ncpaprop 1.3.17

NCPA Infrasound Propagation Modelling Package

Contributions from: Roger Waxler, Doru Velea, Claus Hetzer, Jelle Assink, Phil Blom, Joel Lonzaga

Technical coordination: Roger Waxler

Contents

1 Introduction

ncpaprop is a software package aiming at providing a comprehensive set of tested and validated numerical models for simulating the long range propagation of infrasonic signals through the earth's atmosphere. The algorithms implemented in **ncpaprop** are designed for frequencies large enough that the effects of buoyancy can be neglected and small enough that propagation to ranges of hundreds to thousands of kilometers is possible without significant signal attenuation. Nominally, **ncpaprop** can, without modification, be used to efficiently model narrowband propagation from 0.1 to 10 Hz and broadband propagation from 0.05 Hz to 2 or 3 Hz. The models become increasingly inefficient with increasing frequency so that run times can become prohibitive if higher frequencies are considered.

The intent behind **ncpaprop** is to provide reliable software engines for the modeling of infrasound propagation rather than a user-friendly working environment. As such no graphical interfaces are included. Rather **ncpaprop** provides a suite of UNIX style command line programs that can be scripted into user-supplied graphical interfaces as desired. **ncpaprop** is, at this time, not stand-alone, but makes use of several publicly available numerical libraries. **ncpaprop** proper is written in C++; however, some of the programs contained in the external libraries are written in Fortran.

Atmospheric state models are user provided and can be input as series of ascii files. Ascii file input provides the user with a high degree of flexibility, allowing the use of any atmospheric model. In addition, input from G2S binary files is supported (for a discussion of the G2S atmospheric state specifications see, for example, the discussion in §4 of [7]). Stratified and range dependent atmospheric models are supported. Winds are treated both in the effective sound speed approximation, in which the influence of the winds are approximated by adding the horizontal component of the along-path wind to the sound speed [8, 4] and rigorously in what will be referred to as high Mach number models. Atmospheric attenuation can be implemented through user provided ascii files or through the built-in Sutherland-Bass model [10].

ncpaprop provides both geometrical acoustic and full wave models. In the **ncpaprop** package proper the geometrical acoustics consists of 2-d and 3-d ray tracing programs as well as a non-linear ray theory model that calculates the signal amplitude along a ray in the weak shock theory approximation. The 2-d ray tracing is done in the effective sound speed approximation. Both 2-d and 3-d ray tracing is done in a stratified atmosphere. The non-linear ray theory is for a 3-d, range dependent medium, but requires input from a more advanced geometrical acoustics package to utilize its full capabilities. For more advanced geometrical acoustics modeling we recommend the GeoAc package available from the Los Alamos National Laboratory at https://github.com/philblom/GeoAc/archive/master.zip.

The full wave models provided by **ncpaprop** consist of a suite of normal mode models of increasing complexity and a Parabolic Equation (PE) model. All the full wave models are currently restricted to the planar (2-d) approximation in which it is assumed that propagation paths do not deviate from a fixed vertical plane. In effect, the influence of cross winds are being neglected. Effective sound speed and high Mach number normal mode models are provided in the stratified approximation in addition to an effective sound speed normal mode model for range dependent atmospheres. The PE model is an effective sound speed model. Atmospheric attenuation is handled in the normal mode models using first order perturbation theory. A full complex mode model is included which treats the attenuation rigorously (in the approximation that attenuation is modelled using an attenuation coefficient), however, complex mode solvers are computationally intensive and prone to instability so its use is largely for purposes of cross testing and validation. Attenuation is treated non-perturbatively in the PE model.

This manual is organized as folows. There is a chapter detailing the installation of the package. This is followed by a chapter discussing how the code is generally run and what atmospheric specifications are used. Then, each individual program is discussed in a chapter containing a matematical inroduction, details on running the program, and examples. Each program provides a manual, or help, page which is reproduced and discussed.

2 Installation notes

2.1 Prerequisites

2.1.1 Operating System

ncpaprop is designed to run in a UNIX environment. It has been tested on Ubuntu and Fedora Linux operating systems, and MacOS Mountain Lion and Yosemite (10.8-10.10).

2.1.2 Libraries

Fourier transforms are performed in **ncpaprop** using the **FFTW** libraries, http://www.fftw.org. The **GSL** libraries, http://www.gnu.org/software/gsl/, are used for interpolation and splining. The **SLEPc** library, http://www.slepc.upv.es/, is used to solve the large dimensional eigenvalue problems required for the normal mode models. **SLEPc** requires the **PETSc** libraries, http://www.mcs.anl.gov/petsc/, to which it can be seen as a submodule. The current version of **ncpaprop** is designed to use, and has been tested with, versions 3.2-p3 of **SLEPc** and 3.2-p5 of **PETSc**. As these are not the current versions of **SLEPc** and **PETSc** they are packaged with the **ncpaprop** distribution.

2.1.3 Compilers

ncpaprop is written in C++ and was built and tested using the GNU C++ compiler **G++**. The **PETSc** and **SLEPc** libraries are partially written in Fortran and were built and tested using the GNU Fortran compiler **gfortran**. Both of these are required, and they must be of the same version.

2.2 Procedure

ncpaprop uses a standard GNU-style configure script to detect and locate the above prerequisites and generate appropriate makefiles. The `configure` script will search the standard library locations, plus if Fink is detected it will search /sw/lib. An additional flag can be invoked to allow the installer to automatically download and install the prerequisites in system-standard locations if desired. The details of this process are explained below.

2.2.1 Installation Overview

To begin installation, uncompress the distribution with

```
tar xvzf ncpaprop-1.3.17.tar.gz
```

and cd into `ncpaprop-1.3.17`. Then run

```
./configure
```

to have the installer search for prerequisites and exit with an explanatory message if one is missing. Optional flags that can be used by the `configure` script include:

`--enable-auto-dependencies`
: Attempt to download and install missing prerequisites automatically. Use this option only if installation in the default installation locations is satisfactory. The **ncpaprop** installer supports the **apt-get** and **yum** package managers on Linux, and **port** and **fink** on MacOS. The exact commands run by each package manager are detailed in later sections.

`--disable-library-guess`
: If the detected MacOS package manager is Fink, the standard C++ libraries are often not placed in a default-searched location. To handle this, the installer will attempt to guess the location of the libraries and add that guess to the linker's search path. Use this flag to disable this behavior.

`LDFLAGS="-L/path/to/library"`
: The user can also specify additional library search paths using this syntax, often in conjunction with `--disable-library-guess`. For example if the **GSL** or **FFTW** libraries are placed in a non-standard location this option can be used to provide the correct path.

The following tests are run by the `configure` script:

2.2.2 Shells and Interpreters

The configuration script begins by checking for the existence of `bash` and `perl` executables. If these are not both found, the installer exits with an error message. This should never happen on a properly functioning system, but is included as a sanity check.

2.2.3 Package Managers

Depending on whether the host OS is Linux or MacOS, the installer searches for a known package manager. On Linux, the installer will search for `apt-get` or `yum`, in that order, while on MacOS the installer will search for `port` or `fink`, in that order. If no package manager is found, and the `--enable-auto-dependencies` flag is set, the installer will exit with an error message.

2.2.4 g++

The installer will search for either `g++-4` or `g++` on its path, using the first one it finds. (`g++-4` is preferred because its presence indicates that Fink is in use, and Fink does not override the default XCode version of `g++`). If neither of these is found, and the `--enable-auto-dependencies` flag is set, the following commands will be run using `sudo`:

Manager	Command(s)
apt-get	`apt-get -y install g++`
yum	`yum install gcc-c++`
port	`port install gcc48` `port select --set gcc mp-gcc48`
fink	`fink install gcc48`

MacPorts users should note that the 'port select' command will create symlinks in /opt/local/bin that will take precedence over previously installed versions of gcc, g++, and gfortran. This can be undone, if desired, using:

```
sudo port select --set gcc none
hash -r
```

Depending on your system setup, you may have to manually add /opt/local/lib/gcc48 to the library search path using `./configure LDFLAGS=-L/opt/local/lib/gcc48`. The installer will attempt to do this automatically for MacPorts users.

2.2.5 gfortran

The installer will search for `gfortran` on its path. If this is not found, and the `--enable-auto-dependencies` flag is set, the following commands will be run using `sudo`:

Manager	Command(s)
apt-get	`apt-get -y install gfortran`
yum	`yum install gcc-gfortran`
port	`port install gcc48` `port select --set gcc mp-gcc48`
fink	`fink install gcc48`

Note that the `port` and `fink` commands are identical to those for `g++`; `port` and `fink` install a suite of compilers at once, so installing the package for one compiler should install the others at the same time.

2.2.6 Compiler Versions

The installer will then check that the `g++` and `gfortran` versions are identical, and issue an error message if they are not. This possibility is not easily solved automatically, so it is left to the user to determine how it should be solved.

2.2.7 FFTW

The installer will then check for the presence of libraries for FFTW version 3. If they are not found, and the `--enable-auto-dependencies` flag is set, the following commands are issued on Linux hosts using `sudo`:

Manager	Command(s)
apt-get	`apt-get -y install libfftw3-dev`
yum	`yum install fftw-devel`

On MacOS hosts, FFTW is downloaded and installed from source:

```
mkdir prereq
cd prereq
curl -O http://www.fftw.org/fftw-3.3.4.tar.gz
tar xvzf fftw-3.3.4.tar.gz
cd fftw-3.3.4
./configure
make
sudo make install
cd ../..
rm -r prereq
```

2.2.8 GSL

The installer will then check for the presence of libraries for the GSL scientific library. If they are not found, and the `--enable-auto-dependencies` flag is set, the following commands are issued on Linux hosts using `sudo`:

Manager	Command(s)
apt-get	`apt-get -y install libgsl0`
yum	`yum install gsl-devel`

On MacOS hosts, the GSL is downloaded and installed from source:

```
mkdir prereq
cd prereq
curl -O ftp://ftp.gnu.org/gnu/gsl/gsl-1.16.tar.gz
tar xvzf gsl-1.16.tar.gz
cd gsl-1.16
./configure --disable-shared --disable-dependency-tracking
make
sudo make install
cd ../..
rm -r prereq
```

2.2.9 Compilation

Once the configuration script has successfully run, the package can be built simply by running

```
make
```

This will unpack and build the included versions of **PETSc** and **SLEPc** locally. This can take some time. It will then compile the libraries and executables that make up **ncpaprop**. If the Fink `g++-4` is the selected compiler, the installer will try to guess the location of the Fink-installed standard C++ libraries (unless the `--disable-library-guess` flag was set). The guess uses the reported `g++-4` version to calculate its path under the Fink path, so for example if Fink is installed in its default `/sw` location and `g++-4` is version 4.8.4, the guessed library path will be `/sw/lib/gcc4.8/lib`.

One common failure mode is for the linker to attempt to use the standard C++ libraries from a different version than the compiler that was used. This has been more common of a problem for MacOS systems than for Linux systems. The problem is generally indicated by multiple error messages indicating that various common C++ library functions cannot be found. If this occurs, the user should re-run `./configure` with the `--disable-library-guess` flag and, if known, the

```
LDFLAGS="-L/path/to/C++/libraries"
```

instruction.

2.2.10 Testing

To check proper functionality, a suite of test scripts have been written to compare the output of the simpler tools with that generated on the development machine. This allows a sanity check to make sure that identical commands give identical results (within reasonable rounding errors). To confirm that the compilation has been successful, run

```
make test
```

and check the results. If there are errors, check `test/testlog.txt` for clues as to why things didn't work properly. A successful test should print the following output to the screen:

```
$ make test
make -C test test
make[1]: Entering directory ‘/home/claus/ncpaprop/v1.3/test’
rm testlog.txt

*** Running 2D Ray Theory Routines ***
/bin/bash run_raytrace.2d_test.bash >> ./testlog.txt

*** Running 3D Ray Theory Routines ***
```

```
/bin/bash run_raytrace.3d_test.bash >> ./testlog.txt

*** Running Effective-Sound-Speed Modal Routines ***
/bin/bash run_Modess_test.bash >> ./testlog.txt

*** Running Complex Effective-Sound-Speed Modal Routines ***
/bin/bash run_CModess_test.bash >> ./testlog.txt

*** Running One-Way Coupled Modal Routines ***
/bin/bash run_ModessRD1WCM_test.bash >> ./testlog.txt

*** Running Wide-Angle Modal Routines ***
/bin/bash run_WMod_test.bash >> ./testlog.txt

*** Checking 2D Ray Theory Calculation Results ***
/usr/bin/perl compare_raytrace.2d_test
raytrace.2d test OK

*** Checking 3D Ray Theory Calculation Results ***
/usr/bin/perl compare_raytrace.3d_test
raytrace.3d test OK

*** Checking Effective-Sound-Speed Modal Calculation Results ***
/bin/bash compare_Modess_test
Column 2 of Modess test OK
Column 3 of Modess test OK
Column 4 of Modess test OK

*** Checking Complex Effective-Sound-Speed Modal Calculation Results ***
/bin/bash compare_CModess_test
Column 2 of CModess test 1 OK
Column 3 of CModess test 1 OK
Column 4 of CModess test 1 OK
Column 5 of CModess test 1 OK

*** Checking One-Way Coupled Modal Calculation Results ***
/bin/bash compare_ModessRD1WCM_test
Column 2 of ModessRD1WCM test OK
Column 3 of ModessRD1WCM test OK
Column 4 of ModessRD1WCM test OK

*** Checking Wide-Angle Modal Calculation Results ***
/bin/bash compare_WMod_test
Column 2 of WMod test OK
Column 3 of WMod test OK
Column 4 of WMod test OK
make[1]: Leaving directory ‘/home/claus/ncpaprop/v1.3/test’
```

3 General notes

3.1 Running the codes - general rules

The various program executables are to be run in command mode in the Linux/Unix environment. Each command contains the name of the executable file followed by various options. The options are recognized by being preceded by two minus symbols `--`. The program options can be of two kinds: pairs specifying the option followed by the value of the option, of the form [option-name value], or flags. The values can be numbers or strings (arrays of characters). The flags are (boolean) switch signaling that a certain action must occur and do not take any value after them. The general syntax to run a program is:

```
program_name [--option1 val1] [--option2 val2] [...] [--flag1] [...]
```

Running `program_name` without any options sends a manual page to the screen listing the available flags and options with explanations.

With some exceptions the order of the options and flags in a command does not matter. The specification of certain options is essential for the unambiguous definition of a run, while others are specified only as needed. If any one of the essential options is absent from the command line, the program will warn the user of the missing option and abort. If the same option is accidentally present multiple times in the command line the last specification is the one that takes effect.

An alternative way to present options to a program is through an options file `program_name.options`. Each line in this text file contains `option-name : value` pairs separated by a colon. Flags are specified by themselves on separate lines. Note that the command line options take precedence over options supplied in an `.options` file. Examples of such `.options` files are provided in the `samples` directory.

The output of the code is typically saved in text files that are column based. The file names are fixed (hard-coded) and, hopefully, suggestive of their content. All output is saved in the directory from which the run command is given. Unless otherwise specified, signal amplitudes are relative to a reference unit amplitude at 1 km. In all cases pressures scaled by the square root of density, $\rho^{-\frac{1}{2}}p$, are written to files rather than the pressure itself.

3.2 Atmospheric specifications

Necessary input to all propagation codes in this package is the specification of the state of the atmosphere through which propagation is to be modelled. To provide portability and maximum flexibility to the user the state of the atmosphere is input in column-based text files to be provided by the user. Each such file specifies a stratified atmosphere and contains columns such as pressure, density, temperature and the three wind components u, v and w as a function of altitude. Since the vertical wind component w is not used by most of the algorithms its specification is optional. The column order is indicated to the program with the option `--atmosfileorder` followed by a string of characters specifying the order of the columns in the file. For example the profile file may have 7 columns in the order `zuvwtdp`:

```
z - altitude [ km above Mean Sea Level (MSL) ]
u - West-to-East wind speed [ m/s ]
v - South-to-North wind speed [ m/s ]
w - vertical wind (usually zero) [ m/s ]
t - temperature [ Kelvin ]
d - density [ g/cm^3 ]
p - ambient pressure [ hectoPascals ]
```

Note that the default physical units are NOT all in SI. In particular density is specified in g/cm^3 and pressure in hectoPascals. The default wind unit is m/s; however, for files in which the winds are given in km/s the user should specify `--wind_units kmpersec` on the command line.

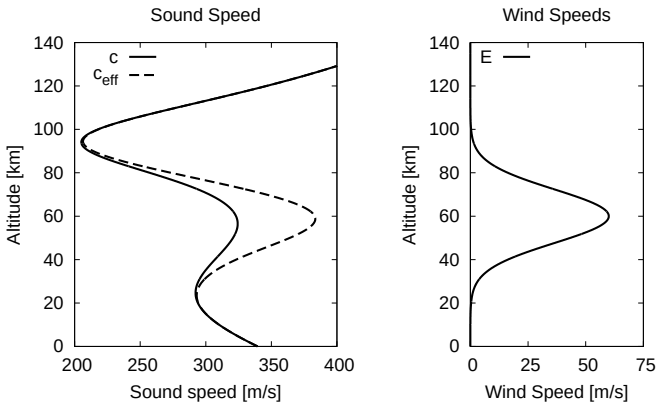

Figure 1: The NCPA model atmosphere, the "toy" model. The figure on the left shows sound speeds: the solid curve is the adiabatic sound speed and the dashed curve is the effective sound speed profile in the azimuth direction of 90 degrees (clockwise from North). The figure on the right shows the model zonal wind speed. The meridional wind speed is taken to be zero. The figures are taken from [11].

An example of an atmospheric profile file with the above structure is provided in the `samples` directory in file `NCPA_canonical_profile_zuvwtdp.dat`. Most tutorial examples included in this or the on-line help are based on this profile. The corresponding sound speed, effective sound speed for azimuth of 90^0 and wind speed are illustrated in Figure 1.

To model propagation through a stratified medium only a single specification file is required. Range dependent profiles are specified by providing a directory of atmospheric specification files. In addition, the reading of G2S binary files is supported; however, there is no guarantee that the structure of the G2S binary files will remain unchanged.

4 Modess - Modal Effective Sound Speed

Modess implements a normal mode expansion for propagation of a single tone in a stratified atmosphere in the effective sound speed approximation. Recall that in the effective sound speed approximation the influence of the atmospheric winds is approximated by adding the along-path component of the horizontal winds to the sound speed and then propagating through the resulting effective sound speed, c_{eff}, as if there were no wind. This approximation is valid for sufficiently low angle propagation paths and sufficiently low wind speeds (low Mach numbers). Attenuation by the atmosphere, implemented via the addition of an attenuation coefficient as the imaginary part of the wave number, is taken into account as a perturbation. The perturbative treatment of the atmospheric attenuation produces accurate results for tropospheric and stratospheric ducting, and for frequencies low enough that attenuation is not significant, but generally should not be considered accurate for signals returned from the thermosphere. The ground surface is assumed to be rigid.

4.1 Mathematical Background

Let z denote altitude above the ground surface and let the subscript H denote horizontal displacement. Let $r = \|\mathbf{x}_H\|$ be the radial horizontal range. Let ρ_0 be the mean density of the atmosphere, let α be the atmospheric attenuation coefficient, ω the angular frequency, and p the deviation from mean pressure at horizontal position $\mathbf{x}_H$ and altitude z. Then **Modess** solves the following generalized Helmholtz equation:

$$\Big[\nabla_H^2 + \rho_0 \frac{\partial}{\partial z}\Big(\frac{1}{\rho_0}\frac{\partial}{\partial z}\Big) + \Big(\frac{\omega}{c_{eff}(z)} + i\alpha(\omega, z)\Big)^2\Big] p(r, z, \omega) = 0. \tag{1}$$

The pressure deviation p satisfies the boundary condition

$$\frac{\partial p}{\partial z}\Big|_{z=0} = 0$$

on the ground surface. If the signal source is assumed to be at zero range, $r = 0$, and altitude z_S then p is asymptotic, for small distance $r, (z - z_S) \downarrow 0$, to the field produced by a unit acoustic source,

$$\lim_{r,(z-z_S)\downarrow 0} \Big(p(r, z, \omega) - \frac{1}{\sqrt{r^2 + (z - z_S)^2}}\Big) = 0.$$

The solution is obtained by the method of normal modes, see e.g. [5]. Computed is the modal sum for the far field pressure deviation

$$p(r, z, \omega) \approx \frac{e^{i\pi/4}}{\sqrt{8\pi r}} \sqrt{\frac{\rho_0(z)}{\rho_0(z_s)}} \sum_j \psi_j(z_s)\psi_j(z) \frac{e^{i(k_j + i\alpha_j)r}}{\sqrt{k_j}} \tag{2}$$

where the modal wave numbers k_j and mode functions ψ_j are real valued and k_j^2 are the eigenvalues and ψ_j the corresponding eigenfunctions of the eigenvalue problem

$$\Big(\frac{d^2}{dz^2} + \frac{\omega^2}{c_{eff}^2}\Big)\psi(z) = k^2\psi(z) \quad \text{with} \quad \psi'(0) = 0.$$

Note that terms related to buoyancy have been dropped. The modal attenuation coefficients α_j are estimated using leading order perturbation theory, see e.g. [6]. One finds

$$\alpha_j = c_j \int_0^\infty \frac{\alpha(\omega, z)}{c_{eff}(z)} \, (\psi_j(z))^2 \, dz.$$

Here $c_j = \frac{\omega}{k_j}$ is the modal phase velocity.

The eigenvalue problem is solved by replacing the continuous vertical domain with a discrete grid, using finite differences to approximate the derivatives with respect to z, truncating the vertical domain at some large altitude, T, and then solving the resulting finite dimensional eigenvalue problem. A uniformly spaced grid, with spacing h, is used and a nearest neighbor centered difference approximation is used for the second derivative:

$$\frac{d^2 f}{dz^2} \approx \frac{f(z-h) - 2f(z) + f(z+h)}{h^2}.$$

The resulting matrix is of the form

$$\mathbf{M} = \frac{1}{h^2} \begin{pmatrix} -1 - h^2 \frac{\omega^2}{c_{eff}(0)^2} & 1 & 0 & \dots & 0 \\ 1 & -2 - h^2 \frac{\omega^2}{c_{eff}(h)^2} & 1 & \dots & \vdots \\ 0 & 1 & -2 - h^2 \frac{\omega^2}{c_{eff}(2h)^2} & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & \dots & \dots & 1 & -2 - h^2 \frac{\omega^2}{c_{eff}(T)^2} \end{pmatrix}.$$

The eigenvalue problem reduces to a large dimensional linear algebra problem: find the eigenvalues k^2 and corresponding eigenvectors Ψ that satisfy

$$M\Psi = k^2 \Psi.$$

With this simple choice for the discrete approximation to the second derivative a very large number of vertical points are required, generally tens of thousands, to obtain accurate results; however, the resulting matices are tridiagonal, allowing for extremely efficient numerical algorithms. The benefits of having tridiagonal matrices to diagnalize turns out to outweigh the problems encountered in dealing with the resulting large matrices. The number of vertical points required has been determined by checking that the results converge. Generally, the number increases with increasing frequency, but also with increasing complexity of the effective sound speed. The default is chosen to be 20,000, which has been found to be sufficient for the NCPA toy atmosphere up to about 5 Hz.

Only a relatively small subset of the eigenvalues of the approximating matrix **M** are needed for the modal sum. These correspond to the phase velocities (and hence modal wave numbers) needed for the evaluation and convergence of the modal sum. Modes can propagate in regions of the atmosphere where their phase velocities are greater than the effective soundspeed. It follows that only wave numbers whose corresponding phase velocities are greater than the minimum of the effective soundspeed need to be considered. The maximum phase velocity is chosen large enough to insure convergence of the modal sum. In practice one increases the maximum phase velocity until convergence is achieved. In **Modess** one increases the maximum phase velocity by increasing the maximum altitude of the computational domain.

These criteria are depicted graphically in Fig. 2 for eastward propagation in the NCPA toy atmosphere. The minimum phase velocity is chosen so that all propagating modes are included in the modal sum. If the source and receiver altitudes are to be arbitrary then the minimum phase velocity corresponds to the minimum of the effective sound speed. This is depicted by the blue arrows in the figure. If either the source or receiver are at zero altitude then fewer modes are required, as depicted by the red arrows. In this case the minimum phase velocity is determined by estimating the modal penetration to the ground in the WKB approximation. Once the relevant range of phase velocities, and thus of eigenvalues of **M**, has been determined the eigenvalues and corresponding mode functions are computed using the Krylov algorithms in the **SLEPc** package.

Once the relevant wavenumbers and modes have been calculated, **Modess** computes the modal sum Equ. 2 and then saves the resulting $p(r, z, \omega)$ to files. Note that $p(r, z, \omega)$ depends on azimuth through the effective sound speed. $p(r, z, \omega)$ will be referred to as the (complex valued) transmission loss despite the fact

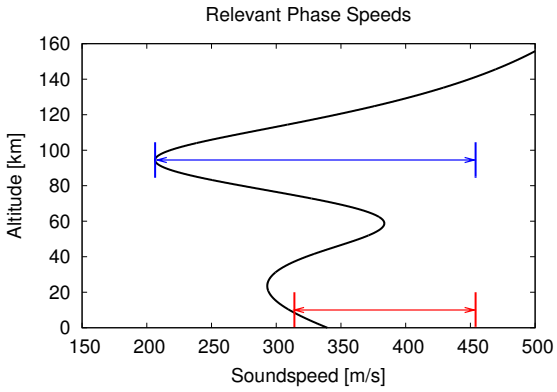

Figure 2: Phase velocities used in **Modess** for the NCPA toy atmosphere. The black curve is the effective sound speed for eastward propagation. The blue double-headed arrow shows the range of phase velocities needed in the modal sum. The red double-headed arrow shows the range of phase velocities needed for propagation to or from the ground. The minimum trace velocity is chosen so that all propagating modes are computed. The maximum trace velocity is chosen so that the modal sum converges in the far field.

that transmission loss is more generally defined to be the negative of the magnitude of $p(r, z, \omega)$ expressed in dB relative to 1. In addition to the complex valued transmission loss, **Modess** computes the incoherent transmission loss $I(r, z, \omega)$ defined by (see [5])

$$I(r, z, \omega) = \frac{1}{\sqrt{8\pi r}} \sqrt{\frac{\rho_0(z)}{\rho_0(z_s)}} \sqrt{\sum_j \frac{e^{-2\alpha_j r}}{k_j} \big(\psi_j(z_s)\psi_j(z)\big)^2}\,.$$

The incoherent transmission loss is the mean transmission loss in the approximation that the relative phases of the modal contributions are random. It provides a simple approximation for an average transmission loss in a fluctuating atmosphere.

4.2 Running Modess

Making sure that the executable for **Modess** is in the system's path, it can be run by entering

```
Modess [--option1 val1] [--option2 val2] [...] [--flag1] [...]
```

on a command line. Generally, options are followed by values, either numbers, strings or filenames. Flags are not. Entering `Modess` without any options or flags sends the following help page to the screen:

```
By default 'Modess' computes the 1D transmission loss (TL)
at the ground or the specified receiver height and saves the data to 2 files:
   file tloss_1d.nm - considering attenuation in the atmosphere
   file tloss_1d.lossless.nm  - no attenuation
Additionally, if the flag --write_2D_TLoss is present on the command line
the 2D TL is saved to file tloss2d.nm
The user can also choose to propagate in N different radial directions
i.e. (N by 2D mode) by using the option --Nby2Dprop .

The options below can be specified in a colon-separated file "Modess.options"
or at the command line. Command-line options override file options.
 --help -h                Print this message and exit
```

```
To use an arbitrary 1-D atmospheric profile in ASCII format
(space or comma-separated) the following options apply:

REQUIRED (no default values):
 --atmosfile  <filename>  Uses an ASCII atmosphere file
                          referenced to Mean Sea Level (MSL).
 --atmosfileorder         The order of the (z,u,v,w,t,d,p) fields
                          in the ASCII file (Ex: 'zuvwtdp')
                          The units assumed in the ASCII file are
                          z[km], t [kelvin], d [g/cm^3], p [hectoPa]
                          The wind speeds are in m/s by default;
                          however if the winds are given in km/s then use
                          option --wind_units kmpersec
 --skiplines              Lines at the beginning of the ASCII file to skip
 --freq                   Frequency [Hz]

REQUIRED for propagation in one direction (no default values):
 --azimuth                Degrees in range [0,360], clockwise from North

REQUIRED for propagation in N directions i.e. (N by 2D) (no default values):
 --azimuth_start          Start azimuth ([0,360] degrees, clockwise from North)
 --azimuth_end            End azimuth ([0,360] degrees, clockwise from North)
 --azimuth_step           Step by which the azimuth is changed (in degrees)

OPTIONAL [defaults]:
 --maxheight_km           Calculation grid height in km above MSL [150 km]
 --zground_km             Height of the ground level above MSL [ 0 km ]
 --Nz_grid                Number of points on the z-grid from ground to maxheight [20000]
 --sourceheight_km        Source height in km Above Ground Level (AGL) [0]
 --receiverheight_km      Receiver height in km AGL [ 0 ]
 --maxrange_km            Maximum horizontal propagation distance from origin [1000 km]
 --Nrng_steps             Number of range steps to propagate [1000]
 --ground_impedance_model Name of the ground impedance models to be employed:
                          [rigid], others TBD
 --Lamb_wave_BC           If ==1 it sets admittance = -1/2*dln(rho)/dz; [ 0 ]
 --wind_units             Specify 'kmpersec' if the winds are given
                          in km/s [ mpersec ]
 --use_attn_file          Option to specify a file name containing user-provided
                          attenuation coefficients to be loaded instead of
                          the default Sutherland-Bass attenuation.
                          The text file should contain two columns:
                              height (km AGL) and
                              attenuation coefficients in np/m.
 --modal_starter_file     Specifies the file name of the modal starter.
                          It can be used as a starter in the PE module (pape).
                          The columns are: | z_km | Real(scP) | Imag (scP) |
                          where scP is the scaled pressure such that it can
                          be directly ingested into 'pape' to obtain the
                          transmission loss.

FLAGS (no value required):
 --write_2D_TLoss         Outputs the 2D transmission loss to
                          default file: tloss2D.nm
```

```
 --write_phase_speeds     Output the phase speeds to
                          default file: phasespeeds.nm
 --write_speeds           Output both the phase speeds and the group speeds to
                          default file: speeds.nm
 --write_modes            Output the modes to default files:
                          mode_<mode_count>.nm
                          Also outputs the modal phase and group speeds to
                          default file 'speeds.nm
 --write_dispersion       Output the mode dispersion to
                          default file: dispersion_<freq>.nm
 --Nby2Dprop              Flag to perform (N by 2D) propagation i.e.
                          propagation in N directions specified by
                          options: azimuth_start, azimuth_end, azimuth_step
                          The output is saved into default files:
                              Nby2D_tloss_1d.lossless.nm
                              Nby2D_tloss_1d.nm
 --write_atm_profile      Save the interpolated atm. profile to
                          default file: atm_profile.nm
 --turnoff_WKB            Turn off the WKB least phase speed estimation
                          an approx. that speeds-up ground-to-ground propag.
                          It has the value 1 (true) if any of the flags
                          write_2D_TLoss, write_phase_speeds, write_modes
                          or write_dispersion are true.


OUTPUT Files -  Format description (column order):
  tloss_1d.nm:            r, 4*PI*Re(P), 4*PI*Im(P), (incoherent TL)
  tloss_1d.lossless.nm:
  tloss_2d.nm:            r, z, 4*PI*Re(P), 4*PI*Im(P)
  Nby2D_tloss_1d.nm:      r, theta, 4*PI*Re(P), 4*PI*Im(P), (incoherent TL)
  Nby2D_tloss_1d.lossless.nm:

  phasespeeds.nm:         Mode#, phase speed [m/s], imag(k)
  speeds.nm:              Mode#, phase speed [m/s], group speed [m/s], imag(k)
  mode_<mode_count>.nm    z, (Mode amplitude)
  dispersion_<freq>.nm    Contains one line with entries: freq (# of modes)
                          followed for each mode 'i' by quadruples:
                          (real(k(i)) imag(k(i)) Mode(i)(z_src) Mode(i)(z_rcv)
  atm_profile.nm          z,u,v,w,t,d,p,c,c_eff

 Examples (run from 'samples' directory):
    ../bin/Modess --atmosfile NCPA_canonical_profile_zuvwtdp.dat
                  --atmosfileorder zuvwtdp --azimuth 90 --freq 0.1

    ../bin/Modess --atmosfile NCPA_canonical_profile_zuvwtdp.dat
                  --atmosfileorder zuvwtdp --azimuth 90 --freq 0.1 --write_2D_TLoss

    ../bin/Modess --atmosfile NCPA_canonical_profile_zuvwtdp.dat
                  --atmosfileorder zuvwtdp --freq 0.1 --Nby2Dprop
                  --azimuth_start 0 --azimuth_end 360 --azimuth_step 1
```

To use **Modess** an ascii file containing the atmospheric specifications must be loaded using `--atmosfile` followed by the filename. The order of the columns in the specification file needs to be specified using `--atmosfileorder` followed by a string indicating the order; eg. zuvwtdp if the columns orders are altitude,

zonal wind, meridional wind, vertical wind, temperature, density and pressure. Units are as indicated. The option `--skiplines`, followed by a non-negative integer, is generally used to skip any header lines that might be in the file. **Modess** always functions at a single frequency, specified using the `--freq` option.

By default **Modess** writes the transmission loss at a fixed azimuth as a function of range to a receiver on the ground, $p(r, 0, \omega)$, to a file named `tloss_1d.nm`; the azimuth must be specified using `--azimuth` followed by an angle in degrees. This is referred to as the 1D transmission loss. For use in testing and debugging, the lossless 1D transmission loss is also written to a file named `tloss_1d.lossless.nm`. Receiver altitudes other than $z = 0$ can be set by using the `--receiverheight_km` option. Similarly, the source altitude can be set using `--sourceheight_km`. In general, `tloss_1d.*` files have four columns: range r, Re p, Im p, and incoherent transmission loss I.

If the flag `--write_2D_TLoss` is set then the 2D transmission loss, by which we understand the transmission loss as a function of range and altitude, is written to the file `tloss_2d.nm`. This file has four columns; r, z, Re p, Im p in that order; written in line-separated blocks of constant r. The 2D transmission loss magnitude is useful in identifying the various propagation paths and their relative magnitudes. The propagation paths are obscured by the incoherent transmission loss and so is not very informative as a function of range and altitude and is not written to the file.

The other flag that is directly related to propagation in `--Nby2Dprop`. If this flag is set then the 1D transmission losses are calculated for a range of azimuths θ and then saved to the file `Nby2Dtloss_1d.nm` and `Nby2Dtloss_1d.lossless.nm`. The range of azimuths is specified using the options `--azimuth_start`, `--azimuth_end`, and `--azimuth_step`, each followed by an angle. `--Nby2Dprop` is used to model propagation from a source location to a portion of the horizontal plane, for a fixed receiver altitude. The files `Nby2Dtloss_1d.*` have five columns; r, θ, Re p, Im p, and I; written in line separated blocks of constant r. If `--Nby2Dprop` is set then it is advised that `--write_2D_TLoss` not be set.

It is critical that `--maxheight_km` be less than the maximum altitude contained in the atmospheric specification file. Otherwise the program will crash. Most of the other options and flags are self explanatory. Exceptions are `--ground_impedance_model` which is designed to allow for an impedance boundary condition on the ground (this option is not yet supported), `--Lamb_wave_BC` followed by 0 or 1 which implements the Lamb wave impedance condition when it has the value 1 (this is the lowest order buoyant effect), `--use_attn_file` followed by a file name which allows the user to input an arbitrary attenuation profile through a user-supplied ascii file, and finally `--modal_starter_file` followed by a file name which writes a modal starter to the given file to be used with the package's PE model.

4.3 Running Modess: examples

The **Modess** help page ends with three examples. The examples set the frequency to 0.1 Hz to keep run times short. The first is a simple example illustrating the primary input modes for Modess. It is assumed that the user runs it in the `samples` directory. Note that if the **ncpaprop** `bin` directory is in the system's path one may enter `Modess` rather than `../bin/Modess`. The command line entry for the example is

```
../bin/Modess --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
              --atmosfileorder zuvwtdp --azimuth 90 --freq 0.1
```

In this example the four required options are set. Otherwise the default settings are used. Ground-to-ground propagation in the NCPA toy atmosphere (see Figure 1) at 90 degrees azimuth (from North), eastward propagation, at frequency of 0.1 Hz is modelled. By default the signal is propagated out to 1000 km in range with range steps of 1 km. The atmospheric profiles are specified in the column-based text file `NCPA_canonical_profile_zuvwtdp.dat` that is included in the `samples` directory. In the above example the profile file has 7 columns in the order `zuvwtdp` (refer to section 3.2 for an explanation of atmospheric specifications).

The output is, by default, the one-dimensional (1D) transmission loss on the ground for both lossy and lossless cases and is written into the text files `tloss_1d.nm` and `tloss_1d.lossless.nm` as described above

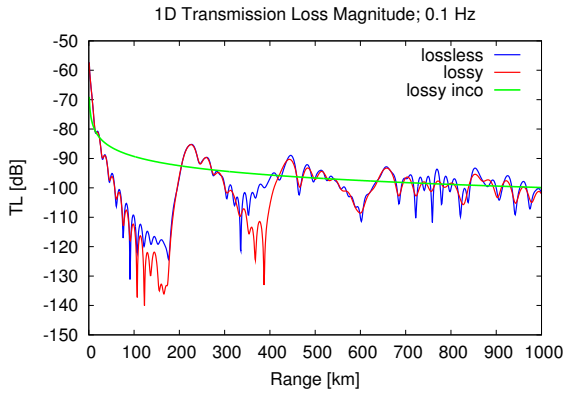

Figure 3: 1D transmission loss magnitude at 0.1 Hz obtained with **Modess** for eastward ground-to-ground propagation in the NCPA toy atmosphere shown in Figure 1. Shown are the lossless transmission loss magnitude, the lossy transmission loss magnitude and the lossy incoherent transmission loss.

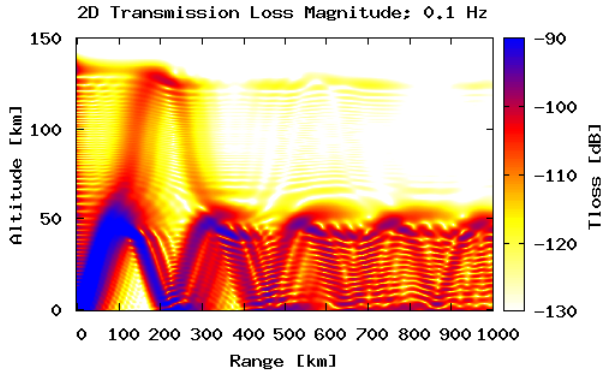

Figure 4: 2D transmission loss magnitude at 0.1 Hz obtained with **Modess** for eastward propagation in the NCPA toy model. The source is placed on the ground at the origin.

in section 4.2. An example of the magnitude of the complex 1D transmission loss output is shown in Figure 3. Note that in general the output from all programs will be saved in the directory from which the code is run.

In the second example the `--write_2D_Tloss` flag is appended to the command line entry for the first example:

```
../bin/Modess --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
              --atmosfileorder zuvwtdp --azimuth 90 --freq 0.1 --write_2D_TLoss
```

As described above in section 4.2, the output is written to the file `tloss2d.nm`. The 2D transmission loss magnitude that results from eastward propagation in the NCPA toy model is plotted in figure 4.

In the third example the `--Nby2Dprop` flag is appended to the command line entry for the first example. This flag requires that values for the options `--azimuth_start`, `--azimuth_end`, and `--azimuth_step` be given. The command line entry is

```
../bin/Modess --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
              --atmosfileorder zuvwtdp --freq 0.1 --Nby2Dprop \
              --azimuth_start 0 --azimuth_end 360 --azimuth_step 1
```

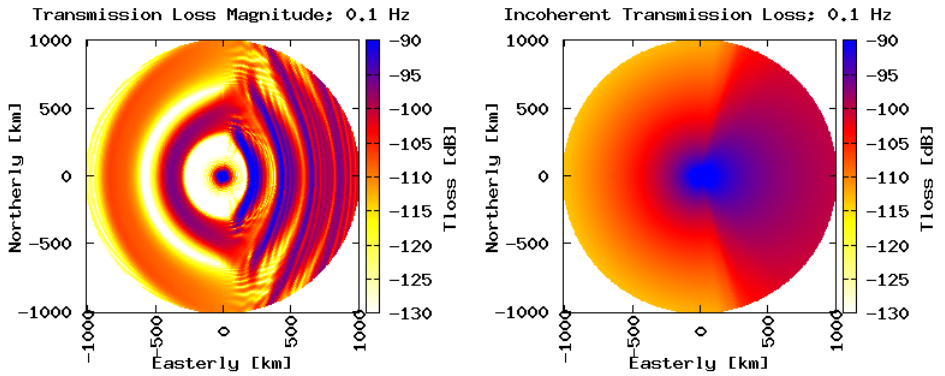

Figure 5: N by 2D transmission loss magnitude and incoherent transmission loss obtained with **Modess** for ground-to-ground propagation in the NCPA toy model atmosphere from a source at the origin to all azimuths.

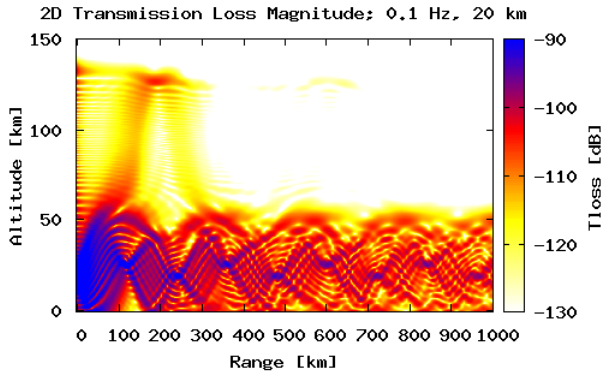

Figure 6: 2D transmission loss magnitude for propagation from an elevated source at 20 km altitude to the ground, obtained with **Modess** for propagation in the NCPA toy atmosphere.

The output is written to the files `Nby2D_tloss_1d.nm` and `Nby2D_tloss_1d.lossless.nm` as described above in section 4.2. That the lossless transmission losses are also computed and saved as are the incoherent transmission losses. The resulting transmission loss magnitude is plotted in figure 5.

A final example, not included in the **Modess** help page consider the calculation of a 2D transmission loss from an elevated source. For eastward propagation in the NCPA toy atmosphere from a source at 20 km altitude the command line entry is

```
../bin/Modess --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
              --atmosfileorder zuvwtdp --azimuth 90 --freq 0.1 --write_2D_TLoss \
              --sourceheight_km 20
```

The resulting 2D transmission loss magnitude is plotted in figure 6. Note that the calculation only goes out to a range of 1 km.

These examples can be run using a `.options` file rather than the full command line entry. Entering `Modess` without any arguments writes the help page to the screen even if there is a `.options` file present so that at least one argument must be entered as well. If, for example,

```
Modess --write_2D_TLoss
```

is entered on the command line then the `Modess.options` file for the last example, that of the calculation of the 2D transmission loss for an elevated source, should contain the following entries:

```
atmosfile : NCPA_canonical_profile_zuvwtdp.dat
atmosfileorder : zuvwtdp
skiplines : 0
azimuth : 90
freq : 0.1
sourceheight_km : 20
```

5 WMod - Wide-Angle High-Mach Modal Code

WMod implements a normal mode expansion for propagation of a single tone in a stratified atmosphere. In contrast to **Modess**, **Wmod** does not use the effective sound speed approximation but rather treats the background wind rigorously in the planar (2-d) approximation, with the assumption that vertical wind shear is small over the scale of a wavelength and can thus be neglected. Since it does not use the effective sound speed approximation, **WMod** can accurately model the high propagation angles required for long range propagation that depends on refraction from the upper atmosphere and can, in principle, be used to model propagation under conditions of arbitrarily high background wind speeds. **WMod** represents a significant improvement over **Modess**; however, computation times are much longer. As in **Modess**, atmospheric attenuation in bf Wmod is taken into account using first order perturbation theory.

5.1 Mathematical Background

As in section 4.1 let z denote altitude above the ground surface and let the subscript H denote horizontal displacement. Let $r = \|\mathbf{x}_H\|$ be the radial horizontal range. Let ρ_0 be the mean density of the atmosphere, let α be the atmospheric attenuation coefficient, ω the angular frequency, and p the deviation from mean pressure at horizontal position $\mathbf{x}_H$ and altitude z. Then **WMod** solves the following generalized Helmholtz equation:

$$\Big[\nabla_H^2 + \rho_0 \frac{\partial}{\partial z}\Big(\frac{1}{\rho_0}\frac{\partial}{\partial z}\Big) + \Big(\frac{\omega}{c(z)} + i\frac{\mathbf{v}_{0,H}}{c(z)} \cdot \nabla_H + i\alpha(\omega, z)\Big)^2\Big]\hat{p}_A(r, z, \omega) = 0. \tag{3}$$

The pressure deviation p satisfies the boundary condition

$$\frac{\partial p}{\partial z}\Big|_{z=0} = 0$$

on the ground surface. If the signal source is assumed to be at zero range, $r = 0$, and altitude z_S then p is asymptotic, for small distance $r, (z - z_S) \downarrow 0$, to the field produced by a unit acoustic source,

$$\lim_{r,(z-z_S)\downarrow 0} \Big(p(r, z, \omega) - \frac{1}{\sqrt{r^2 + (z - z_S)^2}}\Big) = 0.$$

As for **Modess**, the solution is again obtained by the method of normal modes. Computed is the modal sum for the far field pressure deviation

$$p(r, z, \omega) \approx \frac{e^{i\pi/4}}{\sqrt{8\pi r}} \sqrt{\frac{\rho_0(z)}{\rho_0(z_s)}} \sum_j \psi_j(z_s)\psi_j(z) \frac{e^{i(k_j + i\alpha_j)r}}{\sqrt{k_j}},$$

precisely as in equation 2. As for **Modess** the modal wave numbers k_j and mode functions ψ_j are real valued; however, k_j (rather than k_j^2) are now the eigenvalues and ψ_j the corresponding eigenfunctions of the quadratic (since k appears quadratically) eigenvalue problem

$$\Big(\frac{d^2}{dz^2} + \frac{\omega^2}{c^2}\Big)\psi(z) = \Big((1 - \nu^2)k^2 + 2\nu\frac{\omega}{c}k\Big)\psi(z) \quad \text{with} \quad \psi'(0) = 0$$

where, for $\hat{\mathbf{k}}_H$ the unit vector in the horizontal direction of propagation, $\nu = \hat{\mathbf{k}}_H \cdot \frac{\mathbf{v}_{0,H}}{c}$. The modal attenuation coefficients are estimated perturbatively as in section 4.1.

The eigenvalue problem is solved as in section 4.1 by replacing the continuous vertical domain with a uniformly spaced discrete grid, using finite differences to approximate the derivatives with respect to z, truncating the vertical domain at some large altitude, T, and then solving the resulting finite dimensional eigenvalue problem. Using the notation introduced in section 4.1 the eigenvalue problem reduces to a large dimensional linear algebra problem: find the eigenvalues k and corresponding eigenvectors Ψ that satisfy

$$M\Psi = Ak^2\Psi + Bk\Psi$$

where A and B are diagonal matrices with entries in the n, n position given by $1 - \nu(nh)^2$ and $2\nu(nh)\frac{\omega}{c(nh)}$ respectively.

As with the effective sound speed approximation, only a small subset of the set of the eigenvalues is required; however, determining the relevant range of eigenvalues is not as straightforward for the quadratic eigenvalue problem being considered here as it is for the linear eigenvalue problem associated with the effective sound speed. In practice, it has been found sufficient for the relevant phase velocities to be chosen as in the effective soundspeed case, as depicted in Fig. 2. Once the range of phase velocities has been determined, the eigenvalues and corresponding mode functions are computed using the quadratic eigenvalue solver contained in the **SLEPc** package.

5.2 Running WMod

Making sure that the executable for **Wmod** is in the system's path, it can be run by entering

```
WMod [--option1 val1] [--option2 val2] [...] [--flag1] [...]
```

on a command line. Generally, options are followed by values, either numbers, strings or filenames. Flags are not. Entering `WMod` without any options or flags sends the following help page to the screen:

```
By default 'WMod' computes the 1D transmission loss (TL)
at the ground and saves the data to 2 files:
   file wtloss_1d.nm - considering attenuation in the atmosphere
   file wtloss_1d.lossless.nm  - no attenuation
Additionally, if the flag --write_2D_TLoss is present on the command line
the 2D TL is saved to file wtloss2d.nm
The user can also choose to propagate in N different radial directions
i.e. (N by 2D mode) by using the option --Nby2Dprop .

The options below can be specified in a colon-separated file "WMod.options"
or at the command line.  Command-line options override file options.
 --help -h                Print these instructions and exit

To use an arbitrary 1-D atmospheric profile in ASCII format
(space or comma-separated) the following options apply:

REQUIRED (no default values):
 --atmosfile  <filename>  Uses an ASCII atmosphere file
                          referenced to Mean Sea Level (MSL).
 --atmosfileorder         The order of the (z,t,u,v,w,p,d) fields
                          in the ASCII file (Ex: 'zuvwtdp')
                          The units assumed in the ASCII file are
                          z[km], t [kelvin], d [g/cm^3], p [hectoPa]
                          The wind speeds are in m/s by default;
                          however if the winds are given in km/s then use
                          option --wind_units kmpersec
 --skiplines              Lines at the beginning of the ASCII file to skip [0]
 --azimuth                Degrees in range [0,360), clockwise from North
 --freq                   Frequency [Hz]

REQUIRED for propagation in one direction (no default values):
 --azimuth                Degrees in range [0,360], clockwise from North

REQUIRED for propagation in N directions i.e. (N by 2D) (no default values):
```

```
 --azimuth_start          Start azimuth ([0,360] degrees, clockwise from North)
 --azimuth_end            End azimuth ([0,360] degrees, clockwise from North)
 --azimuth_step           Step by which the azimuth is changed (in degrees)

OPTIONAL [defaults]:
 --maxheight_km           Calculation grid height in km above MSL [150 km]
 --zground_km             Height of the ground level above MSL [0 km]
 --Nz_grid                Number of points on the z-grid from ground to
                         maxheight [20000]
 --sourceheight_km        Source height in km Above Ground Level (AGL) [0]
 --receiverheight_km      Receiver height in km AGL [0]
 --maxrange_km            Maximum horizontal propagation distance from origin
                          [1000 km]
 --Nrng_steps             Number of range steps to propagate [1000]
 --ground_impedance_model Name of the ground impedance models to be employed:
                          [rigid], others TBD
 --Lamb_wave_BC           If ==1 it sets admittance = -1/2*dln(rho)/dz; [ 0 ]
 --wind_units             Use it to specify 'kmpersec' if the winds are given
                          in km/s [mpersec]
 --use_attn_file          Use it to specify a file name containing user-provided
                          attenuation coefficients to be loaded instead of
                          the default Sutherland-Bass attenuation.
                          The text file should contain two columns:
                              height (km AGL) and
                              attenuation coefficients in np/m.

FLAGS (no value required):
 --write_2D_TLoss         Outputs the 2D transmission loss to
                          default file: wtloss2D.nm
 --write_phase_speeds     Output the phase speeds to
                          default file: wphasespeeds.nm
 --write_modes            Output the modes to default files:
                          wmode_<mode_count>.nm
 --write_dispersion       Output the mode dispersion to
                          default file: wdispersion_<freq>.nm
 --Nby2Dprop              Flag to perform (N by 2D) propagation i.e.
                          propagation in N directions specified by
                          options: azimuth_start, azimuth_end, azimuth_step
                          The output is saved into default files: " );
                              Nby2D_tloss_1d.lossless.nm" );
                              Nby2D_tloss_1d.nm" );

 --write_atm_profile      Save the interpolated atm. profile to
                          default file: atm_profile.nm
 --turnoff_WKB            Turn off the WKB least phase speed estimation
                          an approx. that speeds-up ground-to-ground propag.
                          It has the value 1 (true) if any of the flags
                          write_2D_TLoss, write_phase_speeds, write_modes
                          or write_dispersion are true.


OUTPUT Files -  Format description (column order):
  wtloss_1d.nm:           r, 4*PI*Re(P), 4*PI*Im(P), (incoherent TL)
  wtloss_1d.lossless.nm:
```

```
 wtloss_2d.nm:           r, z, 4*PI*Re(P), 4*PI*Im(P)
 Nby2D_wtloss_1d.nm:     r, theta, 4*PI*Re(P), 4*PI*Im(P), (incoherent TL)
 Nby2D_wtloss_1d.lossless.nm:

 wphasespeeds.nm:         Mode#, phase speed [m/s], imag(k)
 wmode_<mode_count>.nm    z, (Mode amplitude)
 wdispersion_<freq>.nm    Contains one line with entries: freq (# of modes)
                          followed for each mode i by quadruples:
                          (real(k(i)) imag(k(i)) Mode(i)(z_src) Mode(i)(z_rcv)
 atm_profile.nm           z,u,v,w,t,d,p,c,c_eff

Examples (run from 'samples' directory):

   ../bin/WMod --atmosfile NCPA_canonical_profile_zuvwtdp.dat
   --atmosfileorder zuvwtdp --skiplines 0 --azimuth 90 --freq 0.1

   ../bin/WMod --atmosfile NCPA_canonical_profile_zuvwtdp.dat
   --atmosfileorder zuvwtdp --skiplines 0 --azimuth 90 --freq 0.1
   --write_2D_TLoss --sourceheight_km 60 --receiverheight_km 60

   ../bin/WMod --atmosfile NCPA_canonical_profile_zuvwtdp.dat
   --atmosfileorder zuvwtdp --freq 0.1 --Nby2Dprop
   --azimuth_start 0 --azimuth_end 360 --azimuth_step 1
```

The options and flags for **WMod** are the same as for **Modess**. The reader is referred to the **Modess** documentation above for explanations. The only difference is that the output filenames now have a `w` preceding `tloss` in each case.

5.3 Running WMod: examples

The **WMod** help page ends with three examples. The examples set the frequency to 0.1 Hz to keep run times short. The first is a simple example illustrating the primary input modes for **WMod**. It is assumed that the user runs it in the `samples` directory. Note that if the **ncpaprop** `bin` directory is in the system's path one may enter `WMod` rather than `../bin/WMod`. The command line entry for the example is

```
../bin/WMod --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
             --atmosfileorder zuvwtdp --skiplines 0 --azimuth 90 --freq 0.1
```

In this example the four required options are set. Otherwise the default settings are used. Ground-to-ground propagation in the NCPA toy atmosphere (see Figure 1) at 90 degrees azimuth (from North), eastward propagation, at frequency of 0.1 Hz is modelled. By default the signal is propagated out to 1000 km in range with range steps of 1 km. The atmospheric profiles are specified in the column-based text file `NCPA_canonical_profile_zuvwtdp.dat` that is included in the `samples` directory. In the above example the profile file has 7 columns in the order `zuvwtdp` (refer to section 3.2 for an explanation of atmospheric specifications).

The output is, by default, the one-dimensional (1D) transmission loss on the ground for both lossy and lossless cases and is written into the text files `wtloss_1d.nm` and `wtloss_1d.lossless.nm` as described above in the `WMod` help page; see section 5.2 and section 4.2 for more discussion. An example of the magnitude of the complex 1D transmission loss output is shown in Figure 7. Note that in general the output from all programs will be saved in the directory from which the code is run.

In the second example the `--write_2D_Tloss` flag is appended to the command line entry for the first example:

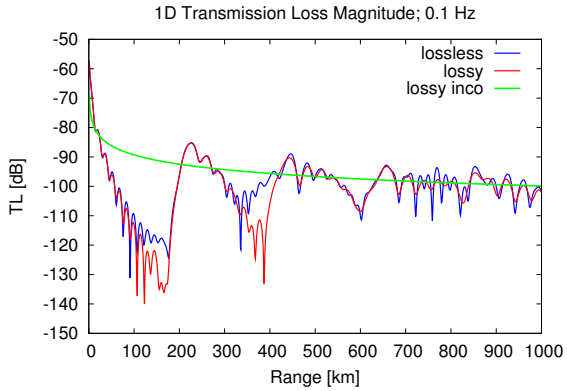

Figure 7: 1D transmission loss magnitude at 0.1 Hz obtained with **WMod** for eastward ground-to-ground propagation in the NCPA toy atmosphere shown in Figure 1. Shown are the lossless transmission loss magnitude, the lossy transmission loss magnitude and the lossy incoherent transmission loss.

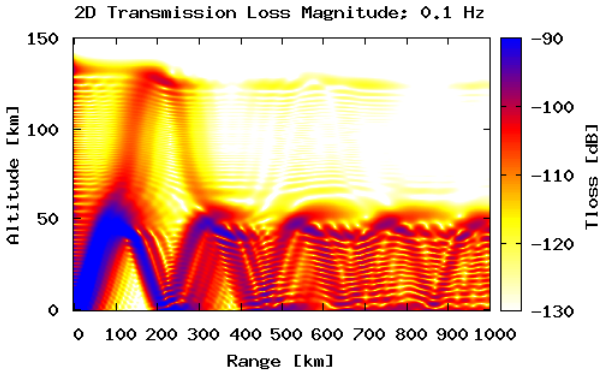

Figure 8: 2D transmission loss magnitude at 0.1 Hz obtained with **WMod** for eastward propagation in the NCPA toy model. The source is placed on the ground at the origin.

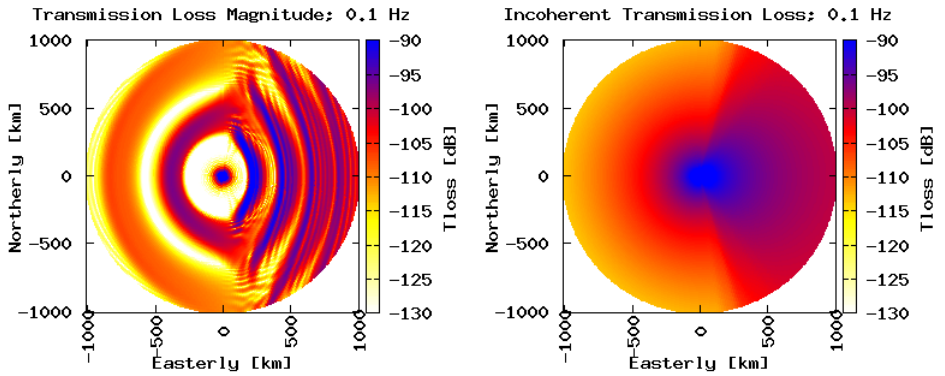

Figure 9: N by 2D transmission loss magnitude and incoherent transmission loss obtained with **WMod** for ground-to-ground propagation in the NCPA toy model atmosphere from a source at the origin to all azimuths.

```
../bin/WMod --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
            --atmosfileorder zuvwtdp --skiplines 0 --azimuth 90 \
            --freq 0.1 --write_2D_TLoss
```

As described above and in section 4.2, the output is written to a default file, in this case `wtloss2d.nm`. The 2D transmission loss magnitude that results from eastward propagation in the NCPA toy model is plotted in figure 8.

In the third example the `--Nby2Dprop` flag is appended to the command line entry for the first example. This flag requires that values for the options `--azimuth_start`, `--azimuth_end`, and `--azimuth_step` be given.

```
../bin/WMod --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
              --atmosfileorder zuvwtdp --freq 0.1 --Nby2Dprop \
              --azimuth_start 0 --azimuth_end 360 --azimuth_step 1
```

The output is written to the files `Nby2D_wtloss_1d.nm` and `Nby2D_wtloss_1d.lossless.nm` as described above in section 5.2. The lossless transmission losses are also computed and saved as are the incoherent transmission losses. The resulting transmission loss magnitude is plotted in figure 9. This example can be run with the command `WMod --Nby2Dprop` if the `bin` directory is in the system path and the working directory contains the file `WMod.options` with the entries

```
atmosfile : NCPA_canonical_profile_zuvwtdp.dat
atmosfileorder : zuvwtdp
skiplines : 0
azimuth : 90
freq : 0.1
sourceheight_km : 0
azimuth_start : 0
azimuth_end : 360
azimuth_step : 1
```

6 CModess - Complex Modal Effective Sound Speed

CModess implements a normal mode expansion for propagation of a single tone in a stratified atmosphere, over a rigid ground surface, in the effective sound speed approximation. The difference between **CModess** and **Modess** is that attenuation by the atmosphere, implemented as for **Modess** via the addition of an attenuation coefficient as the imaginary part of the wave number, is here treated exactly by solving the non-self-adjoint eigenvalue problem for the vertical modes. As a consequence the resulting mode functions and eigenvalues are complex valued.

Solving non-self-adjoint eigenvalue problem is known to be difficult. Since eigenvalues are complex valued searches must be performed in two dimensions in the complex plane rather than in one dimension on a line segment. Further complicating matters, the solutions of the underlying differential equation grow and decay exponentially due to the complex valued eigenvalue parameter. This can lead to numerical instabilities. As a consequence, **CModess** is not as robust as **Modess**. Convergence is not always guaranteed, particularly for larger frequencies (expect problems at 0.5 Hz and above), and when **CModess** does converge runtimes are long. It is recommended that **CModess** be used primarily to cross testing and validation.

6.1 Mathematical Background

CModess solves the same generalized Helmholtz equation, equation 1, as does **Modess** (see 4.1). The difference is that the modal sum is now

$$p(r,z,\omega) \approx \frac{e^{i\pi/4}}{\sqrt{8\pi r}}\sqrt{\frac{\rho_0(z)}{\rho_0(z_s)}}\sum_j \psi_j(z_s)\psi_j(z)\frac{e^{i(k_j+i\alpha_j)r}}{\sqrt{k_j+i\alpha_j}}$$

and the modal wave numbers, to be written here as $\kappa_j = k_j + i\alpha_j$, and mode functions ψ_j are complex valued with real and imaginary parts given by k_j and α_j respectively. Then κ_j^2 are the eigenvalues and ψ_j the corresponding eigenfunctions of the non-self-adjoint eigenvalue problem

$$\Big(\frac{d^2}{dz^2} + \big(\frac{\omega}{c_{eff}} + i\alpha\big)^2\Big)\psi(z) = \kappa^2\psi(z) \quad \text{with} \quad \psi'(0) = 0.$$

Phase velocities are defined as before, from the real part of the wave number: $c_j = \frac{\omega}{k_j}$.

It is to be noted that for non-self-adjoint problems there is no guarantee that there is an associated normal mode expansion. However, for the type of problem considered here, that of a second order ordinary differential equation on a half line, it has been shown that there is indeed an associated modal expansion [3].

As for **Modess**, the eigenvalue problem is solved by replacing the continuous vertical domain with a discrete uniformly spaced grid with spacing h. Finite differences are used to approximate the derivatives with respect to z and the vertical domain is truncated at some large altitude, T. The resulting finite dimensional eigenvalue problem is then solved. Letting $K(z) = \frac{\omega}{c_{eff}(z)} + i\alpha(\omega, z)$ the resulting matrix is of the form

$$\mathbf{M} = \frac{1}{h^2}\begin{pmatrix} -1-h^2K(0)^2 & 1 & 0 & \dots & 0 \\ 1 & -2-h^2K(h)^2 & 1 & \dots & \vdots \\ 0 & 1 & -2-h^2K(2h)^2 & \dots & \vdots \\ \vdots & \vdots & \vdots & \ddots & 1 \\ 0 & \dots & \dots & 1 & -2-h^2K(T)^2 \end{pmatrix}.$$

The eigenvalue problem reduces to a large dimensional, non-Hermitian, linear algebra problem: find the eigenvalues κ^2 and corresponding eigenvectors Ψ that satisfy

$$M\Psi = \kappa^2\Psi.$$

As with the Hermitian eigenvalue problem associated with the perturbative approximation used for **Modess**, only a small subset of the set of the eigenvalues is required; however, determining the relevant range of eigenvalues is not as straightforward for the non-Hermitian eigenvalue problem being considered here as it is for the Hermitian problem needed for **Modess**. In practice, it has been found sufficient for the relevant phase velocities to be chosen as in the effective soundspeed case, as depicted in Fig. 2. The search for the imaginary parts of the wave numbers is commenced at zero. Once the range of phase velocities has been determined, the eigenvalues and corresponding mode functions are computed using the non-Hermitian eigenvalue solver contained in the **SLEPc** package.

6.2 Running CModess

Making sure that the executable for **CModess** is in the system's path, it can be run by entering

```
CModess [--option1 val1] [--option2 val2] [...] [--flag1] [...]
```

on a command line. Generally, options are followed by values, either numbers, strings or filenames. Flags are not. Entering `CModess` without any options or flags sends the following help page to the screen:

```
By default 'CModess' computes the 1D transmission loss (TL)
at the ground or the specified receiver height and saves the data to 2 files:
   file tloss_1d.cnm - considering attenuation in the atmosphere
   file tloss_1d.lossless.cnm  - no attenuation
Additionally, if the flag --write_2D_TLoss is present on the command line
the 2D TL is saved to file tloss2d.cnm
The user can also choose to propagate in N different radial directions
i.e. (N by 2D mode) by using the option --Nby2Dprop .

The options below can be specified in a colon-separated file "CModess.options"
or at the command line. Command-line options override file options.
 --help -h                Print this message and exit

To use an arbitrary 1-D atmospheric profile in ASCII format
(space or comma-separated) the following options apply:

REQUIRED (no default values):
 --atmosfile  <filename>  Uses an ASCII atmosphere file
 --atmosfileorder         The order of the (z,u,v,w,t,d,p) fields
                          in the ASCII file (Ex: 'zuvwtdp')
                          The units assumed in the ASCII file are
                          z[km], t [kelvin], d [g/cm^3], p [hectoPa]
                          The wind speeds are in m/s by default;
                          however if the winds are given in km/s then use
                          option --wind_units kmpersec
 --skiplines              Lines at the beginning of the ASCII file to skip
 --freq                   Frequency [Hz]

REQUIRED for propagation in one direction (no default values):
 --azimuth                Degrees in range [0,360], clockwise from North
```

```
REQUIRED for propagation in N directions i.e. (N by 2D) (no default values):
 --azimuth_start          Start azimuth ([0,360] degrees, clockwise from North)
 --azimuth_end            End azimuth ([0,360] degrees, clockwise from North)
 --azimuth_step           Step by which the azimuth is changed (in degrees)

OPTIONAL [defaults]:
 --maxheight_km           Calculation grid height in km above MSL [150 km]
 --zground_km             Height of the ground level above MSL [0 km]
 --Nz_grid                Number of points on the z-grid from ground to maxheight
                          [20000]
 --sourceheight_km        Source height in km Above Ground Level (AGL) [0]
 --receiverheight_km      Receiver height in km AGL [0]
 --maxrange_km            Maximum horizontal distance from origin to propagate
                          [1000 km]
 --Nrng_steps             Number of range steps to propagate [1000]
 --ground_impedance_model Name of the ground impedance models to be employed:
                          [rigid], others TBD
 --Lamb_wave_BC           If ==1 it sets admittance = -1/2*dln(rho)/dz; [ 0 ]
 --wind_units             Use it to specify 'kmpersec' if the winds are given
                          in km/s [mpersec]
 --use_attn_file          Use it to specify a file name containing user-provided
                          attenuation coefficients to be loaded instead of
                          the default Sutherland-Bass attenuation.
                          The text file should contain two columns:
                              height (km AGL) and
                              attenuation coefficients in np/m.

FLAGS (no value required):
 --write_2D_TLoss         Outputs the 2D transmission loss to
                          default file: tloss2D.cnm
 --write_phase_speeds     Output the phase speeds to
                          default file: phasespeeds.cnm
 --write_modes            Output the modes to default files:
                          mode_<mode_count>.cnm
 --write_dispersion       Output the mode dispersion to
                          default file: dispersion_<freq>.cnm
 --Nby2Dprop              Flag to perform (N by 2D) propagation i.e.
                          propagation in N directions specified by
                          options: azimuth_start, azimuth_end, azimuth_step
                          The output in saved into default files:
                              Nby2D_tloss_1d.lossless.cnm
                              Nby2D_tloss_1d.cnm
 --write_atm_profile      Save the inteprolated atm. profile to
                          default file: atm_profile.cnm
 --turnoff_WKB            Turn off the WKB least phase speed estimation
                          an approx. that speeds-up ground-to-ground propag.
                          It has the value 1 (true) if any of the flags
                          write_2D_TLoss, write_phase_speeds, write_modes
                          or write_dispersion are true.


OUTPUT Files -  Format description (column order):
  tloss_1d.cnm:              r, 4*PI*Re(P), 4*PI*Im(P), (incoherent TL)
```

```
 tloss_1d.lossless.cnm:
 tloss_2d.cnm:              r, z, 4*PI*Re(P), 4*PI*Im(P)
 Nby2D_tloss_1d.cnm:        r, theta, 4*PI*Re(P), 4*PI*Im(P), (incoherent TL)
 Nby2D_tloss_1d.lossless.cnm:

 phasespeeds.cnm:           Mode#, phase speed [m/s], imag(k)
 mode_<mode_count>.cnm      z, (Mode amplitude)
 dispersion_<freq>.cnm      Contains one line with entries: freq (# of modes)
                            followed for each mode 'i' by (Re, Im) parts of:
                            (k(i)) Mode(i)(z_src) Mode(i)(z_rcv)
 atm_profile.cnm            z,u,v,w,t,d,p,c,c_eff

Examples (run from 'samples' directory):

   ../bin/CModess --atmosfile NCPA_canonical_profile_zuvwtdp.dat
                  --atmosfileorder zuvwtdp  --azimuth 90 --freq 0.1

   ../bin/CModess --atmosfile NCPA_canonical_profile_zuvwtdp.dat
                  --atmosfileorder zuvwtdp --azimuth 90 --freq 0.1 --write_2D_TLoss

   ../bin/CModess --atmosfile NCPA_canonical_profile_zuvwtdp.dat
                  --atmosfileorder zuvwtdp --skiplines 0 --freq 0.1
                  --Nby2Dprop --azimuth_start 0 --azimuth_end 360 --azimuth_step 1
```

The options and flags for **CModess** are the same as for **Modess**. The reader is referred to the **Modess** documentation above for explanations. The only difference is that the output filenames now have end with the suffix `.cnm` rather than `.nm`.

6.3 Running CModess: examples

The **CModess** help page ends with three examples. They are the same examples used to illustrate Modess. The examples set the frequency to 0.1 Hz to keep run times short. The first is a simple example illustrating the primary input modes for **CModess**. It is assumed that the user runs it in the `samples` directory. Note that if the **ncpaprop** `bin` directory is in the system's path one may enter `CModess` rather than `../bin/CModess`. The command line entry for the example is

```
../bin/CModess --atmosfile NCPA_canonical_profile_zuvwtdp.dat
               --atmosfileorder zuvwtdp --azimuth 90 --freq 0.1
```

In this example the four required options are set. Otherwise the default settings are used. Ground-to-ground propagation in the NCPA toy atmosphere (see Figure 1) at a frequency of 0.1 Hz is modelled. Eastward propagation, at 90 degrees azimuth (from North) is both modelled. By default the signal is propagated out to 1000 km in range with range steps of 1 km. The atmospheric profiles are specified in the column-based text file `NCPA_canonical_profile_zuvwtdp.dat` that is included in the `samples` directory. In the above example the profile file has 7 columns in the order `zuvwtdp` (refer to section 3.2 for an explanation of atmospheric specifications).

The output is, by default, the one-dimensional (1D) transmission loss on the ground for both lossy and lossless cases and is written into the text files `tloss_1d.cnm` and `tloss_1d.lossless.cnm` as described above in the `CModess` help page; see section 6.2 and section 4.2 for more discussion. Examples of the magnitude of the complex 1D transmission loss output are shown in Figure 10. Eastward propagation, as specified in the example above, and northward propagation, obtained by replacing `--azimuth 90` with `--azimuth 0`, are shown. Note that in general the output from all programs will be saved in the directory from which the code is run.

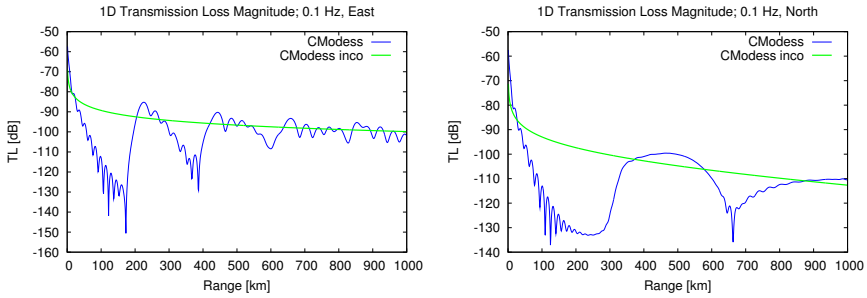

Figure 10: 1D transmission loss magnitude at 0.1 Hz obtained with **CModess** for ground-to-ground propagation in the NCPA toy atmosphere shown in Figure 1 for eastward, left plot, and northward, right plot. Shown are the transmission loss magnitude and incoherent transmission loss.

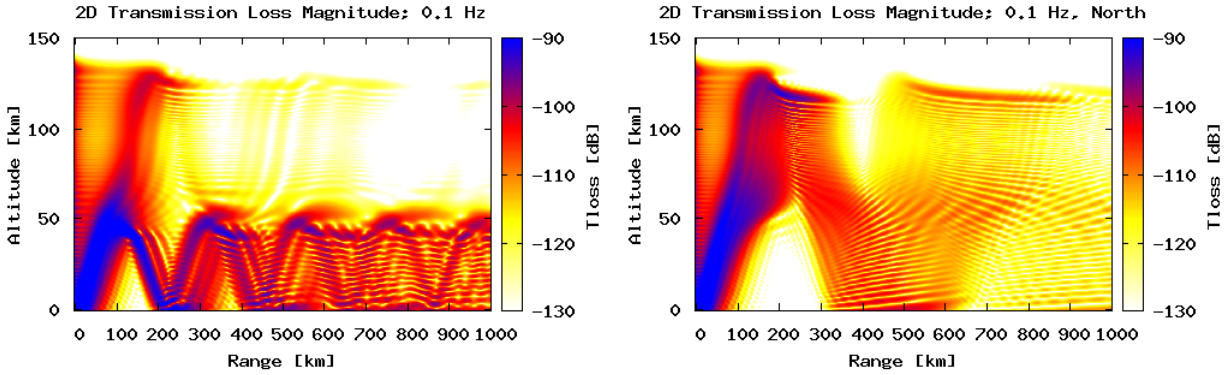

Figure 11: 2D transmission loss magnitude at 0.1 Hz obtained with **CModess** for propagation from a source on the ground in the NCPA toy atmosphere shown in Figure 1 for eastward, upper plot, and northward, lower plot, propagation.

In the second example the `--write_2D_Tloss` flag is appended to the command line entry for the first example:

```
../bin/CModess --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
               --atmosfileorder zuvwtdp --skiplines 0 --azimuth 90 \
               --freq 0.1 --write_2D_TLoss
```

As described above and in section 4.2, the output is written to a default file, in this case `tloss2d.cnm`. The 2D transmission loss magnitude that results from eastward propagation, as specified in the example above, as well as northward propagation, obtained by replacing `--azimuth 90` with `--azimuth 0`, in the NCPA toy model is plotted in figure 11.

In the third example the `--Nby2Dprop` flag is appended to the command line entry for the first example. This flag requires that values for the options `--azimuth_start`, `--azimuth_end`, and `--azimuth_step` be given.

```
../bin/CModess --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
               --atmosfileorder zuvwtdp --skiplines 0 --freq 0.1 \
               --Nby2Dprop --azimuth_start 0 --azimuth_end 360 --azimuth_step 1
```

The output is written to the files `Nby2D_tloss_1d.cnm` and `Nby2D_tloss_1d.lossless.cnm` as described above in section 5.2. The lossless transmission losses are also computed and saved as are the incoherent

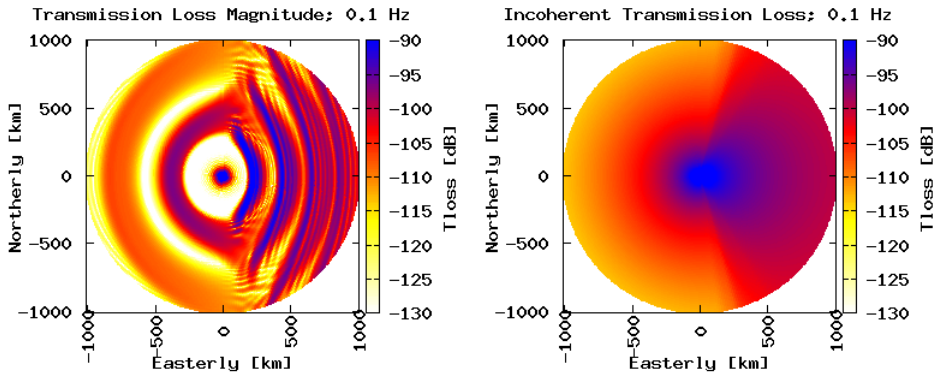

Figure 12: N by 2D transmission loss magnitude and incoherent transmission loss obtained with **WMod** for ground-to-ground propagation in the NCPA toy model atmosphere from a source at the origin to all azimuths.

transmission losses. The resulting transmission loss magnitude is plotted in figure 12.

In all these examples one will find that the runtimes are much longer than for **Modess**. The N by 2D run can take on the order of an hour, even at 0.1 Hz. It will take much longer at higher frequencies.

7 ModBB - Broad-band (pulse) propagation from Normal Modes

ModBB is designed to propagate the infrasonic signal produced by a transient source. **ModBB** produces a broad band signal by Fourier synthesis of frequency-domain solutions obtained with either **Modess** or **WMod**. Modeling with **ModBB** is thus restricted to propagation in a stratified atmosphere. **CModess** has not been included because of the inordinately long runtimes that would be required as well as the severe restriction on frequency required to ensure convergence.

To simulate broad band propagation using a normal mode model it is sufficient to compute the modal dispersion curves $\kappa_j(\omega) = k_j(\omega) + i\alpha(\omega)$ and mode functions $\psi_j(z,\omega)$ over a frequency band large enough to capture the spectrum of the signal source. Here $\omega = 2\pi f$ is the angular frequency and f is the cyclical frequency. **ModBB** computes the dispersion curves and mode functions and saves them in files. Currently, the mode functions are computed and saved at a single altitude. Given a source spectrum or waveform, these files are then used as input to **ModBB** for the synthesis of the resulting propagated waveform. **ModBB** provides a model waveform which can be used to simulate the bandpassed signal from an explosive source. Arbitrary source waveforms or spectra can also be introduced through user-provided files. In addition, the bandpassed impulse response function can be computed.

7.1 Mathematical Background

Let $P(r,z,t)$ be the deviation from mean pressure and use the notation introduced in section 4.1. Then **ModBB** solves the generalized wave equation

$$\left[\nabla_H^2 + \rho_0 \frac{\partial}{\partial z}\left(\frac{1}{\rho_0}\frac{\partial}{\partial z}\right) - \frac{1}{c^2}(\frac{\partial}{\partial t} + \mathbf{v}_{0,H}\cdot\nabla_H)^2\right] P(r,z,t) = 0 \tag{4}$$

subject to the boundary condition

$$\frac{\partial P}{\partial z}\Big|_{z=0} = 0$$

on the ground surface. If $s(t)$ is the pressure deviation extrapolated to $r = 1$ meter and if

$$q(\omega) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} s(t)e^{i\omega t}dt$$

are the Fourier components of s then P can be obtained by Fourier synthesis from a modal sum, p, as given by equation 2. For purposes of modeling the signal propagation by the linear propagation models considered here, $s(t)$ can be considered to be the waveform of the effective acoustic source signal. Using the fact that P and s are both real valued and changing integration variable from angular frequency ω to cyclical frequency f one has

$$P(r,z,t) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} q(\omega)p(r,z,\omega)e^{-i\omega t}d\omega = \mathrm{Re}\sqrt{8\pi}\int_0^{\infty} q(2\pi f)p(r,z,2\pi f)e^{-2\pi i f t}df.$$

The motivation behind the design of **ModBB** is to model the propagation of signals from large explosions. Such signals have abrupt onsets in the near field but decrease rapidly with time and can be said to be of finite duration. The nearly discontinuous signal onsets, however, lead to Fourier transforms which decrease slowly with increasing frequency and which generally cannot be well approximated by signals with finite bandwidth. Despite this the effective acoustic source signal can be modelled by a waveform with finite bandwidth. This is because of the atmospheric attenuation which increases rapidly with increasing frequency so that the high frequency components of the source signal do not propagate to the far field.

Let the bandwidth of the effective source signal be less that F. Then the Fourier integral can be estimated by replacing the continuous integral by a finite discrete sum. Let the onset of the propagated signal occur after a time T_0 and let the duration of the signal be less than T chosen so that the entire signal is contained in the interval between time T_0 and $T_0 + T$. If the sum is produced by sampling evenly in

frequency with sample spacing $\Delta f = \frac{1}{T}$ then the integral above can be efficiently estimated using the Fast Fourier Transform algorithm (FFT) (see, for example, [9]). The FFT algorithm produces $P(r, z, t)$ sampled discretely in time with sample spacing $\Delta t = \frac{1}{F}$. The number of points summed over in the FFT is $N = TF$. Recall that, given the setup here, the FFT algorithm applied to a function of t produces a periodic function f with periodicity F and the inverse transform produces a periodic function of t with period T. Further, for the Fourier transform so obtained the positive frequency components are contained in the first $N/2$ points and the negative frequency components in the rest. Thus, for $m = 0, 1, 2, \ldots, N$,

$$P(r, z, T_0 + m\Delta t) \approx \sqrt{8\pi}\Delta f \sum_{n=1}^{N/2-1} q(2\pi n\Delta f)p(r, z, 2\pi n\Delta f)e^{-2\pi i n\Delta f T_0}e^{-2\pi i \frac{nm}{N}}.$$

Substituting from equation 2 one obtains the formula that **ModBB** computes:

$$P(r, z, T_0 + m\Delta t) \approx \frac{ie^{-i\pi/4}}{\sqrt{r}}\sqrt{\frac{\rho_0(z)}{\rho_0(z_s)}}\Delta f \sum_{n=1}^{N/2-1} q(2\pi n\Delta f) \Big(\sum_j \psi_j(z_s, 2\pi n\Delta f)\psi_j(z, 2\pi n\Delta f)\frac{e^{i\kappa_j(2\pi n\Delta f)r}}{\sqrt{k_j(2\pi n\Delta f)}}\Big) e^{-2\pi i n\Delta f T_0}e^{-2\pi i \frac{nm}{N}}. \quad (5)$$

Since there are no modes at zero frequency the $n = 0$ term is absent from the above sum.

Note that once the modal wave numbers (modal dispersion curves) and mode functions are computed over the required range of frequencies equation 5 can be evaluated. Recall from section4.1 that in equation 5 the range of wave numbers to be included in the modal sum must be determined prior to their being computed. The upper limit for the wave numbers is determined by the program itself. The lower limit is set by the user with the choice of maximum altitude of the computational domain.

It is critical that the signal onset time T_0 and duration T be chosen correctly. If parts of the signal are not contained in the interval between T_0 and T then, due to the periodicity of discrete Fourier transforms, these parts will be aliased into the interval, overlapping with other parts of the waveform. Further, the truncation of the modal sum at a maximal phase velocity introduces a wave number cutoff which produces spurious arrivals. T must be large enough to temporally separate these unphysical components of the output from the true waveform synthesis.

7.2 Running ModBB

To model the propagation of a pulse two largely independent steps are required. First the modal wave number dispersion curves and mode functions must be computed. Then the computed values can be used to perform the modal sums and Fourier synthesis of equation 5 for specified values of r. **ModBB** is designed to perform these steps separately. First the dispersion curves and mode functions are computed using either **Modess** or **WMod** and saved to a file, called a dispersion file. Although the full mode functions are computed by both **Modess** and **WMod**, the current version of **ModBB** only saves the mode functions at a given, fixed height. Once a dispersion file has been written and saved it can be used as input to a pulse propagation routine which evaluates equation 5. Producing a dispersion file is computationally expensive and can take some time. Once a dispersion file is written and saved computing a propagated waveform is essentially instantaneous.

Signal duration and bandwidth are set during the computation of the dispersion curves. **ModBB** requires the user to input the frequency step Δf and bandwidth F through options `--f_step <df>` and `--fmax <F>` respectively. Here `<df>` is the numerical value for Δf and `<F>` the value for F. Once Δf is set signal duration is also set through the relation $T = \frac{1}{\Delta f}$. Δf and F are the parameters with the greatest impact on runtimes. The number of modes increases quadratically with frequency so that as F increases runtimes increase dramatically. Since $\frac{F}{\Delta f}$ is the number of individual frequencies required, the smaller Δf is the more computations are required. On the other hand, it is not possible to simply reduce T and F. T must

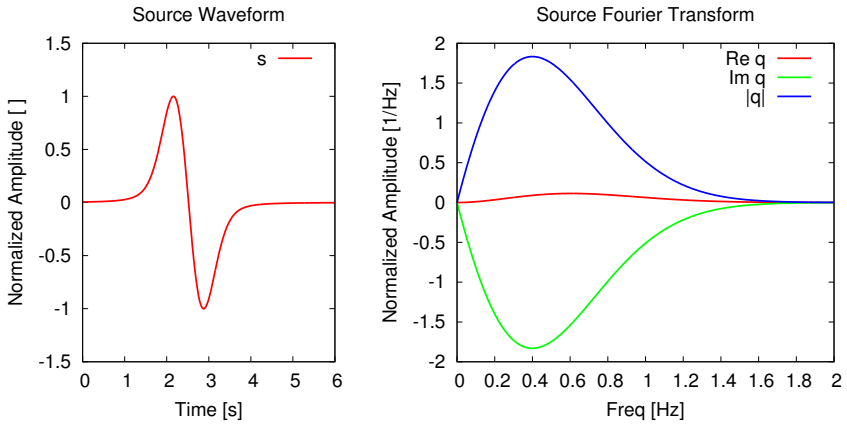

Figure 13: The built-in initial pulse with maximum Fourier transform frequency of 0.4 Hz and maximum frequency of 2 Hz. The waveform is shown in the left panel and the Fourier transform; real part, imaginary part, and magnitude; in the right panel.

be larger than the duration of the propagated signal, typically T can be on the order of 1000 seconds or for which Δf is of the order of 0.001 or less. F must be large enough so that the effective source signal retains the characteristics of a signal from an explosive source: sudden positive pressure onset, one zero crossing, and a long negative pressure tail. We recommend that F be chosen five times larger than the maximum frequency of the source signal under consideration.

For propagating a pulse a dispersion file is required as input. **ModBB** provides four ways to input the effective acoustic source signal $s(t)$, either directly or through its Fourier transform $q(\omega)$. **ModBB** can produce the band pass filtered impulse response function by setting $q = 1$. This is equivalent to choosing $s(t)$ to be a delta function at $t = 0$. Filtering, equivalent to a boxcar filter on the interval $-F < f < F$, is accomplished by the frequency bound in the dispersion file. **ModBB** also has a built-in effective acoustic source function. The built-in source function has one adjustable parameter: the frequency of its maximum Fourier transform. Its waveform is normalized to have maximum amplitude of 1. The built-in source function with the maximum of its Fourier transform at 0.4 Hz is plotted in figure 13. Both source waveform $s(t)$ and Fourier transform $q(\omega)$ are written to files `source_waveform_input_example.dat` and `source_spectrum_input_example.dat` each time the pulse propagation routines are run. These files have the format

```
time[s] pressure[Pa]
```

and

```
freq[Hz] Re(Fourier transform)[Pa/Hz] Im(Fourier transform)[Pa/Hz]
```

respectively. Finally, the input of user-provided source files is supported. The user can provide either a waveform file or a Fourier transform file, in the format given above for the built in source files.

To run **ModBB** make sure its executable is in the system's path and enter

```
ModBB [--option1 val1] [--option2 val2] [...] [--flag1] [...]
```

on a command line. Generally, options are followed by values, either numbers, strings or filenames. Flags are not. Entering `ModBB` without any options or flags sends the following help page to the screen:

```
The options below can be specified in a colon-separated file "ModBB.options"
or at the command line.  Command-line options override file options.
 --help -h                Print this message and exit

One of two algorithms can be used to perform pulse propagation.
The first is based on the Effective Sound Speed Approximation (as in ModESS);
the second is based on the the Wide_Angle High-Mach solution of the
wave equation (see  implementation in WMod).
ModESS is faster but it is accurate for (launch) angles less than 30 deg
and low wind speeds. WMod extends the validity to higher angles and
high Mach numbers but it runs slower.
Options --use_modess and --use_wmod allow the user to choose
the desired algorithm when computing the dispersion data (see step 1 below).

To propagate a pulse, 2 steps must be completed:
 1. A dispersion file must be available or computed
     use the option --out_disp_src2rcv_file

 2. Perform pulse propagation for one of 2 scenarios:
    a. source-to-receiver at one range (option --pulse_prop_src2rcv)
    b. source-to-receiver at several equally spaced ranges
       (option --pulse_prop_src2rcv_grid)

    Several source types can be one of the following:
        delta function              -> see option --get_impulse_resp
        built-in pulse              -> see option --use_builtin_pulse
        user-provided spectrum file -> see option --src_spectrum_file
        user-provided waveform file -> see option --src_waveform_file

To compute a dispersion file the following option is REQUIRED:
 --out_disp_src2rcv_file  <dispersion filename>
                   Output dispersion curves and modal values for
                   source-to-receiver propagation to the specified file

 Example (run in the 'samples' directory):

      Compute dispersion file that will be used to compute the pressure pulse
      at 1 receiver. Assume that we want to end up with a pulse having a spectrum
      with a maximum frequency of f_max=0.5 Hz. Also assume that we want
      the pulse represented on a time record of T=512 seconds. The number of
      positive frequencies necessary for the calculation is T*f_max = 256 i.e.256
      frequencies between 0 and 0.5 Hz. Thus we know f_max=0.5 Hz and
      f_step=f_max/256=0.001953125 Hz. The corresponding run command is:

    ../bin/ModBB --out_disp_src2rcv_file myDispersionFile.dat
                 --atmosfile NCPA_canonical_profile_zuvwtdp.dat
                 --atmosfileorder zuvwtdp --skiplines 0 --azimuth 90
                 --f_step 0.001953125 --f_max 0.5 --use_modess

    Each line in this dispersion file has the format:
           freq n_modes rho_src rho_rcv Re(k_pert) Im(k_pert) V_m(z_src) V_m(z_rcv)
    where m varies from 1 to n_modes.

In addition the following options are REQUIRED:
```

```
 --use_modess              Prompts the use of ModESS algorithm.
 --use_wmod                Prompts the use of WMod algorithm.
 Note that --use_modess and --use_wmod are mutually exclusive.
 --atmosfile  <filename>   Uses an ASCII atmosphere file
                           referenced to Mean Sea Level (MSL).
 --atmosfileorder          The order of the (z,t,u,v,w,p,d) fields in
                           the ASCII file (Ex: 'ztuvpd')
 --skiplines               Lines at the beginning of the ASCII file to skip
 --azimuth                 Value in range [0,360), clockwise from North
 --f_step                  The frequency step
 --f_max                   Maximum frequency to propagate
    Note that in this case the array of frequencies is [f_step:f_step:f_max].

OPTIONAL [defaults]:
 --f_min                   Minimum frequency [f_step Hz]
 --maxheight_km            Calculation grid height in km above MSL [150 km]
 --zground_km              Height of the ground level above MSL [0 km]
 --Nz_grid                 Number of points on the z-grid from ground to maxheight
                           [20000]
 --sourceheight_km         Source height in km Above Ground Level (AGL) [0]
 --receiverheight_km       Receiver height in km AGL [0]
 --maxrange_km             Maximum horizontal distance from origin to propagate
                           [1000 km]
 --ground_impedance_model  Name of the ground impedance models to be employed:
                           [rigid], TBD
 --Lamb_wave_BC            For a rigid ground: if ==1 it sets
                           admittance= = -1/2*dln(rho)/dz; [ 0 ]
 --wind_units              Use it to specify 'kmpersec' if the winds are given
                           in km/s [mpersec]

Options for PULSE PROPAGATION:
 --pulse_prop_src2rcv <dispersion filename>
                    Propagate pulse from source to 1 receiver
                    at a distance specified by option --range_R_km;
 --range_R_km      Propagate pulse to this range [km]
 --waveform_out_file <waveform filename>   Name of the waveform output file

 --pulse_prop_src2rcv_grid <dispersion filename>
                    Propagate pulse from source to array of
                    horizontally equally-spaced receivers

 REQUIRED additional options:
 --R_start_km       Propagation from this range to R_end_km in DR_km steps.
 --R_end_km         Pulse is propagated from R_start_km to this range.
 --DR_km            Range step to propagate from R_start_km to R_end_km.
 --waveform_out_file <waveform filename>
                    Name of the waveform output file.

 OPTIONAL [defaults]:
 --f_center         The center frequency of the pulse; must be <= [f_max/5].
 --max_celerity     Maximum celerity [300 m/s].
 --nfft             Number of points used in the FFT computation [2f_max/f_step].
```

```
SOURCE TYPE options: Use one of the following 4 options to specify the source:
 --get_impulse_resp       Flag to use a delta function as source and
                          to output the impulse response.
 --use_builtin_pulse      Flag to request the use of the built-in source pulse.
 --src_spectrum_file      Specify the file name of the source spectrum
                          at positive frequencies. The file must have 3 columns
                             | Freq | Real(Spectrum) | Imag(Spectrum) |
 --src_waveform_file      Specify the file name of the user-provided
                          source waveform. The file must have 2 columns
                             |Time | Amplitude |
   If none of then source type options are specified the delta function source
   is the default i.e. the output is the impulse response.

 Example: Pulse propagation to a point on the ground at range_R_km
          and output the impulse response:

    ../bin/ModBB --pulse_prop_src2rcv myDispersionFile.dat --range_R_km 240
                 --waveform_out_file mywavf.dat --get_impulse_resp

 Example: Pulse propagation to a point on the ground at range_R_km
          and employ the user-provided source spectrum:

   ../bin/ModBB --pulse_prop_src2rcv myDispersionFile.dat --range_R_km 240
                --waveform_out_file mywavf.dat --max_celerity 300
                --src_spectrum_file source_spectrum_example.dat

 Example: Pulse propagation to several points on the ground 20 km apart
          and employ the user-provided source waveform:

    ../bin/ModBB --pulse_prop_src2rcv_grid myDispersionFile.dat
                 --R_start_km 240 --DR_km 20 --R_end_km 300
                 --waveform_out_file mywavf.dat
                 --src_waveform_file source_waveform_input_example.dat
```

As discussed above, **ModBB** has two distinct functionalities: dispersion curves and mode amplitudes can be computed and written to a dispersion file and existing dispersion files can be used as input to a Fourier synthesis of a propagated pulse. To compute and save dispersion files one uses the option `--out_disp_src2rcv_file` followed by the name of the file. Dispersion files are computed using either **Modess** or **WMod**, the choice being made with the options `--use_modess` or `--use_wmod`. The dispersion file format is a sequence of lines, one per frequency, beginning with the frequency and then containing the number of modes, the mean density of the atmosphere at source and receiver and then a sequence of horizontal wave numbers and mode amplitudes at source and receiver as in

```
freq n_modes rho_src rho_rcv Re(k_m_pert) Im(k_m_pert) V_m(z_src) V_m(z_rcv)
```

for `m` ranging from 1 to `n_modes`. Additional options specific to `--out_disp_src2rcv_file` are `f_step`, `f_max`, and `f_min`. The options `f_step` and `f_max` have been described above. The option `f_min` sets the frequency at which the dispersion file begins; it defaults to `f_step`. Generally, no significant reduction in run time is achieved by setting `f_min` > `f_step` since there are very few modes when the frequency is small. Setting `f_min` can be useful if one needs to compute dispersion files in several steps, to be concatenated into a single file afterwards.

To perform a Fourier synthesis of a propagated pulse one sets either `--pulse_prop_src2rcv`, to propagate to a single receiver location, or `--pulse_prop_src2rcv_grid`, to propagate to an array of receiver

locations, followed, in both cases, by the name of the appropriate dispersion file. For both options a source type must be set as described above. If the built-in pulse is to be used, by setting `--use_builtin_pulse`, then the frequency of maximum Fourier component must be set using `--f_center`; the default value is $F/5$. The user provides an output file using `--waveform_out_file` followed by the output filename. When using `--pulse_prop_src2rcv` the range at which the waveform is to be computed must be set using `--range_R_km`. If `--pulse_prop_src2rcv_grid` is being used then the smallest range in the receiver array, `--R_start_km`, the largest range, `--R_end_km`, and the spacing between ranges, `DR_km`, must all be set. Distances are in kilometers. Note that the altitudes for source and receiver are set in the dispersion file. In all cases **ModBB** computes the propagated waveform in a moving window. The window length is $T/\Delta f$ where Δf is the value for `--f_step` from the dispersion file. The start of the window is set using the option `--max_celerity` followed by a value c. If R is the range at which the waveform is being computed then the window starts at $T_0 = R/c$. Note that with `--pulse_prop_src2rcv` the output file format is time, waveform, Hilbert transformed waveform, with the time record beginning at T_0. With `--pulse_prop_src2rcv_grid` data is stored in blocks of constant range with the format range, time, waveform and with each time record beginning at 0.

Finally, the option `--nfft` sets the size of the FFT to be used in the waveform synthesis. Generally, using the number of frequencies in the dispersion file results in a poorly sampled waveform. Increasing the number of points used by zero padding improves the quality of the resulting waveform plots. It is to be emphasized that no new information is introduced in this way. The synthesized waveform is simply being more finely sampled.

7.3 Running ModBB: examples

As an example consider the following (note that this example is different from the ones that are on the **ModBB** help page). Here a dispersion file is written for eastward propagation in the NCPA toy atmosphere. The bandwidth is chosen to be small, $F = 0.5$ Hz, so that the file computes fairly rapidly. Δf is chosen to be 0.002 corresponding to a time window, T, of 500 seconds. Calculations are done using **Modess**.

```
../bin/ModBB --out_disp_src2rcv_file myDispersionFile.dat
             --atmosfile NCPA_canonical_profile_zuvwtdp.dat
             --atmosfileorder zuvwtdp --azimuth 90
             --f_min 0.002 --f_step 0.002 --f_max 0.5
             --use_modess
```

The dispersion file created above, `myDispersionFile.dat`, is then used to propagate the built-in pulse to 270 km. `--f_center` is not set so that the default $F/5$, in this case 0.1 Hz, is used. Note that `--max_celerity` is set at 320 m/s rather than the default value of 300 m/s for which the resulting time window is not optimal. The waveform and Fourier transform for the built-in pulse is plotted in figure 14. The propagated waveform is plotted in figure 15.

```
../bin/ModBB --pulse_prop_src2rcv myDispersionFile.dat --range_R_km 270
             --waveform_out_file mywavf.dat --use_builtin_pulse --max_celerity 320
```

The example below creates a grid of waveforms using `--pulse_prop_src2rcv_grid`, propagating using the dispersion file created using the example above. Waveforms are written in 20 km increments starting at 220 km and ending at 300 km. Maximum celerity of 320 m/s was used rather than the default value. The results, shifted so that each time window begins at the appropriate T_0, are plotted in figure 16.

```
../bin/ModBB --pulse_prop_src2rcv_grid myDispersionFile.dat
             --R_start_km 220 --DR_km 20 --R_end_km 300
             --waveform_out_file mywavf_grid.dat
             --use_builtin_pulse --max_celerity 320
```

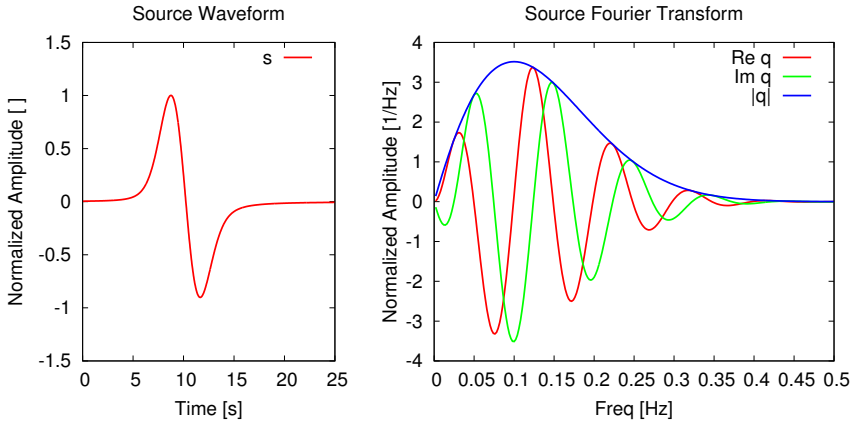

Figure 14: The built-in source waveform and Fourier transform with peak at 0.1 Hz.

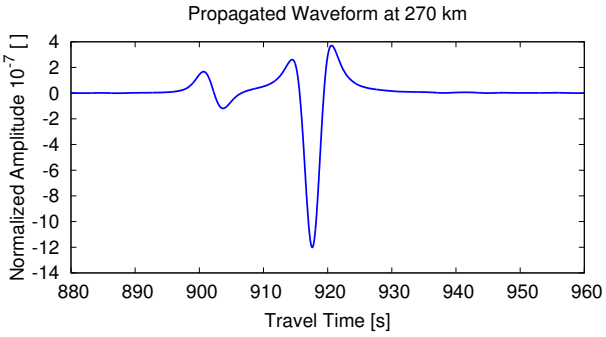

Figure 15: Eastward propagation in the NCPA toy atmosphere. Propagated pulse to 270 km vs time.

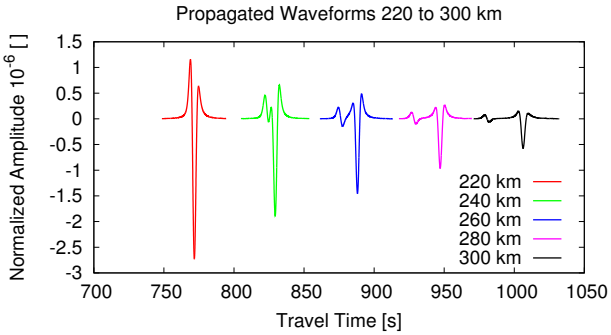

Figure 16: Eastward propagation in the NCPA toy atmosphere. Propagated pulse from 220 to 300 km vs time in increments of 20 km.

8 ModessRD1WCM - Modal Effective Sound Speed Range-Dependent One-Way Coupled Modes

ModessRD1WCM implements a normal mode expansion for propagation of a single tone in a range dependent atmosphere in the effective sound speed approximation and can be considered to be a generalization of the **Modess** program. Propagation is one-way in that reflections induced by range dependence is ignored. As output the code provides single frequency 1D and 2D transmission losses (TL) in the planar approximation. As for **Modess** the atmospheric attenuation is taken into account as a perturbation.

8.1 Mathematical Background

In principle **ModessRD1WCM** solves the same mathematical problem that **Modess** does (see section 4.1). The difference is that a range dependent atmosphere is introduced by dividing the range axis into a number of segments and approximating the field as range independent within each segment. In each segment only the forward propagating field component is included. Equivalently, at the interfaces between segments only the forward-scattered components are considered. The back-scattered parts are neglected, leading to a marching type algorithm (see eg. section 5.9 of reference [5]).

Let there be $N+1$ segments. Let the modes and modal wave numbers in the m^{th} segment be given by $\psi_j^{(m)}$ and $k_j^{(m)} + i\alpha_j^{(m)}$ respectively for $m = 0, 1, 2, 3, \ldots, N$. Let the interfaces between the segments be at ranges r_m for $m = 1, 2, 3, \ldots, N$. In the first segment the algorithm is started by running **Modess** so that for $0 < r < r_1$ the pressure deviation is given by the modal sum equation 2 (where the modes and wave numbers have the superscript (0)). In each successive segment one has

$$p(r, z, \omega) \approx \sqrt{\frac{\rho_0^{(m)}(z)}{r}} \sum_j c_j^{(m)} \psi_j^{(m)}(z) e^{i(k_j^{(m)} + i\alpha_j^{(m)})r}.$$

Using the energy conserving form described in reference [5], at each interface the coefficients $c_j^{(m)}$ are determined by

$$c_j^{(m)} = \frac{1}{2} \sum_{j'} \Big(1 + \frac{k_{j'}^{(m-1)}}{k_j^{(m)}}\Big) c_{j'}^{(m-1)} \Big(\int_0^\infty \psi_j^{(m)}(z) \psi_{j'}^{(m-1)}(z)\, dz \Big) e^{i(k_{j'}^{(m-1)} + i\alpha_{j'}^{(m-1)})r_m}.$$

8.2 Running ModessRD1WCM

ModessRD1WCM runs much as **Modess** does with one significant difference: range dependent atmospheric profiles must be input. This can be done in two ways. A G2S `.env` binary file can be read in using the `--g2senvfile` option followed by the G2S filename (for a discussion of the G2S atmospheric state specifications see, for example, the discussion in §4 of [7]). Alternatively, a sequence of ascii files, one for each stratified segment, can be read in from a directory using the `--use_1D_profiles_from_dir` followed by the name of the directory. Profiles from the given directory must be in the format `profile####.dat` where `####` is an integer ranging from 0000 to 9999. The profiles are read in numerical order. The ranges at which a new profile is read, either from the G2S file or from the profile directory, is set explicitly either using `--use_profile_ranges_km`, followed by a sequence of underscore-separated ranges (in kilometers, see the example below), or in an evenly spaced array using `--use_profiles_at_steps_km` followed by the (fixed) range step in kilometers.

To run **ModessRD1WCM** make sure its executable is in the system's path and enter

```
ModessRD1WCM [--option1 val1] [--option2 val2] [...] [--flag1] [...]
```

on a command line. Generally, options are followed by values, either numbers, strings or filenames. Flags are not. Entering `ModessRD1WCM` without any options or flags sends the following help page to the screen:

```
By default 'ModessRD1WCM' computes the 1D transmission loss (TL)
at the ground or the specified receiver height and saves the data to 2 files:
   file tloss_rd_1d.nm - considering attenuation in the atmosphere
   file tloss_rd_1d.lossless.nm  - no attenuation
Additionally, if the flag --write_2D_TLoss is present on the command line
the 2D TL is saved to file tloss_rd_2d.nm.

The options below can be specified in a colon-separated file
"ModESSRD1WCM.options" or at the command line.
Command-line options override file options.
 --help -h                Print this message and exit

 The atmosphere can be specified from one of 3 different sources:
    1. An .env file containing the atmospheric specifications at certain ranges:
       use option --g2senvfile <filename>
    2. Several ASCII files stored in a given directory:
       use option --use_1D_profiles_from_dir <mydirname>
    3. A single ASCII file. This will just force a range-independent run.
       use option --atmosfile <filename>

The options available are:

REQUIRED (no default values):
 --atmosfileorder         The order of the (z,t,u,v,w,p,d) fields in the file
                          (Ex: 'ztuvpd')
 --skiplines              Lines at the beginning of the ASCII file to skip
 --azimuth                Degrees in range [0,360), clockwise from North
 --freq                   Frequency [Hz]
 --g2senvfile <filename>  Uses an .env binary file (for range-dependent code)
 --use_1D_profiles_from_dir
                          e.g. --use_1D_profiles_from_dir <myprofiles>
                          This option allows to use the ascii profiles stored in
                          the specified directory. The profiles must have names
                          'profiles0001', 'profiles0002', etc. and will be
                          used in alphabetical order at the provided ranges
                          e.g. in conjunction with either
                          option  '--use_profile_ranges_km'
                          or option '--use_profiles_at_steps_km'
                          If there are more requested ranges than existing
                          profiles then the last profile is used repeatedly
                          as necessary.
    Example: >> ../bin/ModESSRD1WCM --atmosfileorder zuvwtdp --skiplines 1
                --azimuth 90 --freq 0.1 --use_1D_profiles_from_dir myprofiles
                --use_profile_ranges_km 100_300_500_600_700

 --atmosfile  <filename>  Uses an ASCII atmosphere file.
                          In this case the run will just be range-independent

OPTIONAL [defaults]:
 --maxheight_km           Calculation grid height in km above MSL [150 km]
 --zground_km             Height of the ground level above MSL [0 km]
 --Nz_grid                Number of points on the z-grid from ground to maxheight
                          [20000]
 --sourceheight_km        Source height in km Above Ground Level (AGL) [0]
```

```
 --receiverheight_km      Receiver height in km AGL [0]
 --maxrange_km            Maximum horizontal propagation distance from origin
                          [1000 km]
 --Nrng_steps             Number of range steps to propagate [1000]
 --ground_impedance_model Name of the ground impedance models to be employed:
                          [rigid], others TBD
 --Lamb_wave_BC           If ==1 it sets admittance = -1/2*dln(rho)/dz; [ 0 ]
 --wind_units             Use it to specify 'kmpersec' if the winds are given in km/s
                          [mpersec]
 --use_attn_file          Use it to specify a file name containing user-provided
                          attenuation coefficients to be loaded instead of
                          the default Sutherland-Bass attenuation.
                          The text file should contain two columns:
                              height (km AGL) and
                              attenuation coefficients in np/m.

 --use_profile_ranges_km
                          e.g. --use_profile_ranges_km  20_50_80.5_300
                          The profiles at certain ranges specified by numbers
                          (in km) in a string such as 20_50_80.5_300 are
                          requested in the propagation. Note that underscores
                          are necessary to separate the numbers.
                          Note also that these are requested ranges;
                          however the left-closest profile available
                          in the .env file will actually be used;
                          for instance we request the profile at 300 km
                          but in the .env file the left-closest profile
                          may be available at 290 km and it is the one used.

 --use_profiles_at_steps_km
                          e.g. --use_profiles_at_steps_km 100
                          The profiles are requested at equidistant intervals
                          specified by this option [1000]


FLAGS (no value required):
 --write_2D_TLoss         Outputs the 2D transmission loss to
                          default file: tloss_rd_2D.nm
 --turnoff_WKB            Turn off the WKB least phase speed estimation
                          an approx. that speeds-up ground-to-ground propag.
                          It has the value 1 (true) if any of the flags
                          write_2D_TLoss, write_phase_speeds, write_modes
                          or write_dispersion are true.


OUTPUT Files -  Format description (column order):
  tloss_rd_1d.nm:           r, 4*PI*Re(P), 4*PI*Im(P), (incoherent TL)
  tloss_rd_1d.lossless.nm:
  tloss_rd_2d.nm:           r, z, 4*PI*Re(P), 4*PI*Im(P)

  Examples to run (from 'samples' directory):

      ../bin/ModessRD1WCM --use_1D_profiles_from_dir profiles
              --atmosfileorder zuvwtdp --skiplines 1 --azimuth 90 --freq 0.1
```

```
        --use_profile_ranges_km 100_200

../bin/ModessRD1WCM --use_1D_profiles_from_dir profiles
       --atmosfileorder zuvwtdp --skiplines 1 --azimuth 90 --freq 0.1
       --use_profiles_at_steps_km 200

../bin/ModessRD1WCM --g2senvfile g2sgcp2011012606L.jordan.env
       --atmosfileorder zuvwtdp --skiplines 1 --azimuth 90 --freq 0.1
       --use_profile_ranges_km 50_100_150_200_250_300

../bin/ModessRD1WCM  --atmosfile NCPA_canonical_profile_zuvwtdp.dat
       --atmosfileorder zuvwtdp --skiplines 1 --azimuth 90 --freq 0.1

 This last example is in fact not a range-dependent case since it just loads
 a single 1D atmospheric profile.
```

8.3 Running ModessRD1WCM: examples

The following example illustrates the functionality and inputs of ModessRD1WCM using user-provided range dependent atmospheric profiles provided by a directory of one-dimensional ascii files.

```
../bin/ModessRD1WCM --use_1D_profiles_from_dir profiles \
       --atmosfileorder zuvwtdp --skiplines 1 --azimuth 90 --freq 0.1 \
       --use_profile_ranges_km 50_100_150_200_250 --write_2D_TLoss
```

The resulting data files are plotted in figure 17.

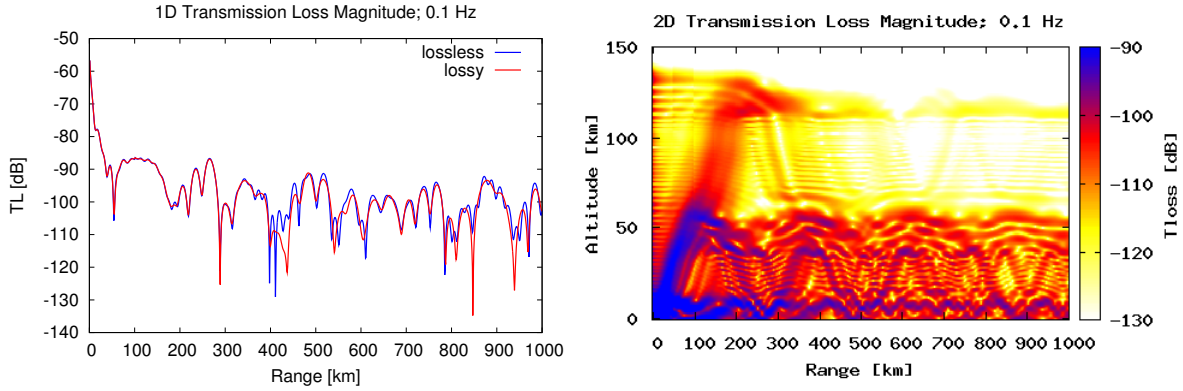

Figure 17: 1D transmission loss magnitude at 0.1 Hz obtained with **ModessRD1WCM** for eastward ground-to-ground propagation in the range dependent profiles provided in the directory `profiles`. Shown are the 1D lossless transmission loss magnitude and lossy transmission loss magnitude and the 2D lossy transmission loss magnitude.

The following example illustrates the functionality and inputs of ModessRD1WCM using a user-provided G2S `.env` file.

```
../bin/ModessRD1WCM --g2senvfile g2sgcp2011012606L.jordan.env \
       --atmosfileorder zuvwtdp --skiplines 1 --azimuth 90 --freq 0.1 \
       --use_profile_ranges_km 50_100_150_200_250 --write_2D_TLoss
```

The resulting data files are plotted in figure 18.

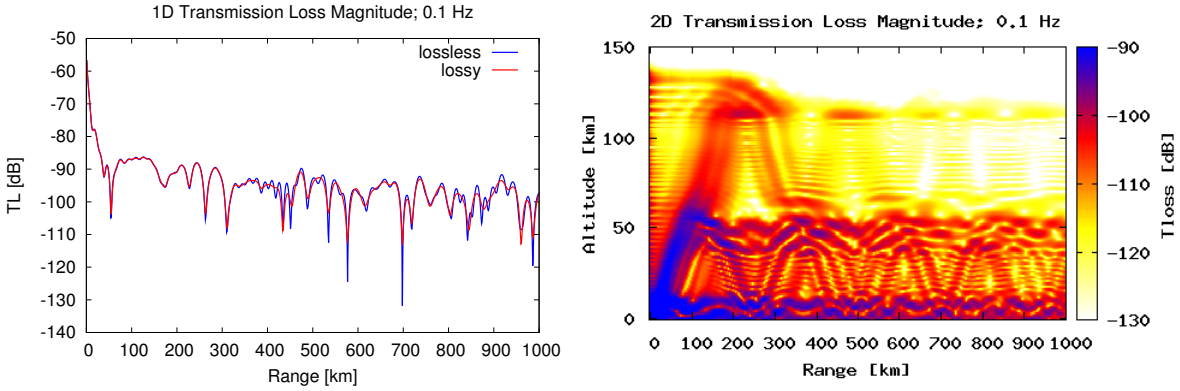

Figure 18: 1D transmission loss magnitude at 0.1 Hz obtained with **ModessRD1WCM** for eastward ground-to-ground propagation in the range dependent profiles provided by a G2S `.env` binary file. Shown are the 1D lossless transmission loss magnitude and lossy transmission loss magnitude and the 2D lossy transmission loss magnitude.

9 pape - Wide-Angle Pade Parabolic Equation Code

9.1 Mathematical Background

pape uses the Parabolic Equation method (see chapter 6 of reference [5]) to solve the single-frequency infrasound propagation in a range-dependent atmosphere over rigid ground in the limit of the effective sound speed approximation. Assuming that the pressure field has weak azimuthal dependence the coresponding Helmholtz equation is:

$$\left[\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial}{\partial r}\right)+\rho_0\frac{\partial}{\partial z}\left(\frac{1}{\rho_0}\frac{\partial}{\partial z}\right)+\left(\frac{\omega}{c_{eff}}+i\alpha\right)^2\right]p(r,z)=0$$

where $p(r,z)$ is the pressure deviation at range r and height z, ω is the angular frequency, ρ_0 is the air density and c_{eff} is the effective (complex) sound speed i.e. the sum of the sound speed and the wind velocity component in the (horizontal) direction of wave propagation. The atmospheric absorption and damping of the acoustic energy is accounted for as the imaginary component of the wavenumber $\frac{\omega}{c}$. The pressure deviation satisfies the boundary condition

$$\left.\frac{\partial p}{\partial z}\right|_{z=0}=0$$

and the asymptotic condition

$$\lim_{r,(z-z_S)\downarrow 0}\left(p(r,z,\omega)-\frac{1}{\sqrt{r^2+(z-z_S)^2}}\right)=0.$$

It is assumed that the solution takes the form

$$p(r,z)=\psi(r,z)H_0^{(1)}(k_0r)$$

where $\psi(r,z)$ is a slowly varying with r envelope and $k_0=\omega/c_0$ is a reference wavenumber. Denoting the index of refraction $n(r,z)=c0/c(r,z))$ the equation to solve becomes:

$$\frac{\partial^2\psi}{\partial r^2}+2ik_0\frac{\partial\psi}{\partial r}+\rho_0\frac{\partial}{\partial z}\left(\frac{1}{\rho_0}\frac{\partial\psi}{\partial z}\right)+k_0^2(n^2-1)\psi=0.$$

This equation can be factored into two one-way equations, for forward propagating and back-propagating energy. Denoting the operators:

$$Q=\rho_0\frac{\partial}{\partial z}\left(\frac{1}{\rho_0}\frac{\partial}{\partial z}\right)+k_0^2n^2\quad\text{ and }\quad q=\frac{1}{k_0^2}(Q-k_0^2),$$

the one way forward-propagating wave equation is:

$$\frac{\partial\psi}{\partial z}=ik_0\left(-1+\sqrt{1+q}\right)\psi. \tag{6}$$

The square root operator in this last equation is implemented numerically with a Pade approximation of order M:

$$\sqrt{1+q}\approx 1+\sum_{m=1}^{M}\frac{a_mq}{1+b_mq},$$

where the coefficients are:

$$a_m=\frac{2}{2M+1}\sin^2\frac{m\pi}{2M+1},\quad b_m=\cos^2\frac{m\pi}{2M+1}.$$

Equation 6 is then solved following the method described in section 6.6.2 of reference [5].

9.2 Running pape

Making sure that the executable for **pape** is in the system's path, it can be run by entering

```
pape [--option1 val1] [--option2 val2] [...] [--flag1] [...]
```

on a command line. Generally, options are followed by values, either numbers, strings or filenames. Flags are not. Entering pape without any options or flags sends the following help page to the screen:

```
By default 'pape' computes the 1D transmission loss (TL)
at the ground or the specified receiver height and saves the data to:
   file tloss_1d.pe - considering attenuation in the atmosphere
Additionally, if the flag --write_2D_TLoss is given the 2D TL is saved
to file tloss_2d.pe.

The options below can be specified in a colon-separated file "PAPE.options"
or at the command line.  Command-line options override file options.
Be sure to precede the options with two minuses: --

 --help -h                Print this message and exit

 The atmosphere can be specified from one of 4 different sources:
    1. An .env file containing the atmospheric specifications at certain ranges:
       use option --g2senvfile <filename>
    2. Several ASCII files stored in a given directory:
       use option --use_1D_profiles_from_dir <mydirname>
    3. A single ASCII file. This will just force a range-independent run.
       use option --atmosfile1d <filename>
    4. A built-in NCPA canonical profile.
       use option (flag) --ncpatoy

The available options are:

REQUIRED (no default values):
 --atmosfileorder         The order of the (z,u,v,w,t,d,p) fields in the file
                          (Ex: 'ztuvpd')
 --skiplines              Lines at the beginning of the ASCII file to skip
 --azimuth                Degrees in range [0,360), clockwise from North
 --freq                   Frequency [Hz]
 --g2senvfile <filename>  Uses an .env binary file (for range-dependent code)
 --use_1D_profiles_from_dir
                          e.g. --use_1D_profiles_from_dir <myprofiles>
                          This option allows to use the ascii profiles stored in
                          the specified directory. The profiles must have names
                          'profiles0001', 'profiles0002', etc. and will be
                          used in alphabetical order at the provided ranges
                          e.g. in conjunction with either
                          option  '--use_profile_ranges_km'
                          or option '--use_profiles_at_steps_km'
                          If there are more requested ranges than existing
                          profiles then the last profile is used repeatedly
                          as necessary.
    Example: >> ../bin/pape --atmosfileorder zuvwtdp --skiplines 1
                --azimuth 90 --freq 0.1 --use_1D_profiles_from_dir myprofiles_dir
```

```
               --use_profile_ranges_km 100_300_500_600_700

 --atmosfile1d  <filename>  Uses an ASCII 1D atmosphere file.
                           In this case the run will just be range-independent.

OPTIONAL [defaults]:
 --maxheight_km            Calculation grid height in km above MSL [150 km]
 --zground_km              Height of the ground level above MSL [0 km]
 --Nz_grid                 Number of points on the z-grid from ground to maxheight [20000]
 --sourceheight_km         Source height in km Above Ground Level (AGL) [0]
 --receiverheight_km       Receiver height in km AGL [0]
 --maxrange_km             Maximum horiz. propagation distance from origin [1000 km]
 --rng_step                A usually fractional number specifying the range step
                           in wavelengths: e.g. --rng_step 0.1 means a step
                           of 0.1*wavelength [default is 0.1].
 --ground_impedance_model  Name of the ground impedance models to be employed:
                           [rigid], others TBD
 --wind_units              Specify 'kmpersec' if the winds are given
                           in km/s [ mpersec ]
 --n_pade                  Number of Pade coefficients [4]

 --starter_type            Specifies one of 3 available PE starter
                           fields: gaussian, greene, modal.
                           The default is 'gaussian'.
                           'modal' requires a precomputed starter field
                           obtained by running Modess with option
                           --modal_starter_file.




 --use_profile_ranges_km
                           e.g. --use_profile_ranges_km  20_50_80.5_300
                           The profiles at certain ranges specified by numbers
                           (in km) in a string such as 20_50_80.5_300 are
                           requested in the propagation. Note that underscores
                           are necessary to separate the numbers.
                           In this example the ranges at which the profiles
                           are considered are: 0, 20, 50, 80.5, 300 km i.e.
                           0 is always the first distance even if not specified.
                           Note also that these are requested ranges;
                           however the left-closest profile available
                           in the .env file will actually be used;
                           for instance we request the profile at 300 km
                           but in the .env file the left-closest profile
                           may be available at 290 km and it is the one used.
                           This convention may change in the future.
                           This option is used in conjunction with
                               --use_1D_profiles_from_dir

 --use_profiles_at_steps_km
                           e.g. --use_profiles_at_steps_km 100
                           The profiles are requested at equidistant intervals
                           specified by this option [1000]
```

```
                          This option is used in conjunction with
                              --use_1D_profiles_from_dir

 --use_attn_file          Use it to specify a file name containing user-provided
                          attenuation coefficients to be loaded instead of
                          the default Sutherland-Bass attenuation.
                          The text file should contain two columns:
                              height (km AGL) and
                              attenuation coefficients in np/m.


FLAGS (no value required):
 --ncpatoy                Use built-in NCPA canonical profile
 --write_2D_TLoss         Outputs the 2D transmission loss to
                          default file: tloss_2d.pe
 --do_lossless            Computation is done with no atm. absorption


 The column order of the output files is as follows (P is complex pressure):
  tloss_1d.pe:           r, 4*PI*Re(P), 4*PI*Im(P)
  tloss_2d.pe:        r, z, 4*PI*Re(P), 4*PI*Im(P)


-------------------------------------------------------------------------
Examples to run with various options (from the 'samples' directory):

    ../bin/pape  --ncpatoy --azimuth 90 --freq 0.1 --write_2D_TLoss

    ../bin/pape --g2senvfile g2sgcp2011012606L.jordan.env
                --atmosfileorder zuvwtdp --skiplines 0 --azimuth 90
                --freq 0.3 --sourceheight_km 0 --receiverheight_km 0
                --maxheight_km 180 --starter_type gaussian --n_pade 6
                --maxrange_km 500

    ../bin/pape --atmosfile1d NCPA_canonical_profile_zuvwtdp.dat
                --atmosfileorder zuvwtdp --skiplines 0 --azimuth 90 --freq 0.1
                --sourceheight_km 0 --receiverheight_km 0 --maxheight_km 180
                --starter_type gaussian --n_pade 4 --maxrange_km 500

    ../bin/pape --use_1D_profiles_from_dir ../samples/profiles
                --atmosfileorder zuvwtdp --skiplines 1 --azimuth 90 --freq 0.1
                --sourceheight_km 0 --receiverheight_km 0 --maxheight_km 180
                --starter_type gaussian --n_pade 6 --maxrange_km 1000
                --use_profiles_at_steps_km 20

    ../bin/pape --use_1D_profiles_from_dir ../samples/profiles
                --atmosfileorder zuvwtdp --skiplines 1 --azimuth 90 --freq 0.1
                --sourceheight_km 0 --receiverheight_km 0 --maxheight_km 180
                --starter_type gaussian --n_pade 6 --maxrange_km 1000
                --use_profile_ranges_km 0_20_60_400
```

The functionality of **pape** is much like that of **ModessRD1WCM**, see section 7.2. In particular, the input of atmospheric profiles is much the same, with the exception that **pape** has a flag for the use of the NCPA toy model. Further, if a G2S `.env` file is used as input the set of atmospheric profiles are read in order

from the `.env` file itself; `--use_profile_ranges_km` and `--use_profiles_at_steps_km` have no effect. The two programs share the same options and flags with some exceptions. Like all PE models, **pape** requires the use of a starter to simulate a near field source function. **pape** supports the use of a Gaussian starter, a Greens function starter and a modal starter. The Modal starter requires that there be a modal starter file that can be created with **Modess**. The option `--rng_step` sets the step size in the PE's marching scheme in fractions of a wavelength. One may use `--n_pade` to set the order of the Pade approximation to use. In addition, to run **pape** without attenuation requires the use of the flag `--do_lossless`.

9.3 Running pape: examples

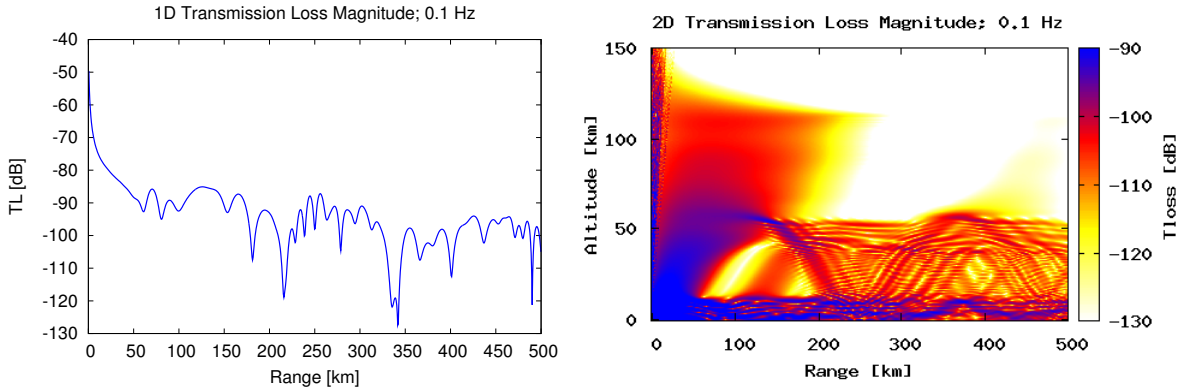

Figure 19: 1D transmission loss magnitude at 0.3 Hz obtained with **pade** for eastward ground-to-ground propagation in the range dependant profiles provided by a G2S `.env` binary file. Shown are the 1D lossless transmission loss magnitude and lossy transmission loss magnitude and the 2D lossy transmission loss magnitude.

The following example illustrates the functionality and inputs of ModessRD1WCM using a user-provided G2S `.env` file.

```
../bin/pape --g2senvfile g2sgcp2011012606L.jordan.env
            --atmosfileorder zuvwtdp --skiplines 0 --azimuth 90
            --freq 0.3 --sourceheight_km 0 --receiverheight_km 0
            --maxheight_km 180 --starter_type gaussian --n_pade 6
            --maxrange_km 500
```

The resulting data files are plotted in figure 19.

The following example illustrates the functionality and inputs of ModessRD1WCM using user-provided range dependant atmospheric profiles provided by a directory of one-dimensional ascii files.

```
../bin/pape --use_1D_profiles_from_dir profiles
            --atmosfileorder zuvwtdp --skiplines 1 --azimuth 90 --freq 0.1
            --sourceheight_km 0 --receiverheight_km 0 --maxheight_km 180
            --starter_type gaussian --n_pade 6 --maxrange_km 1000
            --use_profile_ranges_km 0_20_60_400
```

The resulting data files are plotted in figure 20.

9.4 tdpape - Time Domain Pade Parabolic Equation

tdpape provides Fourier synthesis of broad band signals using frequency-domain solutions obtained with

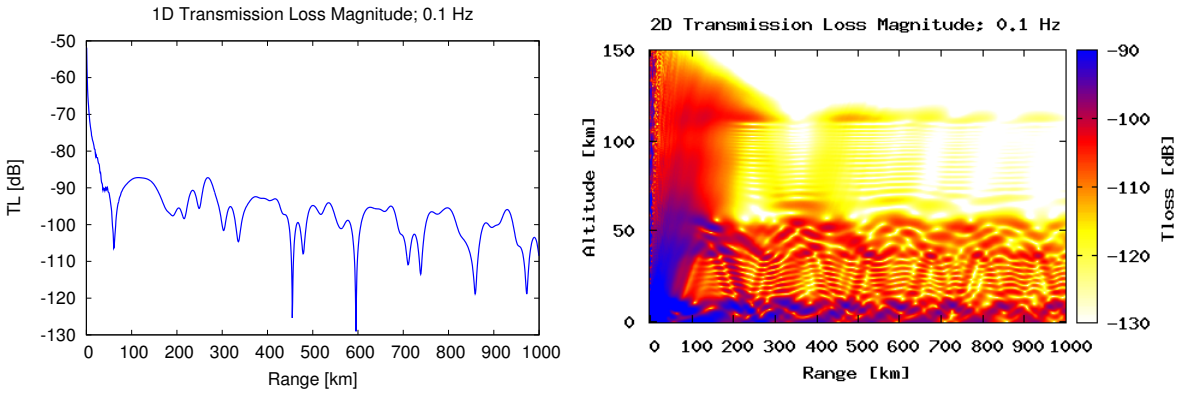

Figure 20: 1D transmission loss magnitude at 0.1 Hz obtained with **pade** for eastward ground-to-ground propagation in the range dependant profiles provided in the directory `profiles`. Shown are the 1D lossless transmission loss magnitude and lossy transmission loss magnitude and the 2D lossy transmission loss magnitude.

`pape`. It has functionality similar to **ModBB** (section 7.2) except that range dependant atmospheric profiles are supported. See Chapter 8 of reference [5] for more details on Fourier synthesis of broad band signals.

9.5 Running tdpape

To run **tdpape** make sure its executable is in the system's path and enter

```
tdpape [--option1 val1] [--option2 val2] [...] [--flag1] [...]
```

on a command line. Generally, options are followed by values, either numbers, strings or filenames. Flags are generally not. Entering `tdpape` without any options or flags sends the following help page to the screen:

```
The options below can be specified at the command line or in a colon-separated
file "tdpape.options". Command-line options override file options.
Be sure to precede all options with two minuses (--).

 --help -h                Print this message and exit


To propagate a pulse (waveform), 2 steps must be completed:
 1. Pre-computed single-frequency output files from pape must be available
    and stored in the same directory. If N frequencies were computed
    the template for the file names is
    <#n>papeTL<freq>; e.g. 004_papeTL_0.87776
    Please refer to script xrun_papeBB.sh for an example on how to
    compute single-frequency files in batch mode.

 2. Perform pulse propagation for 2 scenarios:
    a. source-to-one-receiver at one range (see option --pulse_prop_src2rcv)
    b. source-to-several-receivers at equally spaced ranges (i.e. on a grid)
       (see option --pulse_prop_src2rcv_grid)

 Output:
```

```
   The text output file has the column order:
   for option --pulse_prop_src2rcv: | Time (seconds) | Waveform |
   for option --pulse_prop_src2rcv_grid:
   | Range [km] | Time [seconds] | Waveform |
   Here 'Range' changes to reflect all receiver ranges on the defined grid.

 Four types of sources are available:
       delta function              -> see option --get_impulse_resp
       built-in pulse              -> see option --use_builtin_pulse
       user-provided spectrum file -> see option --src_spectrum_file
       user-provided waveform file -> see option --src_waveform_file


Options:
 --pulse_prop_src2rcv <directory name of pre-computed pape files>
                    Propagate pulse from source to 1 receiver
                    at a distance specified by option --range_R_km;
 --range_R_km       Propagate pulse to this range [km]
 --waveform_out_file <waveform filename>   Name of the waveform output file.

 --pulse_prop_src2rcv_grid <directory name of pre-computed pape files>
                    Propagate pulse from source to array of
                    horizontally equally-spaced receivers

 REQUIRED additional options:
 --R_start_km       Propagation from this range to R_end_km in DR_km steps.
 --R_end_km         Pulse is propagated from R_start_km to this range.
 --DR_km            Range step to propagate from R_start_km to R_end_km.

 OPTIONAL [defaults]:
 --max_celerity     Maximum celerity [300 m/s].
 --f_center         The center frequency of the pulse; must be <= [f_max/5].

SOURCE TYPE options: Use one of the following 4 options to specify the source:
 --get_impulse_resp       Flag to use a (band-limited) delta function as source and
                          to output the impulse response (this is the default).
 --use_builtin_pulse      Flag to request the use of the built-in source pulse.
 --src_spectrum_file      Specify the file name of the source spectrum
                          at positive frequencies. The file must have 3 columns
                             | Freq | Real(Spectrum) | Imag(Spectrum) |
 --src_waveform_file      Specify the file name of the user-provided
                          source waveform. The file must have 2 columns
                             |Time | Amplitude |
   If none of then source type options are specified the delta function source
   is the default i.e. the output is the impulse response.

 QUICK-START EXAMPLES (run from the 'samples' directory):
 (Assume that the pre-computed single frequency files reside in myTDPape_dir.)

 Example 1: Pulse propagation to a point on the ground at range_R_km
            and output the impulse response:

    ../bin/tdpape --pulse_prop_src2rcv myTDPape_dir --range_R_km 240
                  --waveform_out_file mywavf.dat --max_celerity 320 --get_impulse_resp
```

```
Example 2: Pulse propagation to a point on the ground at range_R_km
           and employ the built-in source pulse:

   ../bin/tdpape --pulse_prop_src2rcv myTDPape_dir --range_R_km 240
             --waveform_out_file mywavef.dat --max_celerity 320 --use_builtin_pulse

Example 3: Pulse propagation to several points on the ground 20 km apart
           and employ the user-provided source waveform:

   ../bin/tdpape --pulse_prop_src2rcv_grid  myTDPape_dir
             --R_start_km 200 --R_end_km 240 --DR_km 20
             --waveform_out_file mywavef.dat --max_celerity 320
             --src_waveform_file source_waveform_input_example.dat

Example 4: Pulse propagation to a point on the ground at range_R_km
           and employ the user-provided source spectrum:

   ../bin/tdpape --pulse_prop_src2rcv myTDPape_dir --range_R_km 240
               --waveform_out_file mywavf.dat --max_celerity 320
               --src_spectrum_file source_spectrum_example.dat
```

9.6 Running tdpape: example

Here is a simple example run to illustrate the main input for `tdpape`.

```
../bin/tdpape --pulse_prop_src2rcv myTDPape_dir --range_R_km 240
              --waveform_out_file mywavf.dat --max_celerity 320 --get_impulse_resp
```

In this example the impulse response for ground to ground propagation to a receiver at range = 240 km is calculated.

The directory `myTDPapedir` should contain single-frequency output files from `pape` covering the spectrum band desired. Assuming that band of interest contains N frequencies the template for the file names is `<#n>_papeTL_<freq>` e.g. `004_papeTL_0.87776`. The script `xrun_papeBB.sh` is a shell script providing an example on running `pape` in batch mode and storing single frequency output in the a desired directory (e.g.`myTDpape_dir`).

10 raytrace.2d and raytrace.3d - Geometric Acoustics

Geometrical acoustics is not heavily supported because of the availability of advanced geometrical acoustics packages elsewhere. See, for example, the GeoAc package available from the Los Alamos National Laboratory at https://github.com/philblom/GeoAc/archive/master.zip. Included here are **raytrace.2d** and **raytrace.3d**. These are range-independent geometric infrasound propagation algorithms for computing raypaths and signal transmission loss relative to 1 km in the Effective Sound Speed Approximation. The algorithms return a 2-D (r and z) or 3-D (x, y, and z) trace of the raypath calculated for a given elevation and azimuth. The 2-D calculation uses the effective sound speed approximation for its computation, while the 3-D calculation uses a moving-medium formulation.

10.1 Mathematical Background: 2-D

The mathematical structure of the 2-D geometrical acoustics module is fully described in [2]. Let r be the range along the propagation path, z be the altitude, θ be the takeoff angle, and $c_{eff}(z)$ be the effective sound speed in the propagation direction at z. Define the following auxiliary quantities:

$$\mathcal{R} \equiv \frac{\partial r}{\partial \theta}, \mathcal{Z} \equiv \frac{\partial z}{\partial \theta}, \eta \equiv \frac{\partial \zeta}{\partial \theta} \tag{7}$$

The 2-D geometric acoustics calculation is then performed by using a 4th-order Runge-Kutta method to solve the following system of 6 equations:

$$\frac{dr}{ds} = \frac{c_{eff}(z)}{c_{eff}(0)} \cos\theta \tag{8}$$

$$\frac{dz}{ds} = c_{eff}(z)\zeta(s,\theta) \tag{9}$$

$$\frac{d\zeta}{ds} = -\frac{c'_{eff}(z)}{c^2_{eff}(z)} \tag{10}$$

$$\frac{d\mathcal{R}}{ds} = \frac{c'_{eff}(z)}{c_{eff}(0)} \mathcal{Z}(s,\theta)\cos\theta - \frac{c_{eff}(z)}{c_{eff}(0)} \sin\theta \tag{11}$$

$$\frac{d\mathcal{Z}}{ds} = c'_{eff}(z)\mathcal{Z}(s,\theta)\zeta(s,\theta) + c_{eff}(z)\eta(s,\theta) \tag{12}$$

$$\frac{d\eta}{ds} = \left[2\left(\frac{c'_{eff}(z)}{c_{eff}(z)}\right)^2 - \frac{c''_{eff}(z)}{c_{eff}(z)}\right] \frac{\mathcal{Z}(s,\theta)}{c_{eff}(z)} \tag{13}$$

10.2 Running raytrace.2d

Making sure that the executable for **raytrace.2d** is in the system's path, it can be run by entering

```
raytrace.2d [--option1 val1] [--option2 val2] [...] [--flag1] [...]
```

on a command line. Generally, options are followed by values, either numbers, strings or filenames. Flags are not. Entering `raytrace.2d` without any options or flags sends the following help page to the screen:

```
Usage:

The options below can be specified in a colon-separated file "raytrace.2d.options" or at
the command line.  Command-line options override file options.
```

```
 --help -h                Print this message and exit

To use an arbitrary 1-D atmospheric profile in ASCII format (space or comma-separated)
the following options apply:
REQUIRED (no default values):
 --atmosfile  <filename>  Uses an ASCII atmosphere file
 --atmosfileorder         The order of the (z,t,u,v,w,p,d) fields in the ASCII file
                          (Ex: 'ztuvpd')
 --elev                   Value in range (-90,90)
 --azimuth                Value in range [0,360), clockwise from north
 --maxraylength           Maximum ray length to calculate (km) [none]
OPTIONAL [defaults]:
 --skiplines              Lines at the beginning of the ASCII file to skip [0]
 --maxelev                Maximum elevation angle to calculate [--elev value]
 --delev                  Elevation angle step [1]
 --maxazimuth             Maximum azimuth to calculate [--azimuth value]
 --dazimuth               Azimuth angle step [1]
 --sourceheight           Height at which to begin raytrace [ground level]
 --maxheight              Height at which to cut off calculation [150 km]
 --maxrange               Maximum distance from origin to calculate (km) [no maximum]
 --stepsize               Ray length step size for computation, km [0.01]
 --skips                  Maximum number of skips to allow.  Use 0 for no limits.  [0]
 --wind_units             Specify 'kmpersec' if the winds are given in km/s [mpersec]
FLAGS (no value required):
 --partial                Report the final, incomplete raypath as well as the complete
                          bounces.

Examples (run from 'samples' directory):
../bin/raytrace.2d --azimuth 90 --elev 1 --delev 1 --maxelev 45 --skips 1 \
--atmosfile NCPA_canonical_profile_zuvwtdp.dat --atmosfileorder zuvwtdp \
--maxraylength 800 --maxheight 140
cat raypath_az090* > raypaths.2d.dat
rm raypath_az090*
```

To use **raytrace.2d** an ascii file containing the atmospheric specifications must be loaded using the option `--atmosfile` followed by the filename. The order of the columns in the specification file needs to be specified using `--atmosfileorder` followed by a string indicating the order; eg. zuvwtdp if the columns orders are altitude, zonal wind, meridional wind, vertical wind, temperature, density and pressure. Units are as indicated. The option `--skiplines`, followed by a non-negative integer, is generally used to skip any header lines that might be in the file.

The output is a set of space-delimited ASCII file with columns for r (range, km), z (height, km), and A (amplitude relative to 1km). A shell-style commented header contains summary information about the azimuth and elevation angles used to generate the file, as well as calculated values for turning height, final range, travel time and celerity. Each file is named as "raypath_az[azimuth]_elev[elevation].txt".

10.3 Running raytrace.2d: examples

The **raytrace.2d** help page ends with one example command, a simple example illustrating the primary input modes for raytrace.2d. It is assumed that the user runs it in the `samples` directory. Note that if the **ncpaprop** `bin` directory is in the system's path one may enter `raytrace.2d` rather than `../bin/raytrace.2d`. The command line entry for the example is

```
../bin/raytrace.2d --azimuth 90 --elev 1 --delev 1 --maxelev 45 --skips 1 \
```

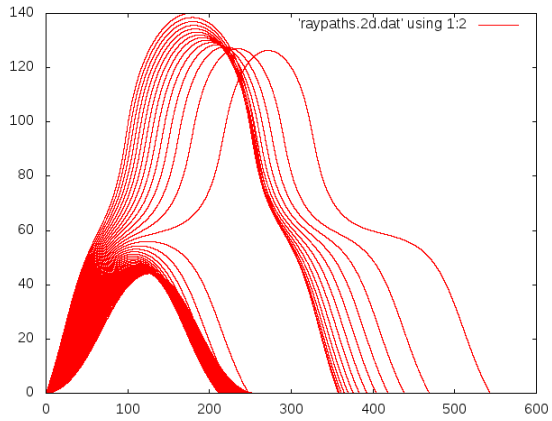

Figure 21: 2D ray trace output for the example commands provided.

```
--atmosfile NCPA_canonical_profile_zuvwtdp.dat \
--atmosfileorder zuvwtdp --maxraylength 800 --maxheight 140
```

Ground-to-ground propagation in the NCPA toy atmosphere (see Figure 1) at 90 degrees azimuth (from North), eastward propagation is modelled Propagation parameters are given in `raytrace.2d.options` or, as here, can be specified on the command line. The calculation is stopped if the ray exceeds 140 km in altitude or hits the ground (only 1 skip is requested), or if the total raypath length exceeds 800 km. The atmospheric profiles are specified in a column-based text file `NCPA_canonical_profile_zuvwtdp.dat`. The column order is specified with option `--atmosfileorder`. In the above example the profile file has 7 columns in the following order `zuvwtdp` (refer to section 3.2 for an explanation of atmospheric specifications). The atmospheric file used should always extend past the maximum height used for the calculation, or the program will crash.

The output from the example command is shown in Figure 21.

10.4 Mathematical Background: 3-D

The mathematical structure of the 3-D geometrical acoustics module is fully described in [1]. Let $\vec{x}$ be the displacement vector and ν be the Eikonal equation for propagation in three dimensions, $\vec{c_p}$ be the propagation velocity, c be the thermodynamic sound speed, and $\vec{v_0}$ be the wind. The 3-D module uses a 4th-order Runge-Kutta method to solve the coupled equations

$$\frac{\partial \vec{x}}{\partial s} = \frac{\vec{c_p}}{c_p} \tag{14}$$

$$\frac{\partial \nu_j}{\partial s} = -\frac{1}{c_p}\left[\nu \frac{\partial c}{\partial x_j} + \vec{\nu} \cdot \frac{\partial \vec{v_0}}{\partial x_j}\right] \tag{15}$$

Twelve other equations are also solved, which represent

$$\frac{\partial}{\partial s}\left(\frac{\partial \vec{r}}{\partial \theta}, \frac{\partial \vec{\nu}}{\partial \theta}, \frac{\partial \vec{r}}{\partial \phi}, \frac{\partial \vec{\nu}}{\partial \phi}\right) \tag{16}$$

These equations are voluminous in their expanded forms and are not reproduced here.

10.5 Running raytrace.3d

Making sure that the executable for **raytrace.3d** is in the system's path, it can be run by entering

```
raytrace.3d [--option1 val1] [--option2 val2] [...] [--flag1] [...]
```

on a command line. Generally, options are followed by values, either numbers, strings or filenames. Flags are not. Entering `raytrace.3d` without any options or flags sends the following help page to the screen:

```
Usage:

The options below can be specified in a colon-separated file "raytrace.3d.options" or at
the command line.  Command-line options override file options.
 --help -h                Print this message and exit

To use an arbitrary 1-D atmospheric profile in ASCII format (space or comma-separated)
the following options apply:
REQUIRED (no default values):
 --atmosfile  <filename>  Uses an ASCII atmosphere file
 --atmosfileorder         The order of the (z,t,u,v,w,p,d) fields in the ASCII file
                          (Ex: 'ztuvpd')
 --elev                   Value in range (-90,90)
 --azimuth                Value in range [0,360), clockwise from north
 --maxraylength           Maximum ray length to calculate (km) [none]
OPTIONAL [defaults]:
 --skiplines              Lines at the beginning of the ASCII file to skip [0]
 --maxelev                Maximum elevation angle to calculate [--elev value]
 --delev                  Elevation angle step [1]
 --maxazimuth             Maximum azimuth to calculate [--azimuth value]
 --dazimuth               Azimuth angle step [1]
 --sourceheight           Height at which to begin raytrace [ground level]
 --maxheight              Height at which to cut off calculation [150 km]
 --maxrange               Maximum distance from origin to calculate (km) [no maximum]
 --stepsize               Ray length step size for computation, km [0.01]
 --skips                  Maximum number of skips to allow.  Enter 0 for no maximum. [0]
 --wind_units             Specify 'kmpersec' if the winds are given in km/s [mpersec]
 FLAGS (no values required):
 --partial                Report the final, incomplete raypath as well as the complete
                          bounces.

Examples (run from 'samples' directory):
../bin/raytrace.3d --azimuth 90 --elev 1 --delev 1 --maxelev 45 --skips 1 \
--atmosfile NCPA_canonical_profile_zuvwtdp.dat --atmosfileorder zuvwtdp \
--maxraylength 800 --maxheight 140 --skiplines 1
cat raypath_az090* > raypaths.3d.dat
rm raypath_az090*
```

To use **raytrace.3d** an ascii file containing the atmospheric specifications must be loaded using the option `--atmosfile` followed by the filename. The order of the columns in the specification file needs to be specified using `--atmosfileorder` followed by a string indicating the order; eg. zuvwtdp if the columns orders are altitude, zonal wind, meridional wind, vertical wind, temperature, density and pressure. Units are as indicated. The option `--skiplines`, followed by a non-negative integer, is generally used to skip any header lines that might be in the file.

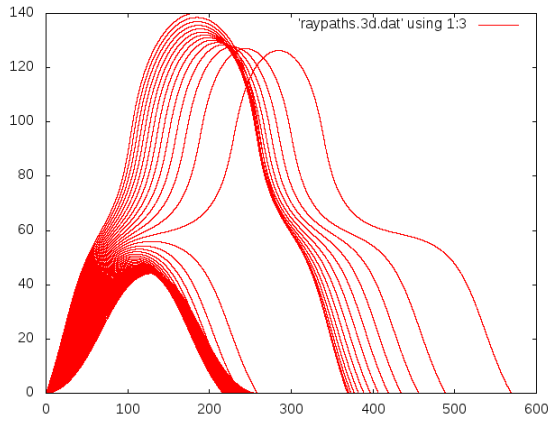

Figure 22: 3D ray trace output for the example command provided.

The output is a set of space-delimited ASCII file with columns for x (km), y (km), z (km), A (relative amplitude), and J (Jacobian). A shell-style commented header contains summary information about the azimuth and elevation angles used to generate the file, as well as calculated values for turning height, final range, travel time and celerity. Each file is named as "raypath_az[azimuth]_elev[elevation].txt".

10.6 Running raytrace.3d: examples

The **raytrace.3d** help page ends with one example command, a simple example illustrating the primary input modes for raytrace.3d. It is assumed that the user runs it in the `samples` directory. Note that if the **ncpaprop** `bin` directory is in the system's path one may enter `raytrace.3d` rather than `../bin/raytrace.3d`. The command line entry for the example is

```
../bin/raytrace.3d --azimuth 90 --elev 1 --delev 1 --maxelev 45 --skips 1 \
                   --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
                   --atmosfileorder zuvwtdp --maxraylength 800 --maxheight 140
```

Ground-to-ground propagation in the NCPA toy atmosphere (see Figure 1) at 90 degrees azimuth (from North), eastward propagation is modelled Propagation parameters are given in `raytrace.3d.options` or, as here, can be specified on the command line. The calculation is stopped if the ray exceeds 140 km in altitude or hits the ground (only 1 skip is requested), or if the total raypath length exceeds 800 km. The atmospheric profiles are specified in a column-based text file `NCPA_canonical_profile_zuvwtdp.dat`. The column order is specified with option `--atmosfileorder`. In the above example the profile file has 7 columns in the following order `zuvwtdp` (refer to section 3.2 for an explanation of atmospheric specifications). The atmospheric file used should always extend past the maximum height used for the calculation, or the program will crash.

The output from the example command is shown in Figure 22. Column 1 (x) of the output file is used as the independent variable rather than the magnitude of the x-y vector for the sake of simplicity, since propagation is almost due eastward.

11 wnlrt -Weakly Non Linear Ray Tracing

wnlrt is a non-linear geometrical acoustics program that calculates the second order (weak shock theory) transport equation along a ray path. The program requires geometrical acoustics input. In particular, the ray path and the linear amplitudes (the Jacobean for the transformation from cartesian to ray coordinates) must be provided by the user as input. The algorithm uses a spilt step approach in which the ray path is traversed in small steps. Non-linear effects are treated in the time domain in each step. The solution is then transformed to the frequency domain where atmospheric attenuation is applied.

11.1 Mathematical Background

wnlrt uses a weakly non-linear ray tracing algorithm (the original code was developed by Joel B. Lonzaga) to propagate a signal from an impulsive source to a receiver through a range dependent 3-d atmosphere with winds. For the theory and algorithm refer to reference [7].

11.2 Running wnlrt

wnlrt is not a stand-alone program but requires input from a geometrical acoustics program. The required data and file format is detailed in the help page provided by the program and reproduced below. Making sure that the executable for **wnlrt** is in the system's path, it can be run by entering

```
wnlrt [--option1 val1] [--option2 val2] [...] [--flag1] [...]
```

on a command line. Generally, options are followed by values, either numbers, strings or filenames. Flags are not. Entering `wnlrt` without any options or flags sends the following help page to the screen:

```
'wnlrt' uses a weakly non-linear ray tracing algorithm (originally authored by
Joel B. Lonzaga) to propagate a signal from an impulsive source to a receiver
through a stratified atmosphere with winds. For the theory and algorithm refer to:
Joel B. Lonzaga, Roger M. Waxler, Jelle D. Assink and Carrick L. Talmadge,
"Modeling waveforms of infrasound arrivals from impulsive sources using
weakly non-linear ray theory", Geophys. J. Int. (2015) 200, 1337-1361.

Usage:
./wnlrt [--option1 val1] [--option2 val2] [--flag1] [...]

The options below can be specified at the command line or in a colon-separated
file "wnlrt.options". Command-line options override file options.
Be sure to precede all options with two minuses (--). The program options can
be of two kinds: pairs of [--option_name value] or flags. The values can be
numbers or strings (arrays of characters). The flags do not take a value on the
command line; they are boolean switches signaling the program for a certain
action to occur.

 --help -h            Print this message and exit

PREREQUISITE
This code requires the user to provide an eigenray file obtained from an external
ray-tracing package GeoAc1.1.1 authored by Phil Blom, currently at Sandia
National Lab, and whose work began at NCPA under the direction of Roger Waxler.
(As an example see the included ToyAtmo_Eigenray-0.dat file.)
The 15-column eigenray file has a 4-line header providing information about
```

```
the source and receiver locations, the ray launch angles and the column names
as in the following example:
  Source Location (kilometers) : (0, 0, 0).
  Receiver Location (kilometers) : (350, 0, 0).
  theta = 18.0303, phi = 90,  c0(zground) = 0.340322 km/s
  # x [km]  y [km]  z [km]  Geo. Atten. [dB]  Atmo. Atten. [dB] Travel Time [s]  rho
  c [km/s]  u [km/s]  v [km/s]  w [km/s]  Slowness_x [s/km]  Slowness_y [s/km]
  Slowness_z [s/km]  Jacobian [km^2/rad^2]

REQUIRED options:
 --eigenrayfile        Provide name of previously obtained eigenray file.
                       Currently this eigenray file is obtained by running
                       the modified version of GeoAc1.1.1 that outputs
                       relevant parameters along the ray path.

OPTIONAL options [defaults]:
 --waveform            Provide the type of waveform at the source.
                       Can be Nwave or pulse [Nwave].
 --ampl                Provide the initial waveform amplitude. [1]
 --duration            Provide the initial waveform duration [0.5 secs]

OUTPUT text files:
 ray_params.dat
                       Contains ray info and scaled pressure; 6 columns
                       [ x, y, z, raypath_length, travel_time, scaled_pressure ]

 pressure_wf_evolution.dat
                       Stores N waveforms at N time steps along the ray.
                       Contains (N+1) columns in the follwoing order:
                       [ reduced_time, waveform_at_step_1 ... waveform_at_step_N ]

 final_waveform_spectrum.dat
                       Stores waveform spectrum at the ray's end point
                       Contains 3 columns: [ frequency, real part, imag part ]

QUICK-START EXAMPLES:
./wnlrt --eigenrayfile ToyAtmo_Eigenray-0.dat
./wnlrt --eigenrayfile ToyAtmo_Eigenray-0.dat --waveform Nwave --ampl 500 --duration 0.5
```

11.3 Running wnrt: example

Here is a simple example run to illustrate the main input for `wnlrt`.

```
./wnlrt --eigenrayfile ToyAtmo_Eigenray-0.dat
```

This example is for of an N-wave of unit amplitude and 0.5 second duration along a raypath defined in file ToyAtmo_Eigenray-0.dat. The raypath is shown in the left panel in Figure 23 . In the right panel a waterfall of the (scaled) N-wave evolution is presented at various points on the raypath. It illustrates the non-linear effect of propagation: the N-wave stretches gradually and "softens" in frequency content due to absorption in the air.

The actual eigenray files to be used with this module are obtained from an external ray-tracing package GeoAc1.1.1. Its author, Phil Blom, began work on the linear ray tracing under Roger Waxler's direction at the National Center for Physical Acoustics (NCPA). Phil Blom is currently (Sep. 2015) at Los Alamos

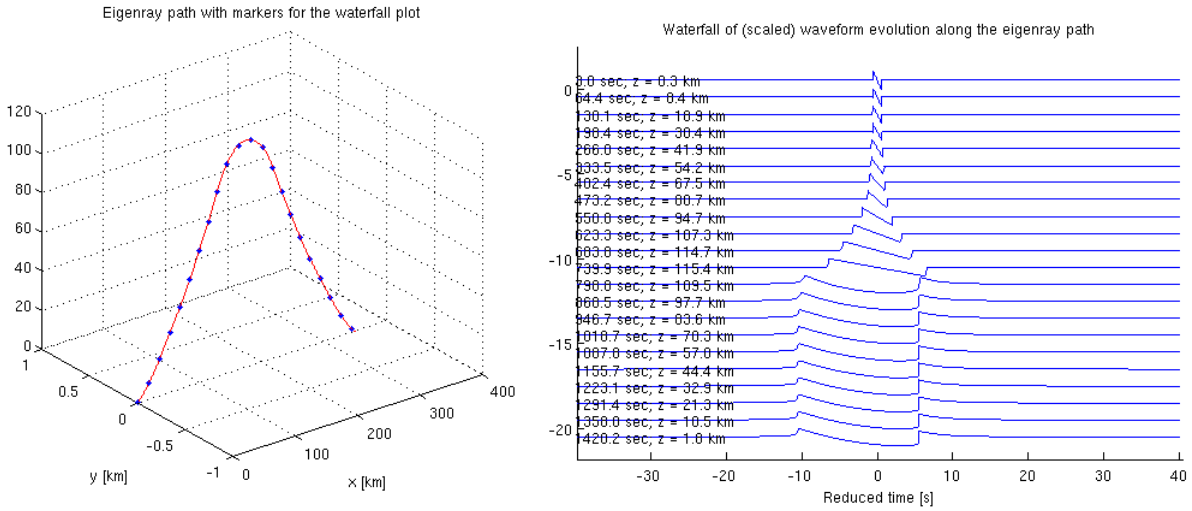

Figure 23: N-wave waveform evolution along a raypath.

National Laboratory where he continues to maintain and improve GeoAc1.1.1. The eigenray file is a 15-column text file whose format is described in the **wnlrt** help page reproduced above.

References

[1] Philip Blom. *Interaction of the Cyclonic Winds with the Infrasonic Signal Generated by a Large Maritime Storm*. PhD thesis, The University of Mississippi, 2013.

[2] Philip Blom and Roger Waxler. Impulse propagation in the nocturnal boundary layer: Analysis of the geometric component. *J. Acoust. Soc. Am.*, 131(5):pp. 3680 to 3690, 2012.

[3] N. Dunford and J. T. Schwartz. *Linear Operators, part III*. Wiley, New York, 1971.

[4] Oleg Godin. An effective quiescent medium for sound propagating through an inhomogeneous, moving fluid. *J. Acoust. Soc. Am.*, 2002.

[5] Finn B. Jensen, William A. Kuperman, Micheal B. Porter, and Henrik Schmidt. *Computational Ocean Acoustics*. Springer, NY, 2000.

[6] Lev D. Landau and E. M. Lifshitz. *Quantum Mechanics*. Pergamon, London, 1965.

[7] Joel Lonzaga, Roger Waxler, Jelle Assink, and Carrick Talmadge. Modelling waveforms of infrasound arrivals from impulsive sources using weakly non-linear ray theory. *Geophys. J. Int.*, 2015.

[8] Allan Pierce. *Acoustics*. Acoustical Society of America, Woodbury N.Y., 1989.

[9] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 3 edition, 2007.

[10] Louis C. Sutherland and Henry E. Bass. Atmospheric absorption in the atmosphere up to 160 km. *J. Acoust. Soc. Am.*, 99(3):pp. 1012 to 1032, 2004.

[11] Roger Waxler, Läslo G. Evers, Jelle Assink, and Phillip Blom. The stratospheric arrival pair in infrasound propagation. *The Journal of the Acoustical Society of America*, 137(4):1846–1856, 2015.