32}

where

k_b = the bulk propagation constant in the top layer of the ground.

The parameters Z_1, Z_2, and k_b are calculated using equations (30) and (31) from Attenborough's Four Parameter impedance model.
Table 2. Estimation of $\Omega$ and $\sigma$ for various ground surfaces given $n' = 0.750$ and $S_r = 0.875$

<table>
<thead>
<tr>
<th>Description of Surface</th>
<th>$\Omega$</th>
<th>$\sigma$(cgs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dry snow, new fallen 0.1 m over about 0.4 m old snow</td>
<td>0.850</td>
<td>23</td>
</tr>
<tr>
<td>Sugar snow</td>
<td>0.825</td>
<td>48</td>
</tr>
<tr>
<td>In forest, pine, or hemlock</td>
<td>0.825</td>
<td>48</td>
</tr>
<tr>
<td>Grass: rough pasture, airport, public buildings, etc.</td>
<td>0.675</td>
<td>330</td>
</tr>
<tr>
<td>Roadside dirt, ill-defined, small rocks up to 0.1 m mesh</td>
<td>0.575</td>
<td>960</td>
</tr>
<tr>
<td>Sandy silt, hard-packed by vehicles</td>
<td>0.475</td>
<td>3470</td>
</tr>
<tr>
<td>&quot;Clean&quot; limestone chips, thick layer (0.01 to 0.025 m mesh)</td>
<td>0.425</td>
<td>6470</td>
</tr>
<tr>
<td>Old dirt roadway, fine stones (0.5 m mesh) interstices filled</td>
<td>0.400</td>
<td>7500</td>
</tr>
<tr>
<td>Earth, exposed and rain-packed</td>
<td>0.350</td>
<td>17100</td>
</tr>
<tr>
<td>Quarry dust, fine, very hard-packed</td>
<td>0.300</td>
<td>41700</td>
</tr>
<tr>
<td>Asphalt, sealed by dust and light use</td>
<td>0.250</td>
<td>120000</td>
</tr>
</tbody>
</table>
3. Operations

3.1 Purpose of User Interface

The user interface is designed to run the model with numerous scenarios and a minimum of modifications to the input files between each run. The user interface also allows the model to be run interactively or in a batch job format. This user interface will be standard for all of the benign terrain acoustic propagation models released by the U.S. Army Research Laboratory Battlefield Environment Directorate (ARL/BED). This allows for the same data files to be used for future benign terrain acoustic propagation models obtained from ARL/BED. If other data input formats are required, ARL/BED can assist with the modifications to the current user interface. Under most conditions, ARL/BED cannot guarantee compatibility with future upgrades and acoustic propagation models after the original user interface has been modified.

3.2 Format of Input Data Files

There are five input files required by SCAFFIP. The file names can be anything within 64 characters in length, file names must still conform to the limitations set by the computer system being used because SCAFFIP asks for the name of each file as it runs. Example input files are included with the discussion of test cases in appendix B. The five categories required for input are: (1) debug, (2) weather, (3) geometry, (4) frequency, and (5) ground. The syntax of each of the five files is given in this section:

3.2.1 Debug

The debug file contains an integer code that allows for input of additional parameters normally fixed within the program and output of additional files.
The format for the debug file follows:

Code:
- 0 Normal operation
- 1 Output speed of sound profile (sound.o)
- 2 Use radians for wind direction
- 4 Source centered calculation
- 8 Input extra loss (extra.los)
- 16 Input number of points and panels for performing FFT (trans.frm)
- 32 Output panel values (npan.e)
- 64 Output ground impedance (imped.e)
- 128 Output wave number spectrum (wavnum.e)
- 256 Disable smoothing

The value for code can be one of the numbers listed above or a sum of any two or more of the numbers listed above. If normal operation of the model is desired, use a zero in the debug file. If the speed of sound profile and wave number spectrum output is desired, the value in the debug file would be 129 = 1 + 128. Appendix C contains the file formats for the six debug files.

### 3.2.2 Weather

The weather file contains the atmospheric profile. The first line of the file is an integer. The rest of the file contains floating point numbers.

The format for the weather file follows:

\[
\begin{align*}
N \int & \\
Z(1) & T(1) & Rh(1) & P(1) & Wvel(1) & Wdir(1) \\
Z(2) & T(2) & Rh(2) & P(2) & Wvel(2) & Wdir(2) \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
Z(N \int) & T(N \int) & Rh(N \int) & P(N \int) & Wvel(N \int) & Wdir(N \int)
\end{align*}
\]
where

- \( N_{\text{int}} \) - number of interfaces. (There is no limit; however, many interfaces will slow the program down. Try and limit the number of interfaces to no more than 60.)
- \( Z(i) \) - height of the \( i \)th interface in meters, also \( Z(1) = 0 \) and \( Z(i) < Z(i+1) \) always.
- \( T(i) \) - temperature in Kelvin at height \( Z(i) \).
- \( \text{Rh}(i) \) - relative humidity in percent at height \( Z(i) \).
- \( P(i) \) - pressure in atmosphere at height \( Z(i) \).
- \( \text{Wvel}(i) \) - wind speed in meters per second at height \( Z(i) \).
- \( \text{Wdir}(i) \) - direction, in degrees, from which the wind is blowing relative to north at height \( Z(i) \).

### 3.2.3 Geometry

The geometry file contains the information describing the geometry of problem.

The format for the geometry file follows:

\[
Z_s \quad Z_r \quad \text{Rhz} \quad \text{Bear1} \quad \text{Bear2} \quad \text{Resolve}
\]

where

- \( Z_s \) - height of the source in meters.
- \( Z_r \) - height of the receiver in meters.
- \( \text{Rhz} \) - the horizontal distance between the source and receiver in meters.
- \( \text{Bear1} \) - the initial azimuth for scanning in degrees.
- \( \text{Bear2} \) - the final azimuth for scanning in degrees.
- \( \text{Resolve} \) - the angular resolution of the scan in degrees. The finer the angular resolution, the slower the model will run. This can be overcome by using a smaller angle to scan.
3.2.4 **Frequency**

The frequency file contains the information about the frequency to use for the model calculation.

The format for the frequency file follows:

\[ \text{Freq} \quad \text{dB}_{\text{src}} \quad \text{dB}_{\text{back}} \]

where

- **Freq** - frequency in Hertz.
- **dB_{src}** - level of the source at 1 meter in decibel.
- **dB_{back}** - level of the background noise at receiver in decibel.

If zero is used for \( \text{dB}_{\text{src}} \) and \( \text{dB}_{\text{back}} \), the output will be the relative sound pressure level instead of the sound pressure level.

3.2.5 **Ground**

The ground file contains the information to calculate the complex ground impedance. The enclosed table can be used to estimate these parameters.

The format for the ground file follows:

\[ \text{Nground} \quad \text{Sigma1} \quad \text{Om1} \quad \text{Pn1} \quad \text{Sf1} \]
\[ D \]
\[ \text{Sigma2} \quad \text{Om2} \quad \text{Pn2} \quad \text{Sf2} \]

where

- **Nground** - number of layers (1 or 2).
- **Sigma1** - flow resistivity in cgs units of the top layer.
- **Om** - porosity of the top layer.
- **Pn1** - grain shape factor of the top layer (normally use 0.75).
Sf1 - pore shape factor of the top layer (normally use 0.875).
D - depth of the top layer in meters.
Sigma2 - flow resistivity in cgs units of the bottom layer.
Om2 - porosity of the bottom layer.
Pn2 - grain shape factor of the bottom layer (normally use 0.75).
Sf2 - pore shape factor of the bottom layer (normally use 0.875).

Not all of the entries are required. If only one ground layer is used, set Nground = 1, then D, Sigma2, Om2, Pn2, and Sf2 may be omitted from the file. If Nground = 1, then values for D, Sigma2, Om2, Pn2, and Sf2 may be present in the file; however, the program will not read any of these values.

3.3 Explanation of User Interface

A better heading for this section may be the philosophy of the user interface. As mentioned at the beginning of section 3, the user interface is designed to allow easier model running with numerous scenarios and a minimum number of modifications to the input files. This philosophy will be applied to the series of models being adapted by ARL/BED's Acoustics Team. The current user interface is being implemented on all the flat-earth non-turbulent acoustic propagation models currently in the ARL/BED repertoire. As the newer models come out with more complex interactions, a new interface will be developed to incorporate the new inputs required to execute the new models. The overall idea behind the user interfaces will be maintained. The normal operation of the models will require only the basic input requirements. For more advanced users, there are debug flags available to provide for additional input parameters, changing of default parameters, and output of additional information for the model. This philosophy provides a user interface that is easy to use, yet allows for detailed scientific studies to be conducted.

3.4 How To Run SCAFFIP

SCAFFIP is relatively easy to run. There is a C-shell script included with the source code to compile the executable on a Unix workstation. To run the model, type sffp at the command line. The program will ask for the name of
the input files mentioned in section 3.2. The printout labeled output shows what will be seen when the model is run.

3.5 Format of the Output

The output from the model is saved to a file named scand. The format of the output file follows

Az1  
N1   
R(1) dB(1) 
R(2) dB(2)  
:    :    
R(N1) dB(N1) 
Az2  
N2   
R(1) dB(1) 
R(2) dB(2)  
:    :    
R(N2) dB(N2) 
Az3  
N3   
R(1) dB(1) 
R(2) dB(2)  
:    :    
R(N3) dB(N3)  

where

Az   - azimuth of calculation in radians.  
N    - number of range points.  
R(i) - range in meters.  
dB(i) - sound level or relative sound level at R in decibels.
References


<table>
<thead>
<tr>
<th>Acronyms and Abbreviations</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ARL</strong></td>
</tr>
<tr>
<td><strong>BED</strong></td>
</tr>
<tr>
<td><strong>CPIO</strong></td>
</tr>
<tr>
<td><strong>FFP</strong></td>
</tr>
<tr>
<td><strong>FFT</strong></td>
</tr>
<tr>
<td><strong>SCAFFIP</strong></td>
</tr>
<tr>
<td><strong>TAR</strong></td>
</tr>
</tbody>
</table>
Appendix A
Source Code For SCAFFIP
General Version 1.0
ADMIT.C

/*
  * FINDS ADMITTANCE OF ONE BRANCH OF THE TRANSMISSION LINE
  */

#include "complex.h"
#include "ffp.h"

void admit(admittance, gp, last, layer, voltage)
complex *admittance, *voltage;
int last;
struct gp *gp;
struct layer *layer;
{
    complex phase, ref_1, ref_2, unit, z, z1, z2;
    int i2l, index, step, toggle = 0;

    /*
    * direction to source from last (+1 = = up, -1 = = down)
    */
    step = gp->index_source >= last ? 1 : -1;

    /*
    * from interface to section beyond interface (+1 = = up, 0 = = down)
    */
    i2l = (step + 1) / 2;

    /*
    * admittance of last significant line
    */
    Re(*admittance) = Re(layer[last].yO);
    Im(*admittance) = Im(layer[last].yO);

    /*
    * step through network to section in front of source
    */
    Re(unit) = 1;
    Im(unit) = 0;
    for (index = last + step; index <= gp->index_source && step == 1
        || index > gp->index_source && step == -1; index += step)
    {
        /*
        * set toggle and voltage if beyond the receiver
        */
        if (index == gp->index_detector + i2l)
    {
toggle = 1;
Re(*voltage) = 1;
Im(*voltage) = 0;
}

*/

* electrical phase factor (gamma = j * kz)
*/

Re(z) = 0;
Im(z) = -layer[index].thickness;
cmul(z, z, layer[index].kz);
cexp(phase, z);

/*

* reflection coefficient at the end of section
*/
csub(zl, layer[index].y0, *admittance);
cadd(z2, layer[index].y0, *admittance);
cdiv(ref_l, zl, z2);

/*

* reflection coefficient at the beginning of section
*/
cmul(z, phase, phase);
cmul(ref_2, ref_1, z);

/*

* total voltage at the beginning of section
*/
if (toggle)
{
cadd(z1, unit, ref_2);
cmul(z1, *voltage, z1);
cadd(z2, unit, ref_1);
cmul(z2, phase, z2);
cdiv(*voltage, z1, z2);
}

/*

* admittance at the beginning of section
*/
csub(z1, unit, ref_2);
cmul(z1, layer[index].y0, z1);
cadd(z2, unit, ref_2);
cdiv(*admittance, z1, z2);

}
AIR.C

/*
 * AIR ABSORPTION ROUTINE
 *
 * (1) ANSI STANDARD S1.26-198X
 * (2) H. E. Bass et al,
 * Absorption of Sound by the Atmosphere,
 */

#include "complex.h"

/*
 * reference ambient temperature
 * 293.16 degrees Kelvin == 20 degrees Celsius
 */
#define T20 293.15

double air_absorption(frequency, kelvin, pressure, relative_humidity)
  double frequency, kelvin, pressure, relative_humidity;
  {
    double alpha, humidity, nrf, orf, saturation, square, temp;

    temp = T20 / kelvin;
    /
    * saturation pressure
    * reference 2, page 169, equation 72
    */
    saturation = pow(10.0, 8.422 - 10.05916 * temp + 5.023 * log10(temp));
    /
    * percent mole fraction of water vapor
    * reference 1, page 19, equation D10
    */
    humidity = relative_humidity * saturation / pressure;
    /
    * oxygen relaxation frequency
    * reference 1, page 7, equation 9
    */
    orf = pressure * (24 + 40400 * humidity
      * (humidity + 0.02) / (humidity + 0.391));
    /
    * nitrogen relaxation frequency
    * reference 1, page 7, equation 10
  }
/*  
nrf = pressure * sqrt(temp)  
   * (9 + 280 * humidity * exp(4.17 * (1 - pow(temp, 0.3333))));  
/*  
* absorption coefficient alpha (nepers/m)  
* reference 1, page 7, equation 8  
*/  
square = frequency * frequency;  
alpha = 0.01275 * exp(-2239.1 / kelvin) / (orf + square / orf);  
alpha += 0.1068 * exp(-3352.0 / kelvin) / (nrf + square / nrf);  
alpha = square  
   * (1.84e-11 / sqrt(temp) / pressure + alpha * pow(temp, 2.5));  
return alpha;  
}
CLIP.C

/*
 * SIMPLIFY TRANSMISSION LINE BY CLIPPING TECHNIQUE OF Lee, et al
 */

#include "complex.h"
#include "ffp.h"

/*
 * exp(kzmax) == le8
 */
#define kzmax 18.42068074395236547214

void parameters();

void clip(gp, k, last, layer, load, n_layers, omega)
double k, omega;
int *last, load, n_layers;
struct gp *gp;
struct layer *layer;
{
    double decay = 0;
    int i2l, index, step, toggle;

    *last = -1;
    /*
     * set toggle if detector is not between load and source
     */
    toggle = gp->index_detector > gp->index_source
        ? gp->index_source > load : gp->index_source <= load;
    /*
     * direction to load from source (+1 == up, -1 == down)
     */
    step = gp->index_source <= load ? 1 : -1;
    /*
     * conversion from interface to opposite layer (+1 == up, 0 == down)
     */
    i2l = (step + 1) / 2;
    /*
     * step from source to layer nearest load
     */
    for (index = gp->index_source + i2l; index < load && step == 1
        || index > load && step == -1; index += step)
    {
        /*
         * find admittance and wave number for section
         */
    }
parameters(gp, k, index, layer, n_layers, omega);

* sum the exponential decay coefficients
* /

decay -= layer[index].thickness * Im(layer[index].kz);

/*
* check if beyond the detector
* /

if (toggle)
{
    /*
    * replace with infinite section if decay too large
    */
    if (decay >= kzmax / 2)
    {
        *last = index;
        return;
    }
}
else
{
    /*
    * v == 0 if decay too large
    */
    if (decay >= kzmax)
    return;

    /*
    * set toggle if detector is reached
    */
    if (index + step == gp->index_detector + i2l)
    toggle = 1;
}

/*
* find parameters for the load layer
*/
parameters(gp, k, load, layer, n_layers, omega);

/*
* last significant layer in the network
*/
*last = load;
#include <stdio.h>
#include <stddef.h>
#include "complex.h"
#include "ffp.h"

#define PANEL_VALUES dflags[5]

extern int errno;

void cutoff(delta_k, delta_r, frequency, gp, k_max, k_width, layer, n_panels,
omega, points, range, dflags)
double *delta_k, *delta_r, frequency, *k_max, *k_width, omega, range;
int *n_panels, points, *dflags;
struct gp *gp;
struct layer *layer;
{
    double c_min, difference = 0, gamma;
    int index, max, min, nyquist;
    FILE *fp_pv;

    /*
     * highest and lowest of layers between detector and source
     */
    max = gp->index_detector > gp->index_source
        ? gp->index_detector : gp->index_source;
    min = gp->index_detector + gp->index_source - max + 1;
    /*
     * minimum speed from source to detector
     */
    c_min = layer[min].c;
    for (index = min; index <= max; index++)
    {
        if (c_min > layer[index].c)
            c_min = layer[index].c;
    }
    /*
     * total altitude difference between detector and source
     */
    difference += layer[index].thickness;
}

/*
 * find upper cutoff wave number kmax based on an empirical relationship
 */
gamma = 0.0075 * frequency + (2.5 - 6.25e-4 * frequency) / difference;

*k_max = sqrt(gamma * gamma + (omega / c_min) * (omega / c_min));

/*
 * determine the number of panels
 */

nyquist = *k_max * range / points / PI + 1;
if (*n_panels == 0)
    *n_panels = nyquist;
if (*n_panels < 0)
    *n_panels *= -nyquist;

/*
 * index of the point nearest range based on Nyquist criteria
 */

index = *k_max * range / *n_panels / TWO_PI + 1.5;

/*
 * adjust range separations so that the index point is at range
 */

*delta_r = range / (index - 1);

/*
 * adjust upper cutoff wave number according to adjusted delta_r
 */

*k_max = TWO_PI * *n_panels / *delta_r;

/*
 * wave number band width per panel
 */

*k_width = *k_max / *n_panels;
*delta_k = *k_width / points;

/*
 * Write Out Panel Values
 */

if (PANEL_VALUES)
{
    if ((fp_pv = fopen("npan.e","a")) == NULL)
    {
        fprintf(stderr,"error number %d\n",errno);
        perror("npan.e");
        exit(1);
    }
    fprintf(fp_pv,"Krnax = %10.41f\tNyq = %5d\tNpan = %5d\n", *k_max, nyquist, *n_panels);
    fprintf(fp_pv,"Kwidth = %10.41f\tDelR = %10.41f\tDelK = %10.41f\n", *k_width, *delta_r, *delta_k);
    fclose(fp_pv);
}

#include <stddef.h>
#include <stdio.h>
#include <stdlib.h>
#include "complex.h"
#include "ffp.h"

#define N 2048
#define SQ(X) (X) * (X)
#define WAVESPECT dflags[7]
#define SMOOTH dflags[8]

void cutoff();
void fourier();
void profile();
void setup();
void voltage();
void wavenumber();
void zeffective();
extern int errno;

double propmod(a, fd, fp, frequency, src_level, back_level, geometry, ground,
        n_ground_layers, n_interfaces, dflags)
FILE *fp;
double frequency, src_level, back_level;
int fd, n_ground_layers, n_interfaces, *dflags;
struct geometry *geometry;
struct ground *ground;
struct interface *a;
{
    complex amplitude, field[N], impedance, pressure[N], z, z0, z1;
    double delta_k, delta_r, detector, gain[N], k, k_max, k_min, k_width,
           omega, r, range, source, temp, extra;
    float buffer[2];
    int i, j, limit, n_layers, n_panels, points;
    struct gp gp;
    struct layer *layer;
    FILE *fp_wspc;
/* Preliminary Setup */
detector = geometry->Zr;
omega = TWO_PI * frequency;
range = geometry->range;
source = geometry->Zs;

/* Setup The Points and Panels For Transform */
setup(&n_panels, &points, &extra, dflags);

/* Impedance and Wavenumber of Half-Spaces #1 and #N */
zeffective(a[l].c, frequency, ground, &impedance, &gp.k_1,
        n_ground_layers, dflags);
Re(gp.impedance_1) = Re(impedance);
Im(gp.impedance_1) = -Im(impedance);
Im(gp.k_1) = -Im(gp.k_1);
Re(gp.impedance_n) = Re(gp.k_n) = 1;
Im(gp.impedance_n) = Im(gp.k_n) = 0;

/* Initialize The Profile */
layer = (struct layer *) calloc((size_t) n_interfaces + 3,
sizeof(struct layer));
profile(a, detector, frequency, &gp, layer, n_interfaces, &n_layers,
source);

/* Squares of Intrinsic Wave Numbers Within Each Layer */
wavenumber(extra, &gp, layer, n_layers, omega);

/* Upper Cut-Off Wave Number */
cutoff(&delta_k, &delta_r, frequency, &gp, &k_max, &k_width, layer,
        &n_panels, omega, points, range, dflags);

/* Zero The Pressure Array */
for (i = 1; i <= points; i++)
    Re(pressure[i]) = Im(pressure[i]) = 0;

/* \[ z_0 = \frac{1 - i}{2 \sqrt{\pi}} \] */

43
Re(z0) = ONE_2SQRTPI; Im(z0) = -ONE_2SQRTPI;

* Overlap And Add Field From Each Panel *

for (i = 1; i <= n_panels; i++)
{
    /* Starting Wave Number of Each Panel */
    k = k_min = (i - 1) * k_width;
    /* Find The Pressure Field in The Transform Domain */
    for (j = 1; j <= points; j++)
    {
        /* Pressure Amplitude */
        voltage(&amplitude, &gp, k, layer, n_layers, omega);
        cmul(field[j], amplitude, zO);
        temp = sqrt(k + 1e-20);
        Re(field[j]) /= temp;
        Im(field[j]) /= -temp;
    }
    /* Write Out Horizontal Wave Number Spectrum */
    if (WAVESPECT)
    {
        if ((fp_wspc = fopen("wavnum.e","a")) == NULL)
            {fprintf(stderr,"error number %d\n", errno);
                perror("wavnum.e");
                exit(1);}
        fprintf(fp_wspc,"%10.4f %10.4f %10.4f
", k,cmod(amplitude),
                Re(amplitude),Im(amplitude));
        fclose(fp_wspc);
    }
    k += delta_k;
Fourier Transform The Wave Number Amplitudes

fourier(field, points, 1);

Retrieve The Incremental Pressure Amplitudes

\[
\begin{align*}
r &= \text{delta}_r; \\
&\text{for (j = 2; j <= points; j + +)} \\
&\{ \\
&\quad \text{Im(field}[j]\text{]} = -\text{Im(field}[j]\text{]}; \\
&\quad \text{Re(zl)} = \text{extra} \ast \sqrt{\text{SQ}(r) + \text{SQ(source - detector})}; \\
&\quad \text{Im(zl)} = -k_{\text{min}} \ast r; \\
&\quad \text{cexp(z, zl)}; \\
&\quad \text{cmul(amplitude, field}[j], zl); \\
&\quad \text{temp} = \text{delta}_k \div \text{sqrt(r)}; \\
&\quad \text{Re(amplitude)} *= \text{temp}; \\
&\quad \text{Im(amplitude)} *= \text{temp}; \\
&\}\end{align*}
\]

Accumulate The Total Pressure in Pressure[j]

\[
\begin{align*}
&\text{Re(pressure}[j]\text{]} += \text{Re(amplitude)}; \\
&\text{Im(pressure}[j]\text{]} += \text{Im(amplitude)}; \\
&\text{if (i == n_panels && r > range)} \\
&\quad \text{break}; \\
&\quad r += \text{delta}_r; \\
&\}\end{align*}
\]

limit = j - 2; 
if (fp != NULL)
\{
  \text{fprintf(fp, "\%d\n", limit);} 
\}
else
\{
  \text{buffer[0] = limit}; 
  \text{write(fd, (char *) buffer, sizeof(float))}; 
\}
limit++; 
\text{r = delta_r}; 
\text{for (i = 2; i <= limit; i + +)}
\{
/*
Find Gain Relative To 0 dB at 1 meter (Transmission Loss)

\[
\text{gain} = 20 \times \log_{10}(\text{cm}(\text{pressure}[i]));
\]

if (! SMOOTH)
{

/*
  * Perform Smoothing Algorithm On Points
  */
  smooth(gain, limit, dflags);
}

r = delta_r;
for (i = 2; i <= limit; i++)
{

/*
  * Write Out The Levels Versus Range
  */
  gain[i] = src_level + gain[i];
  if ((gain[i] < back_level) && (src_level > 0))
    gain[i] = back_level;
  if (fp != NULL)
    {
      fprintf(fp, "%13lf %13lf\n", r, gain[i]);
    }
  else
    {
      buffer[0] = r;
      buffer[1] = gain[i];
      write(fd, (char *) buffer, sizeof(double));
    }
  r += delta_r;
}

return gain[limit];
}
/*
 * Replaces data by its discrete Fourier transform if sign == 1 or
 * by n times its inverse discrete Fourier transform if sign == -1.
 * data is a complex array of length n.
 * n must be an integral power of 2 (this is not checked for!).
 *
 * W. H. Press et al,
 */

#include "complex.h"

void fourier(data, n, sign)
complex *data;
int n, sign;
{
    complex w, wp, ztemp;
    double temp, theta;
    int index, j = 1, m, max, step;

    /*
     * bit-reversal section
     */
    for (index = 1; index <= n; index++)
    {
        if (j > index)
        {
            Re(ztemp) = Re(data[j]);
            Im(ztemp) = Im(data[j]);
            Re(data[j]) = Re(data[index]);
            Im(data[j]) = Im(data[index]);
            Re(data[index]) = Re(ztemp);
            Im(data[index]) = Im(ztemp);
        }
        m = n / 2;
        while (m >= 2 && m < j)
        {
            j -= m;
            m /= 2;
        }
    }

    // rest of the code...
}
/*
 * beginning of Danielson-Lanczos section
 */

max = 1;
while (max < n)
    { /* while loop executed log2(n) times */

    /*
    * initialize for trigonometric recurrence
    */
    step = 2 * max;
    theta = PI / (max * sign);
    temp = sin(theta / 2);
    Re(wp) = -2 * temp * temp;
    Im(wp) = sin(theta);
    Re(w) = 1;
    Im(w) = 0;

    /*
    * two nested inner loops
    */
    for (m = 1; m <= max; m++)
        { /*
        * Danielson-Lanczos formula
        */
            j = index + max;
            cmul(ztemp, data[j], w);
            csub(data[j], data[index], ztemp);
            cadd(data[index], data[index], ztemp);
        }

    /*
    * trigonometric recurrence
    */
    cmul(ztemp, w, wp);
    cadd(w, w, ztemp);
    max = step;
    }
}
HEADER.C

#include "ffp.h"
#include "complex.h"

/*
 * print a line of 80 dashes
 */
define dash\n    printf("-----------------------------");\n    printf("-----------------------------\n")
define RAD_TO_DEG(x)   (180. * x) / PI

void header(a, frequency, geometry, ground, n_ground_layers, n_interfaces)
    double frequency;
    int n_ground_layers, n_interfaces;
    struct geometry *geometry;
    struct ground *ground;
    struct interface *a;
{
    int index;

    printf("\n\n\nSINGLE FREQUENCIES\n");
dash;
    printf("RECEIVER HEIGHT (Zr) = %7.1lf METERS      SOURCE HEIGHT (Zs) = %8.1lf METERS\n", geometry->Zr, geometry->Zs);
dash;
    printf("RANGE = %7.1lf METERS      BEARING OF SOURCE = %lf DEGREES\n", geometry->range, RAD_TO_DEG(geometry->theta_l));
dash;
    printf("FREQUENCY = %7d Hz\n", (int) frequency);
dash;
    printf(" KELVIN RELATIVE ATMOSPHERIC WIND\n");
    printf("INTERFACE HEIGHT TEMPERATURE HUMIDITY PRESSURE SPEED\n");
    printf(" DIRECTION\n");

    for (index = n_interfaces; index > 0; index--)
        printf("%6d%13.2lf%13.2lf%12.1lf%13.3lf%10.3lf%11.3lf\n", 
            index, a[index].z, a[index].kelvin, 
            a[index].relative_humidity, a[index].pressure, 
            a[index].wind_speed, a[index].wind_theta);
dash;
    if (n_ground_layers == 1)
    {
printf("GROUND LAYER      SIGMA      POROSITY\n");
printf("    %15.2lf%13.2lf\n",
     ground->sigma_l, ground->omega_l);
}
else
{
    printf("GROUND LAYER      SIGMA      POROSITY      DEPTH\n");
    printf("    %15.2lf%13.2lf%10.3lf\n",
     ground->sigma_1, ground->omega_1, ground->depth);
    printf("     %15.2lf%13.2lf\n",
     ground->sigma_2, ground->omega_2);
}
dash;
}
#include <stddef.h>
#include <stdio.h>
#include <stdlib.h>
#include "ffp.h"
#include "complex.h"

#define MET_FORM  dflags[l]
#define openfile(NAME)
    printf("Input name of NAME file:\n");
    scanf("%s", filename);
    printf("%s\n", filename);
    if ((fp = fopen(filename, "r")) == NULL) {
        fprintf(stderr, "error number %d\n", errno);
        perror(filename);
        exit(1);
    }
#define DEG_TO_RAD(x)  (PI * x) / 180.
extern int errno;

void input(a, dflags, frequency, src_level, back_level, geometry, ground,
    n_ground_layers, n_interfaces,n_freq)
int *n_ground_layers, *n_interfaces, *n_freq, *dflags;
double *frequency, *src_level, *back_level;
struct geometry *geometry;
struct ground *ground;
struct interface **a;
{
    FILE *fp;
    char filename[81];
    int code, index;
    double wind_theta, azimuth1, azimuth2, dazimuth;
    
    /*
     * Read Debug File
     */
    openfile(debug);
    fscanf(fp, "%d", &code);
    fclose(fp);
    parser(code, dflags);
    
    /*
     * Read Weather File
     */
    
}
openfile(weather);
fsscanf(fp, "%d", n_interfaces);
*a = (struct interface *) calloc((size_t) *n_interfaces + 1,
sizeof(struct interface));
if (MET_FORM)
{
    /*
    * Wind Direction in Radians
    */
    for (index = 1; index <= *n_interfaces; index++)
    {
        fscanf(fp, "%lf %lf %lf %lf %lf\n", &(*a)[index].z, &(*a)[index].kelvin,
                &(*a)[index].relative_humidity,
                &(*a)[index].pressure, &(*a)[index].wind_speed,
                &(*a)[index].wind_theta);
    }
}
else
{
    /*
    * Wind Direction in Degrees
    */
    for (index = 1; index <= *n_interfaces; index++)
    {
        fscanf(fp, "%lf %lf %lf %lf %lf\n", &(*a)[index].z, &(*a)[index].kelvin,
                &(*a)[index].relative_humidity,
                &(*a)[index].pressure, &(*a)[index].wind_speed, &wind_theta);
        (*a)[index].wind_theta = DEG_TO_RAD(wind_theta);
    }
}
fclose(fp);
/*
 * Read Geometry File
 */
openfile(geometry);
fsscanf(fp, "%lf %lf %lf %lf %lf %lf\n", &(geometry->Zs), &(geometry->Zr), &(geometry->range),
&azimuth1, &azimuth2, &dazimuth);
geometry->theta_1 = DEG_TO_RAD(azimuth1);
geometry->theta_2 = DEG_TO_RAD(azimuth2);
geometry->delta_theta = DEG_TO_RAD(dazimuth);
fclose(fp);
/*
 * Read Frequency File
 */
openfile(frequency);
/*
 * Src_level and Back_level is local to input.c
 */
fscanf(fp, "%lf %lf %lf", frequency, src_level, back_level);
fclose(fp);
/*
 * Read Ground File
 */
openfile(ground);
fscanf(fp, "%d %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %lf %l
/*
  * insert a new layer with top interface height == height
  */

#include "ffp.h"

void insert(height, index, layer, n_layers, z)
double height, *z;
int *index, *n_layers;
struct layer *layer;
{
  /*
  * find the index of height in z if it exists
  */
  for (*index = 1; *index < *n_layers; ++*index)
    if (z[*index] == height)
      /*
      * return the index of z that yields height
      */
      return;
  /*
  * shift array z up to open a slot for height
  */
  for (*index = *n_layers; *index > 1; --*index)
  {
    layer[*index].c = layer[*index-1].c;
    layer[*index].mu = layer[*index-1].mu;
    layer[*index].rho = layer[*index-1].rho;
    z[*index] = z[*index-1];
  }
  /*
  * check if height belongs above the next z
  */
  if (height > z[*index-1])
    break;
}
/*
  * extend the array length and insert height
  */
  + *n_layers;
  z[*index] = height;
#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include "complex.h"
#include "ffp.h"

#define FLAGS       0_WRONLY | O_CREAT | O_TRUNC
#define MODE 0644     /* this is the file mode for a created scan.bin */
#define RAD_TO_DEG(x) (180. * x) / PI

extern int errno;

double propmodO;
void header();
void input();
void parser();
void sound();

main(argc, argv)
int argc;
char **argv;
{
    FILE *fp = NULL;
    char *id = "@(#)December 1991 version - Frank W. Owens";
    double attenuation, frequency, src_level, back_level, theta;
    float buffer;
    int fd, index, n_ground_layers, n_interfaces, dflags[15];
    struct geometry geometry;
struct ground ground;
struct interface *a;
if (argc == 1)
{
    if ((fp = fopen("scand", "w")) == NULL)
    {
        fprintf(stderr, "error number %d\n", errno);
        perror("scan");
        exit(1);
    }
}
else if (argc == 2)
{
    if ((fd = open("scand.bin", FLAGS, MODE)) < 0)
    {
        fprintf(stderr, "error number %d\n", errno);
        perror("scan.bin");
        exit(1);
    }
}
else
{
    fprintf(stderr, "usage: %s [b]\n", *argv);
    exit(2);
}
input(&a, dflags, &frequency, &src_level, &back_level, &geometry, &ground, &n_ground_layers, &n_interfaces);
header(a, frequency, &geometry, &ground, n_ground_layers, n_interfaces);

/*
 * Begin Scanning Through The Azimuths
 */
for (theta = geometry.theta_l; theta <= geometry.theta_2; theta += geometry.delta_theta)
{
    if (argc == 1)
    {
        fprintf(fp, "%lf\n", theta);
    }
    else
    {
        buffer = theta;
        write(fd, (char *) &buffer, sizeof(float));
    }
    sound(a, n_interfaces, theta, dflags);
attenuation = propmod(a, fd, fp, frequency, src_level, 
    back_level, &geometry, &ground, n_ground_layers, 
    n_interfaces, dflags);
printf("Position Angle = %lf\n", RAD_TO_DEG(theta));
printf("%8.21f   %9.41f\n", geometry.range, attenuation);
}
if (argc == 1)
    fclose(fp);
else
    close(fd);
    printf("-------------------------------\n");
    printf("-------------------------------\n");
    return 0;
}
# include "complex.h"
# include "ffp.h"

void parameters(gp, k, l_index, layer, n_layers, omega)
    double k, omega;
    int l_index, n_layers;
    struct gp *gp;
    struct layer *layer;
    {
        complex z;
        /*
        * electrical wave number kz = = - j * gamma
        */
        Re(z) = Re(layer[l_index].ki2) - k * k;
        Im(z) = Im(layer[l_index].ki2);
        csqrt(layer[l_index].kz, z);
        /*
        * Re(gamma) is positive for the correct branch
        */
        if (Im(layer[l_index].kz) > 0)
        {
            Re(layer[l_index].kz) = -Re(layer[l_index].kz);
            Im(layer[l_index].kz) = -Im(layer[l_index].kz);
        }
        /*
        * electrical characteristic admittance
        */
        Re(layer[l_index].yO) =
            Re(layer[l_index].kz) / (omega * layer[l_index].rho);
        Im(layer[l_index].yO) =
            Im(layer[l_index].kz) / (omega * layer[l_index].rho);
        /*
        * admittance for half-spaces
        */
        if (l_index == 1)
        {
            cmul(z, gp->impedance_1, gp->k_l);
            cdiv(layer[l_index].yO, layer[l_index].yO, z);
        }
if (l_index == n_layers)
{
    cmul(z, gp->impedance_n, gp->k_n);
    cdiv(layer[l_index].y0, layer[l_index].y0, z);
}
#include <stdio.h>

void parser(code, dflags)
int code, *dflags;
{
    /*
     * DFLAGS(0) - 2^0 = 1
     */
    dflags[0] = (1 & code);
    /*
     * DFLAGS(1) - 2^1 = 2
     */
    dflags[1] = (2 & code) / 2;
    /*
     * DFLAGS(2) - 2^2 = 4
     */
    dflags[2] = (4 & code) / 4;
    /*
     * DFLAGS(3) - 2^3 = 8
     */
    dflags[3] = (8 & code) / 8;
    /*
     * DFLAGS(4) - 2^4 = 16
     */
    dflags[4] = (16 & code) / 16;
    /*
     * DFLAGS(5) - 2^5 = 32
     */
    dflags[5] = (32 & code) / 32;
    /*
     * DFLAGS(6) - 2^6 = 64
     */
    dflags[6] = (64 & code) / 64;
    /*
     * DFLAGS(7) - 2^7 = 128
     */
    dflags[7] = (128 & code) / 128;
    /*
     * DFLAGS(8) - 2^8 = 256
     */
    dflags[8] = (256 & code) / 256;
}
DFLAGS(9) - $2^9 = 512$

```
dflags[9] = (512 & code) / 512;
```

DFLAGS(10) - $2^{10} = 1024$

```
dflags[10] = (1024 & code) / 1024;
```

DFLAGS(11) - $2^{11} = 2048$

```
```

DFLAGS(12) - $2^{12} = 4096$

```
dflags[12] = (4096 & code) / 4096;
```

DFLAGS(13) - $2^{13} = 8192$

```
dflags[13] = (8192 & code) / 8192;
```

DFLAGS(14) - $2^{14} = 16384$

```
dflags[14] = (16384 & code) / 16384;
```
PROFILE.C

/*
 * INITIALIZE PROBLEM ATMOSPHERIC PROFILE
 */

#include <stddef.h>
#include <stdlib.h>
#include "ffp.h"

double air_absorption();
void insert();

void profile(a, detector, frequency, gp, layer, n_interfaces, n_layers, source)
double detector, frequency, source;
int n_interfaces, *n_layers;
struct interface *a;
struct gp *gp;
struct layer *layer;
{
    double *z;
    int index;

    /*
     * initialize the problem profile (z increases)
     */
    *z = (double *) calloc((size_t) n_interfaces + 3, sizeof(double));
    for (index = 1; index <= n_interfaces; index++)
    {
        layer[index].c = a[index].c;
        layer[index].mu = air_absorption(frequency, a[index].kelvin,
                                        a[index].pressure, a[index].relative_humidity);
        layer[index].rho = 1.2;
        z[index] = a[index].z;
    }
    *n_layers = n_interfaces;

    /*
     * insert detector and source interfaces (lowest first)
     */
    if (source < detector)
    {
        insert(source, &gp->index_source, layer, n_layers, z);
        insert(detector, &gp->index_detector, layer, n_layers, z);
    }
else
{ insert(detector, &gp->index_detector, layer, n_layers, z);
    insert(source, &gp->index_source, layer, n_layers, z);
}

/*
 * convert interface heights to layer thicknesses
 */
for (index = 2; index < *n_layers; index++)
    layer[index].thickness = zf[index] - z[index-1];
#include <stddef.h>
#include <stdio.h>
#include <stdlib.h>

#define DEFAULT_EXTRA_LOSS 1e-4
#define EXTRA_LOSS_FLAG dflags[3]
#define POINTS 1024
#define N_PANELS -2
#define TRANSFORM_FLAG dflags[4]

extern int errno;

void setup(n_panels, points, extra, dflags)
    double *extra;
    int *n_panels, *points, *dflags;
{
    FILE *fp;
    char filename[81];

    /*
     * Extra Loss
     */
    if (EXTRA_LOSS_FLAG)
    {
        if ((fp = fopen("extra.los", "r")) == NULL)
        {
            fprintf(stderr, "error number %d\n", errno);
            perror("extra.los");
            exit(1);
        }
        fscanf(fp, "%lf", extra);
        fclose(fp);
    }
    else
        *extra = DEFAULT_EXTRA_LOSS;

    /*
     * Number of Points and Panels
     */
    if (TRANSFORM_FLAG)
    {
        if ((fp = fopen("trans.frm", "r")) == NULL)
        {
            fprintf(stderr, "error number %d\n", errno);
            perror("trans.frm");
            exit(1);
        }
        fscanf(fp, "%d %d", n_panels, points);
        fclose(fp);
    }
perror("trans.frm");
ext(1);

}
fscanf(fp, "%d %d", points, n_panels);
fclose(fp);

} else
{
    *points = POINTS;
    *n_panels = N_PANELS;
}
SMOOTH.C

#include <stdio.h>

#define N 1024

void smooth(gain,limit,dflags)
double *gain;
int limit, *dflags;
{
  int loop, i, npt;
  double tmp[N], avg;
  /*
   * Copy Data to Working Array
   */
  for (loop = 0; loop <= limit; loop++)
    tmp[loop] = gain[loop];
  /*
   * Smooth The First Five Points
   */
  npt = 3;
  for (loop = 3; loop < 7; loop++)
    {
      avg = 0.;
      for (i = -(npt-1)/2; i <= (npt-1)/2; i++)
        avg += tmp[loop+i];
      /*
       * Replace Gain at R[loop] By 3, 5, 7 or 9 Point Average
       */
      gain[loop] = avg / npt;
      npt +=2;
    }
  /*
   * 11 Point Average For Smoothing
   */
  for (loop = 7; loop <= limit-5; loop++)
    {
      avg = 0.;
      for (i = -5; i <= 5; i++)
        avg += tmp[loop+i];
      /*
       * Replace Gain at R[loop] By 11 Point Average
       */
      gain[loop] = avg / 11.;
    }
}
/*
 * Smooth The Last Five Points
 */

npt = 9;
for (loop = limit-4; loop <= limit; loop++)
{
    avg = 0.;
    for (i = -(npt-1)/2; i <= (npt-1)/2; i++)
        avg += tmp[loop+i];
    /*
     * Replace Gain at R[loop] By 3, 5, 7 or 9 Point Average
     */
    gain[loop] = avg / npt;
    npt -= 2;
}
/*
 * calculate the sound speed profile
 */

#include <stdio.h>
#include "complex.h"
#include "ffp.h"

#define T0 273.16  /* Kelvin freezing point of H2O */
#define R0 8314.16 /* universal gas constant */
#define SOUND_SPEED_OUT_FLAG dflags[0]
#define RECIPROCITY_FLAG dflags[2]

extern int errno;

void sound(a, n_interfaces, bearing, dflags)
    double bearing;
    int n_interfaces, *dflags;
    struct interface *a;
{
    FILE *fp_sound;
    int index;
    double gamma, humidity, lpsat, R, temp, theta;

    for (index = 1; index <= n_interfaces; index++)
    {
        /*
         * Angle Between Wind And Propagation Path
         */
        if (RECIPROCITY_FLAG)
        {
            /*
             * Source Centered Geometry
             */
            theta = a[index].wind_theta - PI - bearing;
        }
        else
        {
            /*
             * Sensor Centered Geometry
             */
            theta = a[index].wind_theta - bearing;
        }

        /*
         * The Partial Pressure of Saturated Water Vapor
         */
        temp = a[index].kelvin / T0;
    }
}
lpsat = -10.79586 / temp - 5.02808 * log10(temp) 
- 1.50474e-4 * pow(10.0, 8.29692 * (1 - temp)) 
+ 4.28730e-4 * pow(10.0, 4.76955 * (1 - temp)) 
+ 8.575983444;

/*
 * The Fraction of Water Molecules in Air
 */
humidity = a[index].relative_humidity 
* pow(10.0, (lpsat - 2)) / a[index].pressure;

/*
 * Ratio of Specific Heats
 */
gamma = (humidity + 7) / (humidity + 5);
R = R0 / (29 - 11 * humidity);

/*
 * Sound Speed Due to Humidity And Temperature
 */
a[index].c = sqrt(a[index].kelvin * gamma * R);

/*
 * Adjustment For Wind
 */
a[index].c += a[index].wind_speed * cos(theta);

} /*
 * Print Sound Speed Profile If Desired
 */
if (SOUND_SPEED_OUT_FLAG)
{
    if ((fp_sound = fopen("sound.o","a")) == NULL)
    {
        fprintf(stderr,"error number %d\n",errno);
        perror("sound.o");
        exit(1);
    }
    for (index = 1; index <= n_interfaces; index++)
        fprintf(fp_sound,"%10.4lf%s10.4lf\n", 
            a[index].z,a[index].c);
    fclose(fp_sound);
}
VOLTAGE.C

/*
 * FINDS THE VOLTAGE detector_voltage AT index_detector
 * DUE TO CURRENT SOURCE source_current AT index_source
 */

#include "complex.h"
#include "ffp.h"

void admit();
void clip();

void voltage(detector_voltage, gp, k, layer, n_layers, omega)
complex *detector_voltage;
double k, omega;
int n_layers;
struct gp *gp;
struct layer *layer;
{
complex admittance_l, admittance_n, source_admittance,
source_current, source_voltage, v;
int last_l, last_n;

/*
 * zero default voltage
 */
Re(*detector_voltage) = Im(*detector_voltage) = Re(v) = Im(v) = 0;

/*
 * find last significant line toward half-space #N
 */
clip(gp, k, &last_n, layer, n_layers, omega);
if(last_n != -1)
{ /* if (v = = 0) return zero volts */

/*
 * find last significant line toward half-space #1
 */
clip(gp, k, &last_1, layer, 1, n_layers, omega);
if(last_1 != -1)
{ /* if (v = = 0) return zero volts */

/*
 * admittance that the source sees looking toward #N
 */
admit(&admittance_n, gp, last_n, layer, &v);

/*/
admittance that the source sees looking toward #1
admit(&admittance_l, gp, last_l, layer, &v);

/*
total source admittance (add in parallel)
cadd(source_admittance, admittance_l, admittanceji);
/*
source current (equation 14)
* /
Re(source_current) = 2 * k /
    (layer[gp->index_source].rho * omega);
Im(source_current) = 0;
/*
source voltage (equation 20)
*/
cdiv(source_voltage, source_current, source_admittance);
/*
detector voltage (equation 23)
*/
cdiv(*detector_voltage, source_voltage, v);
/*
 * SQUARES OF COMPLEX INTRINSIC WAVE NUMBERS FOR EACH LINE SECTION
 */

#include "complex.h"
#include "ffp.h"

void wavenumber(extra, gp, layer, n_layers, omega)
    double extra, omega;
    int n_layers;
    struct gp *gp;
    struct layer *layer;
{
    complex z;
    int index;
    
    /*
    * finite thickness slabs
    */
    for (index = 2; index < n_layers; index++)
    {
        Re(z) = omega / layer[index].c;
        Im(z) = - extra - layer[index].mu;
        cmul(layer[index].ki2, z, z);
    }
    
    /*
    * half-spaces #1 and #N
    */
    Re(z) = omega / layer[1].c * Re(gp->k_1);
    Im(z) = omega / layer[1].c * Im(gp->k_1) - extra - layer[1].mu;
    cmul(layer[1].ki2, z, z);
    Re(z) = omega / layer[n_layers].c * Re(gp->k_n);
    Im(z) = omega / layer[n_layers].c * Im(gp->k_n) - extra - layer[n_layers].mu;
    cmul(layer[n_layers].ki2, z, z);
}
/* FIND THE IMPEDANCE FOR ONE OR TWO GROUND LAYERS */

K. Attenborough,
* Acoustical Impedance Models for Outdoor Ground Surfaces,
* See equation 15 on page 527 of the above reference.
*/

#include "complex.h"

void z2layers(depth, impedance, k1, zbottom, ztop)
complex *impedance, *k1, zbottom, ztop;
double depth;
{
    complex factor, z, zl, z2, ztan;
double a, b;

    /*
      * Zb = zbottom
      * Zt = ztop
      * Zb - i Zt tan(a + ib)       zl
      * impedance = ---------------  Zt = -- Zt = factor Zt
      * Zt - i Zb tan(a + ib)       z2
      */
    a = depth * Re(*k1);
b = depth * Im(*k1);
    /*
      * tan(a + ib) = (tan(a) + i tanh(b)) / (1 - i tan(a)tanh(b))
      */
    Re(z1) = tan(a);
    Im(z1) = tanh(b);
    Re(z2) = 1;
    Im(z2) = -Re(z1) * Im(z1);
cdiv(ztan, zl, z2);
    /*
      * determine the factor
      */
    cmul(z, ztop, ztan);
    Re(z1) = Re(zbottom) + Im(z);
    Im(z1) = Im(zbottom) - Re(z);
cmul(z, zbottom, ztan);
    Re(z2) = Re(ztop) + Im(z);
    Im(z2) = Im(ztop) - Re(z);
cdiv(factor, zl, z2);
*/
* complex impedance
*/
    cmul(*impedance, factor, ztop);
}
#include "complex.h"

#define GAMMA  1.4
#define RHO0   1.21  /* k / m^3 */

void z4parameter(c, frequency, impedance, wavenumber, omega, pn, sf, sigma)
complex *impedance, *wavenumber;
double c, frequency, omega, pn, sf, sigma;
{
    complex kb, ztemp;
    double alpha, beta, q2, x;

    alpha = TWO_PI * frequency;
    beta = alpha / c;

    x = 1000 * sf * sf * sigma / (RHO0 * alpha);

    q2 = pow(omega, -pn);

    Re(ztemp) = 1.5764 * q2;
    Im(ztemp) = GAMMA * omega * x;
    csqrt(kb, ztemp);

    Re(ztemp) = (4 * q2) / (3 * omega);
    Im(ztemp) = x;
    cdiv(*impedance, ztemp, kb);
/*
 * kb has no units
 */

Re(*wavenumber) = beta * Re(kb);
Im(*wavenumber) = beta * Im(kb);
/*
 * CALCULATES EFFECTIVE GROUND IMPEDANCE
 */
#include <stdio.h>
#include <stddef.h>
#include "complex.h"
#define PRT_IMPED dflags[6]

void z21ayers();
void z4parameter();
extern int errno;

void zeffective(c, frequency, ground, impedance, kl, n_ground_layers, dflags)
complex *impedance, *kl1;
double c, frequency;
int n_ground_layers, *dflags;
struct ground *ground;
{
complex k2, zbottom, ztop;
FILE *fp_imp;

/*
 * find impedance for top ground interface
 */
/*
z4parameter(c, frequency, &ztop, k1, ground->omega_1, ground->pn_1, ground->sf_1,
 ground->sigma_1);
Re(*impedance) = Re(ztop);
Im(*impedance) = Im(ztop);
if (n_ground_layers == 2)
{
*/
/*
 * put in the contribution from the bottom ground interface
 */
/*
z4parameter(c, frequency, &zbotttom, &k2, ground->omega_2, ground->pn_2,
 ground->sf_2,
 ground->sigma_2);
z21ayers(ground->depth, impedance, kl, zbottom, ztop);
*/}
/* Write Out Impedance Value */

if (PRT_JMPED)
{
    if ((fp_imp = fopen("imped.e", "a")) == NULL)
    {
        fprintf(stderr, "error number %d\n", errno);
        perror("imped.e");
        exit(1);
    }
    fprintf(fp_imp, "Impedance = (%10.41f, %10.41f)\n",
            Re(*impedance), -Im(*impedance));
    fprintf(fp_imp, "Wavenumber = (%10.41f, %10.41f)\n",
            Re(*kl), -Im(*kl));
    fclose(fp_imp);
}
#include <math.h>

#define ONE_2SQRTPi 0.28209479177387814347
#define PI 3.14159265358979323846
#define TWO_PI 6.28318530717958647692

#ifndef _COMPLEX_
#define _COMPLEX_

typedef struct
{
    double real, imaginary;
} complex;

/*
 * Re(z) is the real part of z
 * Im(z) is the imaginary part of z
 */
#define Re(z) (z).real
#define Im(z) (z).imaginary

/*
 * z is the complex sum of z1 and z2
 */
#define cadd(z, z1, z2)
    Re(z) = Re(z1) + Re(z2);
    Im(z) = Im(z1) + Im(z2)

/*
 * z is the complex quotient of z1 and z2
 * z = z1 / z2
 */
#define cdiv(z, z1, z2)
    
        denominator = Re(z2) * Re(z2) + Im(z2) * Im(z2);
        temp1 = Im(z1) * Re(z2);
        temp2 = Re(z1) * Im(z2);
        Re(z) = ((Re(z1) + Im(z1)) * (Re(z2) + Im(z2)) - temp1 - temp2) / denominator;
        Im(z) = (temp1 - temp2) / denominator;
/*
   *   z is the complex exponential of zl
   */
#define cexp(z, zl)
{  
double r, theta;
  
  r = exp(Re(zl));
  theta = fmod(Im(zl), TWO_PI);
  Re(z) = r * cos(theta);
  Im(z) = r * sin(theta);
}

/*
   *   cmod(z) returns |z|
   */
#define cmod(z) sqrt(Re(z) * Re(z) + Im(z) * Im(z))

/*
   *   z is the complex product of zl and z2
   */
#define cmul(z, zl, z2)  
{  
  double temp1, temp2;
  
  temp1 = Re(zl) * Im(z2);
  temp2 = Im(zl) * Re(z2);
  Re(z) = (Re(zl) + Im(zl)) * (Re(z2) - Im(z2)) + temp1 - temp2;
  Im(z) = temp1 + temp2;
}

/*
   *   z is the principal square root of zl
   */
#define csqrt(z, zl)  
{  
  double a, b, r;
  
  a = Re(zl);
  b = Im(zl);
  r = sqrt(a * a + b * b);
  /*
   *   check for truncation error
   */
}
if ((Re(z) = (r + a) / 2) > 0)
    Re(z) = sqrt(Re(z));
else
    Re(z) = 0;
if ((Im(z) = (r - a) / 2) > 0)
    Im(z) = sqrt(Im(z));
else
    Im(z) = 0;
/*
 * select the principal branch
 */
    if (b < 0)
        Im(z) = -Im(z);
}

/*
 * z is the complex difference of z1 and z2
 */
#define csb(z, z1, z2)
    Re(z) = Re(z1) - Re(z2);
    Im(z) = Im(z1) - Im(z2)
#endif
```c
#ifndef COMPLEX_
#define _COMPLEX_
typedef struct
{
    double real, imaginary;
} complex;
#endif

struct geometry
{
    double delta_theta, range, theta_1, theta_2, Zr, Zs;
};

struct ground
{
    double depth, omega_1, omega_2, pn_1, pn_2,
    sf_1, sf_2, sigma_1, sigma_2;
};

struct interface
{
    double c, kelvin, pressure, relative_humidity,
    wind_speed, wind_theta, z;
};

struct gp
{
    complex impedance_1, impedance_n, k_1, k_n;
    int index_detector, index_source;
};

struct layer
{
    complex ki2, kz, y0;
    double c, mu, rho, thickness;
};
```
Appendix B
Example Cases
## Example Case #1

### Input Files

**Debug File**

(Sound Speed Profile, Source Centered, Wavenumber Spectrum)

**Weather File**

<table>
<thead>
<tr>
<th>Time</th>
<th>Temperature</th>
<th>Wind Speed</th>
<th>Wind Direction</th>
<th>Humidity</th>
<th>Pressure</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>25.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>50.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>75.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>100.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>125.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>150.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>175.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>200.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>225.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>250.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>275.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>300.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>325.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>350.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>375.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>400.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>425.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>450.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>475.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>500.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>525.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>550.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>575.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>600.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>625.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>650.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>675.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>700.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>725.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>750.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>775.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>800.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
<td>--------</td>
<td>-------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>825.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>850.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>875.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>900.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>925.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>950.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>975.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1000.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1025.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1050.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1075.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1100.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1125.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1150.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1175.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1200.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1225.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1250.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1275.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1300.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1325.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1350.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1375.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1400.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1425.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1450.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1475.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>1500.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

**Geometry File**

| 5.0 | 1.0 | 10000.0 | 0.0 | 0.0 | 1.0 |

**Frequency File**

| 100.0 | 0.0 | 0.0 |

**Ground File**

| 1 | 366.0 | 0.27 | 0.5 | 0.5 |
## Output Files

### Realtime Output:

- Input name of debug file: debug2
- Input name of weather file: easel.wea
- Input name of geometry file: geometry2
- Input name of frequency file: frequency2
- Input name of ground file: ground2

### SINGLE FREQUENCIES

<table>
<thead>
<tr>
<th>INTERFACE</th>
<th>HEIGHT (m)</th>
<th>KELVIN TEMPERATURE</th>
<th>RELATIVE HUMIDITY</th>
<th>ATMOSPHERIC PRESSURE</th>
<th>WIND SPEED</th>
<th>WIND DIRECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>61</td>
<td>1500.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>60</td>
<td>1475.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>59</td>
<td>1450.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>58</td>
<td>1425.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>57</td>
<td>1400.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>56</td>
<td>1375.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>55</td>
<td>1350.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>54</td>
<td>1325.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>53</td>
<td>1300.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>52</td>
<td>1275.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>51</td>
<td>1250.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>50</td>
<td>1225.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>49</td>
<td>1200.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>48</td>
<td>1175.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>47</td>
<td>1150.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>46</td>
<td>1125.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>45</td>
<td>1100.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>44</td>
<td>1075.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>43</td>
<td>1050.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>42</td>
<td>1025.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>41</td>
<td>1000.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>40</td>
<td>975.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>39</td>
<td>950.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>38</td>
<td>925.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>37</td>
<td>900.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>875.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>850.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>825.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>800.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>775.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>750.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>725.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>700.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>675.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>650.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>625.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>600.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>575.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>550.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>525.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>500.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>475.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>450.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>425.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>400.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>375.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>350.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>325.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>300.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>275.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>250.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>225.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>200.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>175.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>150.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>125.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>100.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>75.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>50.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>25.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
</tbody>
</table>
GROUND LAYER     SIGMA     POROSITY
1              366.00     0.27

Position Angle = 0.000000
10000.00      -124.9824

Sound Speed Profile:

Case #1
Relative Sound Pressure Level Range:

Case #1

Wavenumber Spectrum:

Case #1
### Example Case #2

#### Input Files

**Debug File**

133

(Sound Speed Profile, Source Centered, Wavenumber Spectrum)

**Weather File**

<table>
<thead>
<tr>
<th>Time (s)</th>
<th>Temperature (°F)</th>
<th>Wind Speed (mph)</th>
<th>Wind Direction</th>
<th>Relative Humidity (%)</th>
<th>Measurement Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>25.00</td>
<td>299.1287</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>50.00</td>
<td>303.4607</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>75.00</td>
<td>307.8239</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>100.00</td>
<td>312.2182</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>125.00</td>
<td>316.6436</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>150.00</td>
<td>321.1002</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>175.00</td>
<td>325.5879</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>200.00</td>
<td>330.1067</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>225.00</td>
<td>334.6567</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>250.00</td>
<td>339.2379</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>275.00</td>
<td>343.8502</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>300.00</td>
<td>348.4936</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>325.00</td>
<td>353.1682</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>350.00</td>
<td>357.8739</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>375.00</td>
<td>362.6108</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>400.00</td>
<td>367.3788</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>425.00</td>
<td>372.1779</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>450.00</td>
<td>377.0082</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>475.00</td>
<td>381.8697</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>500.00</td>
<td>386.7622</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>525.00</td>
<td>391.6860</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>550.00</td>
<td>396.6408</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>575.00</td>
<td>401.6268</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>600.00</td>
<td>406.6440</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>625.00</td>
<td>411.6923</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>650.00</td>
<td>416.7717</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>675.00</td>
<td>421.8823</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>700.00</td>
<td>427.0240</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>725.00</td>
<td>432.1969</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>750.00</td>
<td>437.4009</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>775.00</td>
<td>442.6361</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>800.00</td>
<td>447.9024</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Value</td>
<td>Geometry File</td>
<td>Frequency File</td>
<td>Ground File</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------</td>
<td>---------------</td>
<td>----------------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.825</td>
<td>5.0</td>
<td>100.0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.850</td>
<td>458.5284</td>
<td>0.0</td>
<td>0.27</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.875</td>
<td>463.8881</td>
<td>0.0</td>
<td>0.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.900</td>
<td>469.2790</td>
<td>0.0</td>
<td>0.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.925</td>
<td>474.7010</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.950</td>
<td>480.1541</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.975</td>
<td>485.6384</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.000</td>
<td>491.1539</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.025</td>
<td>496.7005</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.050</td>
<td>502.2782</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.075</td>
<td>507.8871</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.100</td>
<td>513.5271</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.125</td>
<td>519.1982</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.150</td>
<td>524.9005</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.175</td>
<td>530.6340</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.200</td>
<td>536.3985</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.225</td>
<td>542.1943</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.250</td>
<td>548.0211</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.275</td>
<td>553.8792</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.300</td>
<td>559.7683</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.325</td>
<td>565.6886</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.350</td>
<td>571.6401</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.375</td>
<td>577.6227</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.400</td>
<td>583.6364</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.425</td>
<td>589.6813</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.450</td>
<td>595.7573</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.475</td>
<td>601.8644</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1.500</td>
<td>608.0027</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Geometry File
5.0 1.0 10000.0 0.0 0.0 1.0

Frequency File
100.0 0.0 0.0

Ground File
1
366.0 0.27 0.5 0.5
Output Files

**Realtime Output:**

- Input name of debug file: debug2
- Input name of weather file: case2.wea
- Input name of geometry file: geometry2
- Input name of frequency file: frequency2
- Input name of ground file: ground2

**SINGLE FREQUENCIES**

<table>
<thead>
<tr>
<th>INTERFACE</th>
<th>HEIGHT</th>
<th>KELVIN TEMPERATURE</th>
<th>RELATIVE HUMIDITY</th>
<th>ATMOSPHERIC PRESSURE</th>
<th>WIND SPEED</th>
<th>WIND DIRECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>61</td>
<td>1500.00</td>
<td>608.00</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>60</td>
<td>1475.00</td>
<td>601.86</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>59</td>
<td>1450.00</td>
<td>595.76</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>58</td>
<td>1425.00</td>
<td>589.68</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>57</td>
<td>1400.00</td>
<td>583.64</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>56</td>
<td>1375.00</td>
<td>577.62</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>55</td>
<td>1350.00</td>
<td>571.64</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>54</td>
<td>1325.00</td>
<td>565.69</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>53</td>
<td>1300.00</td>
<td>559.77</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>52</td>
<td>1275.00</td>
<td>553.88</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>51</td>
<td>1250.00</td>
<td>548.02</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>50</td>
<td>1225.00</td>
<td>542.19</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>49</td>
<td>1200.00</td>
<td>536.40</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>48</td>
<td>1175.00</td>
<td>530.63</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>47</td>
<td>1150.00</td>
<td>524.90</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>46</td>
<td>1125.00</td>
<td>519.20</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>45</td>
<td>1100.00</td>
<td>513.53</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>44</td>
<td>1075.00</td>
<td>507.89</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>43</td>
<td>1050.00</td>
<td>502.28</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>42</td>
<td>1025.00</td>
<td>496.70</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>41</td>
<td>1000.00</td>
<td>491.15</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>40</td>
<td>975.00</td>
<td>485.64</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>39</td>
<td>950.00</td>
<td>480.15</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>38</td>
<td>925.00</td>
<td>474.70</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>37</td>
<td>900.00</td>
<td>469.28</td>
<td>0.0</td>
<td>1.00</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>875.00</td>
<td>463.89</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>850.00</td>
<td>458.53</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>825.00</td>
<td>453.20</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>800.00</td>
<td>447.90</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>775.00</td>
<td>442.64</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>750.00</td>
<td>437.40</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>725.00</td>
<td>432.20</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>700.00</td>
<td>427.02</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>675.00</td>
<td>421.88</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>650.00</td>
<td>416.77</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>625.00</td>
<td>411.69</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>600.00</td>
<td>406.64</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>575.00</td>
<td>401.63</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>550.00</td>
<td>396.64</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>525.00</td>
<td>391.69</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>500.00</td>
<td>386.76</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>475.00</td>
<td>381.87</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>450.00</td>
<td>377.01</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>425.00</td>
<td>372.18</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>400.00</td>
<td>367.38</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>375.00</td>
<td>362.61</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>350.00</td>
<td>357.87</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>325.00</td>
<td>353.17</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>300.00</td>
<td>348.49</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>275.00</td>
<td>343.85</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>250.00</td>
<td>339.24</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>225.00</td>
<td>334.66</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>200.00</td>
<td>330.11</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>175.00</td>
<td>325.59</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>150.00</td>
<td>321.10</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>125.00</td>
<td>316.64</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>100.00</td>
<td>312.22</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>75.00</td>
<td>307.82</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>50.00</td>
<td>303.46</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>25.00</td>
<td>299.13</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td></td>
</tr>
<tr>
<td>GROUND LAYER</td>
<td>SIGMA</td>
<td>POROSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-------</td>
<td>----------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>366.00</td>
<td>0.27</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Position Angle = 0.000000  
10000.00  -96.3082

**Sound Speed Profile:**

![Sound Speed Profile](image)

Case #2

Sound Speed (m/s) vs. Height (m)
Relative Sound Pressure Level versus Range:

Case #2

Horizontal Range (m)

Wavenumber Spectrum:

Case #2

k (m⁻¹)
### Example Case #3

#### Input Files

**Debug File**

133

(Sound Speed Profile, Source Centered, Wavenumber Spectrum)

**Weather File**

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>25.00</td>
<td>290.5581</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>50.00</td>
<td>286.3196</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>75.00</td>
<td>282.1121</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>100.00</td>
<td>277.9358</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>125.00</td>
<td>273.7907</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>150.00</td>
<td>269.6767</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>175.00</td>
<td>265.5938</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>200.00</td>
<td>261.5421</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>225.00</td>
<td>257.5216</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>250.00</td>
<td>253.5321</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>275.00</td>
<td>249.5738</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>300.00</td>
<td>245.6467</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>325.00</td>
<td>241.7507</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>350.00</td>
<td>237.8858</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>375.00</td>
<td>234.0521</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>400.00</td>
<td>230.2496</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>425.00</td>
<td>226.4781</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>450.00</td>
<td>222.7378</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>475.00</td>
<td>219.0287</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>500.00</td>
<td>215.3507</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>525.00</td>
<td>211.7038</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>550.00</td>
<td>208.0881</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>575.00</td>
<td>204.5036</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>600.00</td>
<td>200.9501</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>625.00</td>
<td>197.4279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>650.00</td>
<td>193.9367</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>675.00</td>
<td>190.4767</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>700.00</td>
<td>187.0479</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>725.00</td>
<td>183.6502</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>750.00</td>
<td>180.2836</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>775.00</td>
<td>176.9482</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>800.00</td>
<td>173.6439</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Value</td>
<td>825.000</td>
<td>850.000</td>
<td>875.000</td>
<td>900.000</td>
<td>925.000</td>
</tr>
<tr>
<td>------------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>Value</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Value</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>Value</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Value</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Value</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Geometry File
5.0 1.0 10000.0 0.0 0.0 1.0

Frequency File
100.0 0.0 0.0

Ground File
1
366.0 0.27 0.5 0.5
Output Files

Realtime Output:

Input name of debug file: debug2
Input name of weather file: case3.wea
Input name of geometry file: geometry2
Input name of frequency file: frequency2
Input name of ground file: ground2

SINGLE FREQUENCIES

<table>
<thead>
<tr>
<th>INTERFACE</th>
<th>HEIGHT</th>
<th>KELVIN TEMPERATURE</th>
<th>RELATIVE HUMIDITY</th>
<th>ATMOSPHERIC PRESSURE</th>
<th>WIND SPEED</th>
<th>WIND DIRECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>61</td>
<td>1500.00</td>
<td>93.77</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>60</td>
<td>1475.00</td>
<td>96.20</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>59</td>
<td>1450.00</td>
<td>98.66</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>58</td>
<td>1425.00</td>
<td>101.16</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>57</td>
<td>1400.00</td>
<td>103.68</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>56</td>
<td>1375.00</td>
<td>106.24</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>55</td>
<td>1350.00</td>
<td>108.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>54</td>
<td>1325.00</td>
<td>111.45</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>53</td>
<td>1300.00</td>
<td>114.10</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>52</td>
<td>1275.00</td>
<td>116.78</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>51</td>
<td>1250.00</td>
<td>119.49</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>50</td>
<td>1225.00</td>
<td>122.24</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>49</td>
<td>1200.00</td>
<td>125.01</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>48</td>
<td>1175.00</td>
<td>127.82</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>47</td>
<td>1150.00</td>
<td>130.65</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>46</td>
<td>1125.00</td>
<td>133.52</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>45</td>
<td>1100.00</td>
<td>136.42</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>44</td>
<td>1075.00</td>
<td>139.35</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>43</td>
<td>1050.00</td>
<td>142.31</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>42</td>
<td>1025.00</td>
<td>145.31</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>41</td>
<td>1000.00</td>
<td>148.33</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>40</td>
<td>975.00</td>
<td>151.39</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>39</td>
<td>950.00</td>
<td>154.47</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>38</td>
<td>925.00</td>
<td>157.59</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>37</td>
<td>900.00</td>
<td>160.74</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
</tr>
<tr>
<td>36</td>
<td>875.00</td>
<td>163.92</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>35</td>
<td>850.00</td>
<td>167.13</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>34</td>
<td>825.00</td>
<td>170.37</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>33</td>
<td>800.00</td>
<td>173.64</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>32</td>
<td>775.00</td>
<td>176.95</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>31</td>
<td>750.00</td>
<td>180.28</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>30</td>
<td>725.00</td>
<td>183.65</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>29</td>
<td>700.00</td>
<td>187.05</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>28</td>
<td>675.00</td>
<td>190.48</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>27</td>
<td>650.00</td>
<td>193.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>26</td>
<td>625.00</td>
<td>197.43</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>25</td>
<td>600.00</td>
<td>200.95</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>24</td>
<td>575.00</td>
<td>204.50</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>23</td>
<td>550.00</td>
<td>208.09</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>22</td>
<td>525.00</td>
<td>211.70</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>21</td>
<td>500.00</td>
<td>215.35</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>20</td>
<td>475.00</td>
<td>219.03</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>19</td>
<td>450.00</td>
<td>222.74</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>18</td>
<td>425.00</td>
<td>226.48</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>17</td>
<td>400.00</td>
<td>230.25</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>16</td>
<td>375.00</td>
<td>234.05</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>15</td>
<td>350.00</td>
<td>237.89</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>14</td>
<td>325.00</td>
<td>241.75</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>13</td>
<td>300.00</td>
<td>245.65</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>12</td>
<td>275.00</td>
<td>249.57</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>11</td>
<td>250.00</td>
<td>253.53</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>10</td>
<td>225.00</td>
<td>257.52</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>9</td>
<td>200.00</td>
<td>261.54</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>8</td>
<td>175.00</td>
<td>265.59</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>7</td>
<td>150.00</td>
<td>269.68</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>6</td>
<td>125.00</td>
<td>273.79</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>100.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>4</td>
<td>75.00</td>
<td>282.11</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>3</td>
<td>50.00</td>
<td>286.32</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>2</td>
<td>25.00</td>
<td>290.56</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>1</td>
<td>0.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>GROUND LAYER</td>
<td>SIGMA</td>
<td>POROSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>--------------</td>
<td>-------</td>
<td>----------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>366.00</td>
<td>0.27</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Position Angle = 0.000000
10000.00    -143.1275

**Sound Speed Profile:**

![Graph of Sound Speed Profile](image)

Case #3
Relative Sound Pressure Level versus Range:

Wavenumber Spectrum:
### Example Case #4

#### Input Files

**Debug File**
- 133

**Weather File**
- 65

<table>
<thead>
<tr>
<th>Distance (m)</th>
<th>Sound Speed (m/s)</th>
<th>1.00</th>
<th>0.00</th>
<th>0.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.000</td>
<td>294.8279</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>5.000</td>
<td>295.6855</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>10.000</td>
<td>296.5445</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>15.000</td>
<td>297.4046</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>20.000</td>
<td>298.2661</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>25.000</td>
<td>299.1287</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>30.000</td>
<td>299.9926</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>35.000</td>
<td>300.8578</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>40.000</td>
<td>301.7242</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>45.000</td>
<td>302.5918</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>50.000</td>
<td>303.4607</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>55.000</td>
<td>304.3309</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>60.000</td>
<td>305.2022</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>65.000</td>
<td>306.0749</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>70.000</td>
<td>306.9487</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>75.000</td>
<td>307.8239</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>80.000</td>
<td>308.7002</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>85.000</td>
<td>309.5778</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>90.000</td>
<td>310.4567</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>95.000</td>
<td>311.3368</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>100.000</td>
<td>312.2182</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>105.000</td>
<td>313.0996</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>110.000</td>
<td>313.9810</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>115.000</td>
<td>314.8636</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>120.000</td>
<td>315.7467</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>125.000</td>
<td>316.6299</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>130.000</td>
<td>317.5134</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>135.000</td>
<td>318.3971</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>140.000</td>
<td>319.2810</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>145.000</td>
<td>320.1652</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>150.000</td>
<td>321.0496</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>155.000</td>
<td>321.9342</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>160.000</td>
<td>322.8190</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>165.000</td>
<td>323.7040</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>170.000</td>
<td>324.5893</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>175.000</td>
<td>325.4748</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>180.000</td>
<td>326.3604</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>185.000</td>
<td>327.2462</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>190.000</td>
<td>328.1322</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>195.000</td>
<td>329.0183</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>200.000</td>
<td>329.9046</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>205.000</td>
<td>330.7911</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>210.000</td>
<td>331.6777</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>215.000</td>
<td>332.5645</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>220.000</td>
<td>333.4514</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>Frequency</td>
<td>Real</td>
<td>Imaginary</td>
<td>Real</td>
<td>Imaginary</td>
</tr>
<tr>
<td>-----------</td>
<td>-------</td>
<td>------------</td>
<td>-------</td>
<td>------------</td>
</tr>
<tr>
<td>230.000</td>
<td>289.7079</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>240.000</td>
<td>288.0113</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>250.000</td>
<td>286.3196</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>260.000</td>
<td>284.6329</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>270.000</td>
<td>282.9511</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>280.000</td>
<td>281.2744</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>290.000</td>
<td>279.6026</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>300.000</td>
<td>277.9358</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>310.000</td>
<td>276.2703</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>320.000</td>
<td>274.6026</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>330.000</td>
<td>272.9358</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>340.000</td>
<td>271.2744</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>350.000</td>
<td>269.6026</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>360.000</td>
<td>267.9358</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>370.000</td>
<td>266.2703</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>380.000</td>
<td>264.6026</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>390.000</td>
<td>262.9358</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>400.000</td>
<td>261.2703</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>410.000</td>
<td>259.6026</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>420.000</td>
<td>257.9358</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>430.000</td>
<td>256.2703</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>440.000</td>
<td>254.6026</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>450.000</td>
<td>252.9358</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

**Geometry File**

<table>
<thead>
<tr>
<th>Geometry</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.0</td>
<td>1.0</td>
<td>10000.0</td>
<td>0.0</td>
<td>0.0</td>
<td>1.0</td>
<td></td>
</tr>
</tbody>
</table>

**Frequency File**

<table>
<thead>
<tr>
<th>Frequency</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Ground File**

<table>
<thead>
<tr>
<th>Ground</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>366.0</td>
<td>0.27</td>
<td>0.5</td>
<td>0.5</td>
<td></td>
</tr>
</tbody>
</table>

103
Output Files

Realtime Output:

Input name of debug file: debug2
Input name of weather file: case4.wea
Input name of geometry file: geometry2
Input name of frequency file: frequency2
Input name of ground file: ground2

SINGLE FREQUENCIES

<table>
<thead>
<tr>
<th>INTERFACE</th>
<th>HEIGHT</th>
<th>KELVIN TEMPERATURE</th>
<th>RELATIVE HUMIDITY</th>
<th>ATMOSPHERIC PRESSURE</th>
<th>WIND SPEED</th>
<th>WIND DIRECTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>65</td>
<td>1500.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>64</td>
<td>1450.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>63</td>
<td>1400.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>62</td>
<td>1350.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>61</td>
<td>1300.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>60</td>
<td>1250.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>59</td>
<td>1200.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>58</td>
<td>1150.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>57</td>
<td>1100.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>56</td>
<td>1050.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>55</td>
<td>1000.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>54</td>
<td>950.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>53</td>
<td>900.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>52</td>
<td>850.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>51</td>
<td>800.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>50</td>
<td>750.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>49</td>
<td>700.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>48</td>
<td>650.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>47</td>
<td>600.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>46</td>
<td>550.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>45</td>
<td>500.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>44</td>
<td>450.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>43</td>
<td>400.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>42</td>
<td>350.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>41</td>
<td>300.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>40</td>
<td>250.00</td>
<td>277.94</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>39</td>
<td>200.00</td>
<td>281.27</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>38</td>
<td>150.00</td>
<td>282.95</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>37</td>
<td>100.00</td>
<td>284.63</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>Product</td>
<td>Quantity</td>
<td>Total Price</td>
<td>Price</td>
<td>Discount</td>
<td>Discounted Price</td>
<td>Tax</td>
</tr>
<tr>
<td>---------</td>
<td>----------</td>
<td>-------------</td>
<td>-------</td>
<td>----------</td>
<td>------------------</td>
<td>-----</td>
</tr>
<tr>
<td>36</td>
<td>250.00</td>
<td>286.32</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>35</td>
<td>240.00</td>
<td>288.01</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>34</td>
<td>230.00</td>
<td>289.71</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>33</td>
<td>220.00</td>
<td>291.41</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>32</td>
<td>210.00</td>
<td>293.12</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>31</td>
<td>200.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>30</td>
<td>190.00</td>
<td>296.54</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>29</td>
<td>180.00</td>
<td>298.27</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>28</td>
<td>170.00</td>
<td>299.99</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>27</td>
<td>160.00</td>
<td>301.72</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>26</td>
<td>150.00</td>
<td>303.46</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>25</td>
<td>140.00</td>
<td>305.20</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>24</td>
<td>130.00</td>
<td>306.95</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>23</td>
<td>120.00</td>
<td>308.70</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>22</td>
<td>110.00</td>
<td>310.46</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>21</td>
<td>100.00</td>
<td>312.22</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>20</td>
<td>95.00</td>
<td>311.34</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>19</td>
<td>90.00</td>
<td>310.46</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>18</td>
<td>85.00</td>
<td>309.58</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>17</td>
<td>80.00</td>
<td>308.70</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>16</td>
<td>75.00</td>
<td>307.82</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>15</td>
<td>70.00</td>
<td>306.95</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>14</td>
<td>65.00</td>
<td>306.07</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>13</td>
<td>60.00</td>
<td>305.20</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>12</td>
<td>55.00</td>
<td>304.33</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>11</td>
<td>50.00</td>
<td>303.46</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>10</td>
<td>45.00</td>
<td>302.59</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>9</td>
<td>40.00</td>
<td>301.72</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>8</td>
<td>35.00</td>
<td>300.86</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>7</td>
<td>30.00</td>
<td>299.99</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>6</td>
<td>25.00</td>
<td>299.13</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>5</td>
<td>20.00</td>
<td>298.27</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>4</td>
<td>15.00</td>
<td>297.40</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>3</td>
<td>10.00</td>
<td>296.54</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>2</td>
<td>5.00</td>
<td>295.69</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>1</td>
<td>0.00</td>
<td>294.83</td>
<td>0.0</td>
<td>1.000</td>
<td>0.000</td>
<td>0.000</td>
</tr>
<tr>
<td>GROUND LAYER</td>
<td>SIGMA</td>
<td>POROSITY</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>-------------</td>
<td>------</td>
<td>----------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>366.00</td>
<td>0.27</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Position Angle = 0.000000
10000.00   -114.5068

**Sound Speed Profile:**

![Sound Speed Profile Graph]

Case #4
Relative Sound Pressure Level versus Range:

Case #4

Horizontal Range (m)

Wavenumber Spectrum:

Magnitude of Spectrum

Case #4

k (m⁻¹)
Appendix C
Debug Features
As mentioned in section 3.2, SCAFFIP allows for input of a debug code to allow for modification of some of the default parameters and provide some additional output information. This allows for a standardization of the input and output format, yet allow for versatility. Currently, the debug files consist of up to six files: two input files and four output files.

**Input Files**

**Code: 8**

**File Name:** extra.los  
**Format:** extra

where  
extra - value for extra loss typically between $10^6$ to $10^2$ (default $10^4$)

**Code: 16**

**File Name:** trans.frm  
**Format:** points Npan

where  
points - number of points to use in calculations. Must be power of 2. (default 1024)  
Npan - set the number of panels to use in calculation.  
Npan < 0 Calculate number of panels (default -2)  
Npan > 0 Set number of panels
Output Files

Code: 1
File Name: sound.o
Format:
  Z(1)   C(1)
  Z(2)   C(2)
  ...
  Z(Nint) C(Nint)

where
  Nint - number of interfaces.
  Z(i)  - height of the ith interface in meters.
  C(i)  - sound speed of the ith interface in m/s.

Code: 32
File Name: npan.e
Format:
  Kmax   Nyq   Npan
  Kwidth DelR  Delk

where
  Kmax  - maximum wave number in summation
  DelR  - range increment
  Delk  - wavenumber increment
  Npan  - number of panels
  Kwidth - wavenumber band width per panel
  Nyq   - nyquist criteria

Code: 64
File Name: imped.e
Format:
  Zreal  Zimg
  Kreal  Kimg
where

\[ Z_{\text{real}}, Z_{\text{img}}, K_{\text{real}}, K_{\text{img}} \] - real and imaginary part of the ground impedance
\[ K_{\text{real}}, K_{\text{img}} \] - real and imaginary part of the wavenumber

Code: 128
File Name: wavnum.e
Format:
\[
\begin{align*}
\text{wavenum}(1) & \quad \text{mag}(1) \\
\text{wavenum}(2) & \quad \text{mag}(2) \\
\vdots & \quad \vdots \\
\text{wavenum(\text{points})} & \quad \text{mag(\text{points})}
\end{align*}
\]

where

\[ \text{points} \] - number of points in transform
\[ \text{wavenum}(i) \] - wavenumber of ith point
\[ \text{mag}(i) \] - magnitude of the kernal of the ith point
Appendix D
User Developed Interface
Requirements for SCAFFIP
This appendix discusses the possibility of the user wishing to develop a personalized front end user interface for SCAFFIP. As mentioned in section 1, deviations from the standard user interface supported by ARL/BED will not automatically be provided debugging assistance. Future upgrades will conform to the standard interface and will not be integrated into a user developed interface. This policy is in effect because of funding and personnel shortages. Limited assistance can be provided for input data and problems related to the propagation models. Appendix A contains the complete source code listing for the model.

The front-end user interface for SCAFFIP is configured so the five input files are read. There are some additional input files that may be read; however, these files deal with the debug number read in. The five primary input files are discussed here. Debug input files are discussed in appendix C. The front-end user interface files for SCAFFIP are main.c, input.c, header.c, parser.c, and sound.c. The file main.c is the main driver program that calls all the other parts of the program. The file input.c is the function that reads the five input files and calls parser to decipher the debug code. The file header.c provides a visual printout of the input parameters. The file sound.c calculates the sound speed profile from the meteorological data and the given geometry. Because the sound speed profile is dependent on a vector relationship between the wind direction and the direction of propagation, the sound speed profile is calculated at each azimuth value. The functions input and sound initialize three of the structures contained in the include file ffp.h.

To develop a new user interface, the new user interface must initialize these three structures: (1) geometry, (2) ground, and (3) interface. The include file ffp.h contains two other structures; however, these structures are initialized and used within propmod exclusively. The geometry structure contains the parameters that describe the structure of the geometry of the problem: source height, receiver height, horizontal range, starting scan angle, ending scan angle, and angular resolution of scan. The ground structure contains all of the parameters for calculating the ground impedance using Attenborough's Four Parameter Model (see section 2.4). The interface structure contains the meteorological profile data. Although this data is used in the function sound, some of this information is required by the function propmod to calculate the
molecular absorption coefficients (see section 2.3). While modifying the standard user interface, the user may choose to input the values for the ground impedance directly instead of using the built-in ground impedance model. This can be accomplished by storing the values into the structure gp and replacing the variable ground in the argument list for propmod by a variable of type gp that contains the user input ground impedance value. An alternative method for storing the ground impedance is to use one of the unused debug numbers and place a conditional statement to read the file containing the ground impedance values or use the ground impedance model to calculate the values. The second method is preferred because it conforms better to the programming philosophy used in the code development.

After the three structures are initialized, the control can be passed off to the propagation function propmod. The argument list for propmod is

\[
\begin{align*}
  a & \quad \text{- interface structure array} \\
  fd & \quad \text{- integer file pointer to a binary output file} \\
  fp & \quad \text{- file pointer to an ASCII output file} \\
  frequency & \quad \text{- frequency of interest} \\
  src\_level & \quad \text{- source level} \\
  back\_level & \quad \text{- background level} \\
  geometry & \quad \text{- geometry structure} \\
  ground & \quad \text{- ground structure} \\
  n\_ground\_layers & \quad \text{- number of ground layers} \\
  n\_interfaces & \quad \text{- number of atmospheric interfaces} \\
  dflags & \quad \text{- array of debug parameters.}
\end{align*}
\]

When the function propmod is called, these are the arguments that must be passed. These are the areas that must be considered when developing a new user interface.

If the output file needs to be modified, the computer code for storing the model results is located at the end of the function propmod. The open statement for the output file is located in main because main controls the scanning loop. If the number of files opened at once were to be minimized, the open statement could be placed in propmod also; however, the open statement would have to
be in setup to always append to the file. This method would only allow the output file to be open when there is data to be written. As an error check, a statement would have to be added to check for the presence of the output file at the beginning of the program to prevent appending to a file already present.

These are the basic areas in which the user may wish to alter the program. Other areas may deal with the input and output of the parameters associated with the debug parameters. These items are discussed in appendix C.
<table>
<thead>
<tr>
<th>Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commandant</td>
</tr>
<tr>
<td>U.S. Army Chemical School</td>
</tr>
<tr>
<td>ATTN: ATZN-CM-CC (Mr. Barnes)</td>
</tr>
<tr>
<td>Fort McClellan, AL 36205-5020</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>NASA Marshall Space Flight Center</td>
</tr>
<tr>
<td>Deputy Director</td>
</tr>
<tr>
<td>Space Science Laboratory</td>
</tr>
<tr>
<td>Atmospheric Sciences Division</td>
</tr>
<tr>
<td>ATTN: E501 (Dr. Fichtl)</td>
</tr>
<tr>
<td>Huntsville, AL 35802</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>NASA/Marshall Space Flight Center</td>
</tr>
<tr>
<td>Atmospheric Sciences Division</td>
</tr>
<tr>
<td>ATTN: Code ED-41</td>
</tr>
<tr>
<td>Huntsville, AL 35812</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>Deputy Commander</td>
</tr>
<tr>
<td>U.S. Army Strategic Defense Command</td>
</tr>
<tr>
<td>ATTN: CSSD-SL-L (Dr. Lilly)</td>
</tr>
<tr>
<td>P.O. Box 1500</td>
</tr>
<tr>
<td>Huntsville, AL 35807-3801</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>Deputy Commander</td>
</tr>
<tr>
<td>U.S. Army Missile Command</td>
</tr>
<tr>
<td>ATTN: AMSMI-RD-AC-AD (Dr. Peterson)</td>
</tr>
<tr>
<td>Redstone Arsenal, AL 35898-5242</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>Commander</td>
</tr>
<tr>
<td>U.S. Army Missile Command</td>
</tr>
<tr>
<td>ATTN: AMSMI-RD-DE-SE (Mr. Lill, Jr.)</td>
</tr>
<tr>
<td>Redstone Arsenal, AL 35898-5245</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>
Commander
Pacific Missile Test Center
Geophysics Division
ATTN: Code 3250 (Mr. Battalino)
Point Mugu, CA 93042-5000

Commander
Code 3331
Naval Weapons Center
ATTN: Dr. Shlanta
China Lake, CA 93555

Lockheed Missiles & Space Co., Inc.
Kenneth R. Hardy
ORG/91-01 B/255
3251 Hanover Street
Palo Alto, CA 94304-1191

Commander
Naval Ocean Systems Center
ATTN: Code 54 (Dr. Richter)
San Diego, CA 92152-5000

Meteorologist in Charge
Kwajalein Missile Range
P.O. Box 67
APO San Francisco, CA 96555

U.S. Department of Commerce Center
Mountain Administration
Support Center, Library, R-51
Technical Reports
325 S. Broadway
Boulder, CO 80303
Commander
Space & Naval Warfare System Command
ATTN: PMW-145-1G
Washington, DC 20362-5100

Director
Naval Research Laboratory
ATTN: Code 4110
(Mr. Ruhnke)
Washington, DC 20375-5000

Commandant
U.S. Army Infantry
ATTN: ATSH-CD-CS-OR (Dr. E. Dutoit)
Fort Benning, GA 30905-5090

USAFETAC/DNE
Scott AFB, IL 62225

Air Weather Service
Technical Library - FL4414
Scott AFB, IL 62225-5458

USAFETAC/DNE
ATTN: Mr. Glauber
Scott AFB, IL 62225-5008

Headquarters
AWS/DOO
Scott AFB, IL 62225-5008

Commander
U.S. Army Combined Arms Combat
ATTN: ATZL-CAW
Fort Leavenworth, KS 66027-5300
Dr. Jerry Davis
North Carolina State University
Department of Marine, Earth, and Atmospheric Sciences
P.O. Box 8208
Raleigh, NC 27650-8208

Commander
U.S. Army CECRL
ATTN: CECRL-RG (Dr. Boyne)
Hanover, NH 03755-1290

Commanding Officer
U.S. Army ARDEC
ATTN: SMCAR-IMI-I, Bldg 59
Dover, NJ 07806-5000

Commander
U.S. Army Satellite Comm Agency
ATTN: DRCPM-SC-3
Fort Monmouth, NJ 07703-5303

Commander
U.S. Army Communications-Electronics
Center for EW/RSTA
ATTN: AMSEL-EW-MD
Fort Monmouth, NJ 07703-5303

Commander
U.S. Army Communications-Electronics
Center for EW/RSTA
ATTN: AMSEL-EW-D
Fort Monmouth, NJ 07703-5303
Commander
U.S. Army Dugway Proving Ground
ATTN: STEDP-MT-M (Mr. Bowers)
Dugway, UT 84022-5000

Commander
U.S. Army Dugway Proving Ground
ATTN: STEDP-MT-DA-L
Dugway, UT 84022-5000

Defense Technical Information Center
ATTN: DTIC-OCP
Cameron Station
Alexandria, VA 22314-6145

Commander
U.S. Army OEC
ATTN: CSTE-EFS
Park Center IV
4501 Ford Ave
Alexandria, VA 22302-1458

Commanding Officer
U.S. Army Foreign Science & Technology Center
ATTN: CM
220 7th Street, NE
Charlottesville, VA 22901-5396

Naval Surface Weapons Center
Code G63
Dahlgren, VA 22448-5000