ncpaprop 2.1.0

NCPA Infrasound Propagation Modeling Package

Contributions from: Roger Waxler, Claus Hetzer, Jelle Assink, Doru Velea

Technical coordination: Roger Waxler

NCPA Atmospheric Acoustics

JAMIE L. WHITTEN NATIONAL CENTER FOR PHYSICAL ACOUSTICS

Koninklijk Nederlands Meteorologisch Instituut
Ministerie van Infrastructuur en Waterstaat

Roger Waxler, Claus Hetzer, Jelle Assink, and Doru Velea. (2021). https://github.com/chetzer-ncpa/ncpaprop-release: NCPAprop v2.1.0 (v2.1.0). Zenodo. https://doi.org/10.5281/zenodo.5562713

ncpaprop 2.1.0

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. Stratified and range dependent atmospheric models are supported, depending on the package. 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 [9, 5] 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 [14].

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. The PE model is an effective sound speed model that supports both stratified and range-dependent atmospheric specifications. Atmospheric attenuation is handled in the normal mode models using first order perturbation theory. Attenuation is treated non-perturbatively in the PE model.

This manual is organized as follows. 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 mathematical 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, Red Hat, and CentOS Linux operating systems, and MacOS Mojave (10.14).

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. **PETSc** routines are also used to perform the matrix calculations required by the PE model.

2.1.3 Compilers

ncpaprop is written in C++ and was built and tested using the GNU C++ compiler **g++**, and will also compile under the MacOS native **clang** compiler. The **PETSc** and **SLEPc** libraries require a corresponding C compiler.

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. An additional flag can be invoked (for Linux installations only) to allow the installer to automatically download and install the prerequisites using the native package manager. The details of this process are explained below.

2.2.1 Installation Overview

To begin installation, select an installation location for the **ncpaprop** directory to reside. Clone the GitHub repository's master branch with

```
git clone https://github.com/chetzer-ncpa/ncpaprop-release.git ncpaprop
```

and cd into `ncpaprop`. Configuration will depend on whether the user has a properly-compiled pre-existing PETSc and SLEPc installation or not. Properly-compiled, in this context, means compiled with C++ support and for both real and complex data types. If this is the case, and the user wishes to link to those installations, the following environmental variables should be set:

`PETSC_DIR`
: The root directory of the **PETSc** installation.

`SLEPC_DIR`
: The root directory of the **SLEPc** installation.

`PETSC_ARCH_REAL`
: The architecture name selected for the (default) `--with-scalar-type=real` **PETSc** build. This frequently defaults to `arch-$OS-c-real` or `arch-$OS-c-debug` if the user does not specify, where `$OS` represents the operating system, and will appear as a subdirectory of `$PETSC_DIR` and `$SLEPC_DIR`.

`PETSC_ARCH_COMPLEX`
: The architecture name selected for the `--with-scalar-type=complex` **PETSc** build. This frequently defaults to `arch-$OS-c-complex` or `arch-$OS-c-complex-debug` if the user does not specify, where `$OS` represents the operating system, and will appear as a subdirectory of `$PETSC_DIR` and `$SLEPC_DIR`.

The user may then simply issue the command

```
./configure
```

to set up the build environment and make sure all prerequisites are met. Alternately, the user may request that local instances of PETSc and SLEPc be installed in the `ncpaprop` directory tree with the `--with-localpetsc` flag. This approach is recommended if the user has no previous experience with **PETSc** and/or doesn't want to bother with a system-wide installation of **PETSc**. The recommended command for this circumstance is:

```
./configure --with-localpetsc
```

Additional flags and variables to control this process include:

`--enable-autodependencies`
: Attempt to download and install missing prerequisites automatically. Use this option only if installation in the default installation locations is satisfactory. The **ncpaprop** installer currently supports the **apt-get** and **yum** package managers on Linux. This option is not supported on MacOS and such users should install **FFTW** and **GSL** using their method of choice. Use of this option on an unsupported system will result in an error.

`LDFLAGS="-L/path/to/library"`
: The user can also specify additional library search paths using this syntax.

To have the installer download and build **PETSc** and **SLEPc** local to **ncpaprop**, use approriate flags and options from the following:

`--with-localpetsc`
: Download and build the current `maint` branch of **PETSc**. Use of this flag also sets `--with-localslepc`.

`--with-localslepc`
: Download and build the latest version of **SLEPc**, which at the time of release is **3.16.0**.

`--with-localpetscdebug`
: Build the `debug` version of **PETSc** rather than the optimized version.

`--enable-localpetscmpi`
: Build **PETSc** with MPI support. If this flag is not set, the installer will build **PETSc** with MPI support disabled..

`local_petsc_dir=VALUE`
: Override the default **PETSc** installation location (defaults to `$NCPAPROP_ROOT/extern/petsc`).

`local_slepc_dir=VALUE`
: Override the default **SLEPc** installation location (defaults to `$NCPAPROP_ROOT/extern/slepc/slepc-$VERSION)`.

`local_petsc_arch_real=VALUE`
: Override the default **PETSc** real architecture name (defaults to `arch-$OS-real` or `arch-$OS-real`).

`local_petsc_arch_complex=VALUE`
: Override the default **PETSc** complex architecture name (defaults to `arch-$OS-complex` or `arch-$OS-complex`).

`local_petsc_version=VALUE`
: Override the default **PETSc** version number for determining the release branch. By default the latest release version will be used.

`local_slepc_version=VALUE`

Override the default **SLEPc** version number for determining the download filename. This will generally be kept up-to-date by the **ncpaprop** maintainers, but is provided in the event of maintenance lag.

For example, to download and build the `3.13.4` versions of **PETSc** and **SLEPc**, use:

```
./configure --with-localpetsc local_petsc_version=3.13.4 local_slepc_version=3.13.4
```

The installation process will check the system for the presence of BLAS libraries and MPI executables; whatever is not found will be automatically downloaded and installed locally by the **PETSc** installer. This process is independent of the `--with-autodependencies` flag, which applies only to the direct prerequisites of **ncpaprop**. If this is undesirable, **PETSc** and **SLEPc** should be installed manually. In general, **PETSc** and **SLEPc** are orders of magnitude more complex than **ncpaprop**, and only limited support for their installation can be provided by the **ncpaprop** developers. Complex questions involving incompatibilies with the user's system are best directed to **PETSc** support.

2.2.2 Compilation

Once the configuration script has successfully run, a set of environmental variables will be presented to the user to be set in their login file. These variables will be used by the Makefiles to indicate where to find the **PETSc**/**SLEPc** libraries, and which version (real vs complex) to use, and are necessary for future recompilations to work properly without re-invoking the `./configure` script. The package can then be built simply by running

```
make
```

2.3 Typesetting this Manual

This manual was produced with LaTeX. The required `.tex` and graphics files are archived in the **ncpaprop** package in the `manual` directory. The main `.tex` file, `NCPA_prop_manual.tex`, is contained in the subdirectory `ncpaprop_run_manual_TeX` along with a utitlity script `runtex.sh`. Entering `. runtex.sh` produces the manual as a pdf file and moves it to the `manual` directory. Entering `. runtex.sh bib` produces the bibliography using bibtex.

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) switches 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 --help` sends a manual page to the screen listing the available flags and options with explanations.

The order of the options and flags in a command generally 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. Additionally, if an unrecognized option or flag is provided, such as a misspelling, the program will also warn the user and abort.

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, thus a base options file can be constructed and deviations from the baseline can be specified using command-line inputs. 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, thus it is often convenient to create a working directory containing all relevant inputs, and run the executables from there. 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. The files are self-describing using a formatted header that associates the column with a tag string and an indication of units. Scalar quantities may also be specified in the header. Formatted lines and normal (i.e. ignorable) comments may both be used in the header as needed; normal comments are preceded by the `#` character, while formatted lines are preceded with `#%`. Text tags are used to indicate to the modules which columns represent which types of quantities, and are expected to match the following convention:

```
Z   - altitude above MSL
U   - West-to-East wind speed
V   - South-to-North wind speed
T   - temperature
RHO - density
P   - ambient pressure
C0  - static sound speed (optional, will be calculated if not provided)
Z0  - ground elevation (scalar, optional)
```

Any unrecognized tags will be ignored. Tags beginning and ending with underscores (for example _ALPHA_) should not be used, as this convention is used for internally-calculated quantities and may cause conflicts. Scalar quantities are indicated using column number 0 and are in the format 0, <tag>, <units>, <value>, while vector quantities use the format <column>, <tag>, <units> with values provided in the columns. Altitude, as the independent quantity, must be provided in column number 1. An example of a formatted header with units and a scalar quantity demonstrates this:

```
#% 0, Z0, m, 45.0
#% 1, Z, km
#% 2, U, m/s
#% 3, V, m/s
#% 4, W, m/s
#% 5, T, degK
#% 6, RHO, g/cm3
#% 7, P, mbar
```

In this example, the static sound speed C0 will be calculated internally from the pressure and density.

Quantities may be provided in any supported units and will be converted internally as needed. Additional units may be added by the user as needed by following the directions given in the comment header in src/common/units.h. Supported units as of this release may be indicated by any of the following codes, which are case-insensitive:

```
temperature: K, DEGK, DEG K, DEGREES K
             C, DEGC, DEG C, DEGREES C
             F, DEGF, DEG F, DEGREES F
distance:    M, METERS
             KM, KILOMETERS
speed:       M/S, MPS, MPERS, M PER S, METERS PER SECOND
             KM/S, KMPS, KMPERS, KM PER S, KILOMETERS PER SECOND
pressure:    PA, PASCAL, PASCALS
             MBAR, MILLIBAR, MILLIBARS
density:     KG/M3, KGPM3, KILOGRAMS PER CUBIC METER
             G/CM3, GPCM3, GRAMS PER CUBIC CENTIMETER
direction:   DEGREES CLOCKWISE FROM NORTH, DEG CW FROM N, AZIMUTH
             DEGREES COUNTERCLOCKWISE FROM EAST, DEG CCW FROM E
angle:       DEG, DEGREES
             RAD, RADIANS
```

An example of an atmospheric profile file with the above structure is provided in the samples directory in file NCPA_canonical_profile_trimmed.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° and wind speed are illustrated in Figure 1. Alternately, a separate file containing only the descriptive header may be specified using the command-line flag --atmosheaderfile. If this flag is used, the indicated file will provide the profile metadata and any formatted header in the profile file will be ignored. An example of this file is provided in the samples directory in file sampleheader.dat and an example of the proper usage is given as the first sample command in the **WMod** package.

To model propagation through a stratified medium only a single specification file is required. Range dependent profiles for the ePape module are specified by providing a summary file that associates a range (in km) with a 1-D atmospheric file in the above format, for example:

```
0.0   profiles/profile0000.dat
89.0  profiles/profile0089.dat
179.0 profiles/profile0179.dat
269.0 profiles/profile0269.dat
359.0 profiles/profile0359.dat
```

The resulting 2-d specification is piecewise stratified with transitions at the midpoints between the indicated ranges. Additional range-dependent formats are under development.

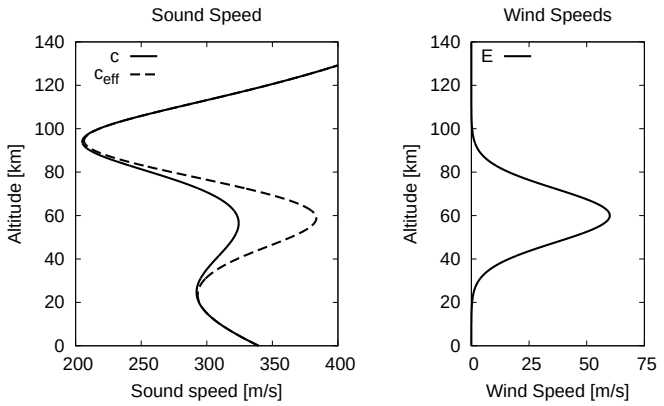


Figure 1: The NCPA model atmosphere, the "toy" model. The figure on the left shows sound speeds: the solid curve is the adiabatic sound speed and the dashed curve is the effective sound speed profile in the azimuth direction of 90 degrees (clockwise from North). The figure on the right shows the model zonal wind speed. The meridional wind speed is taken to be zero. The figures are taken from [16].

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. [7]. 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. [8]. 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 matrices are tridiagonal, allowing for extremely efficient numerical algorithms. The benefits of having tridiagonal matrices to diagonalize 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 are computed using a Sturm alogorithm [12] and the corresponding mode functions are obtained by the method of inverse iteration [10].

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

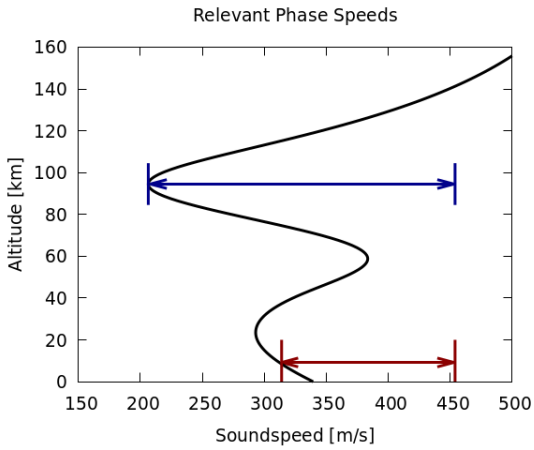


Figure 2: Phase velocities used in **Modess** for the NCPA toy atmosphere. The black curve is the effective sound speed for eastward propagation. The blue double-headed arrow shows the range of phase velocities needed in the modal sum. The red double-headed arrow shows the range of phase velocities needed 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 [7])

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

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 --help` sends the following help page to the screen:

```
By default the program 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 tloss_2d.nm. The user can also choose to propagate in N
different directions i.e. (N by 2D mode) by using the option --multiprop.

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

```
Options Control:
  --help                  Prints help test
  --paramfile             Parameter file name [modess.param]
  --printparams           Print parameter summary to screen

Required Parameters:
  --atmosfile             Atmospheric profile filename
  --freq                  Frequency of analysis (Hz)

Optional Parameters [default]:
  --maxheight_km          Maximum height of analysis in km [150.0]
  --zground_km            Ground height [take from Z0 parameter in atmosfile, or
                          0.0]
  --Nz_grid               Number of vertical grid points to use [20000]
  --sourceheight_km       Source height in km [0.0]
  --receiverheight_km     Receiver height in km [0.0]
  --maxrange_km           Maximum range in km to use for modeling [1000.0]
  --Nrng_steps            Number of range steps to use [1000]
  --ground_impedence_model
                          Impedence model to use. Currently only "rigid" is
                          supported. [rigid]
  --use_attn_file         File name containing attenuation, to override default
                          Sutherland/Bass [n/a]. Columns are
                          Height(km) Attenuation(np/m)
  --modal_starter_file    Filename to output a starter file for the pape module
  --dispersion_file       Filename to output the dispersion information
    --append_dispersion_file
                          Append results to dispersion file rather than
                          overwriting

Modes of Operation:
  --singleprop            Single azimuth propagation. Requires --azimuth
    --azimuth             Azimuth of propagation, in degrees CW from North
                          [0,360)
  --multiprop             Multiple azimuth propagation. Requires --azimuth_start,
                          --azimuth_end, and --azimuth_step
    --azimuth_start       Starting azimuth, in degrees CW from North [0,360)
    --azimuth_end         Ending azimuth, in degrees CW from North [0,360)
    --azimuth_step        Azimuth step, in degrees CW from North [0,360)

Flags:
  --write_2d_tloss        Output 2-D transmission loss to tloss_2d.nm
  --write_phase_speeds    Output phase speeds to phasespeeds.nm
  --write_speeds          Output both the phase speeds and the group speeds to
                          speeds.nm
  --write_modes           Output modes to mode_###.nm. Also implies
                          --write_speeds
  --write_atm_profile     Output atmospheric profile to atm_profile.nm
  --Lamb_wave_BC          Use admittance = -1/2*dln(rho)/dz
  --turnoff_WKB           Turn off the WKB least phase speed estimation
  --wvnum_filter          Use wavenumber filter by phase speed. Requires --c_min
                          and --c_max
    --c_min               Minimum phase speed to keep
```

```
    --c_max              Maximum phase speed to keep


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:
  Nby2D_tloss_1d.lossless.nm:  r, theta, 4*PI*Re(P), 4*PI*Im(P), (incoherent TL)
  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), rho(z_src), rho(z_rcv)
                               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 --singleprop --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
         --azimuth 90 --freq 0.1

    ../bin/Modess --singleprop --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
         --azimuth 90 --freq 0.1 --write_2d_tloss

    ../bin/Modess --multiprop --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
         --freq 0.1 --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. **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 `--multiprop`. 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. `--multiprop` 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 `--multiprop` 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 --singleprop --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
              --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).

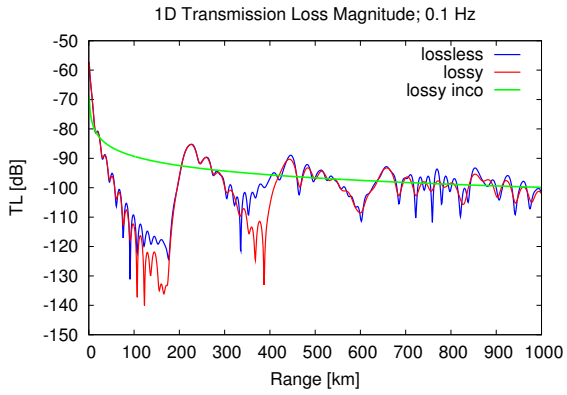


Figure 3: 1D transmission loss magnitude at 0.1 Hz obtained with **Modess** for eastward ground-to-ground propagation in the NCPA toy atmosphere shown in Figure 1. Shown are the lossless transmission loss magnitude, the lossy transmission loss magnitude and the lossy incoherent transmission loss.

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 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:

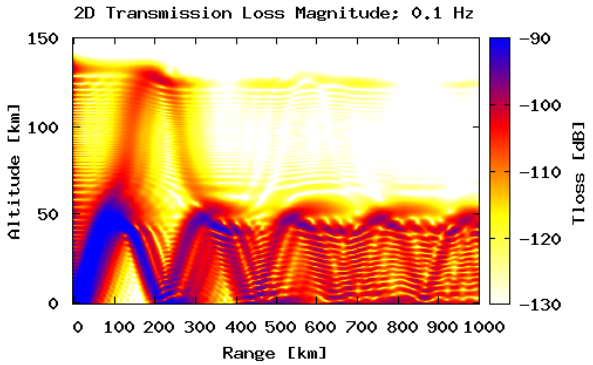


Figure 4: 2D transmission loss magnitude at 0.1 Hz obtained with **Modess** for eastward propagation in the NCPA toy model. The source is placed on the ground at the origin.

```
../bin/Modess --singleprop --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
              --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 `--multiprop` 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 --multiprop --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
              --freq 0.1 --azimuth_start 0 --azimuth_end 360 --azimuth_step 1
```

The output is written to the files `Nby2D_tloss_1d.nm` and `Nby2D_tloss_1d.lossless.nm` as described above in section 4.2. The lossless transmission losses are also computed and saved as are the incoherent transmission losses and 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 --singleprop --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
              --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.

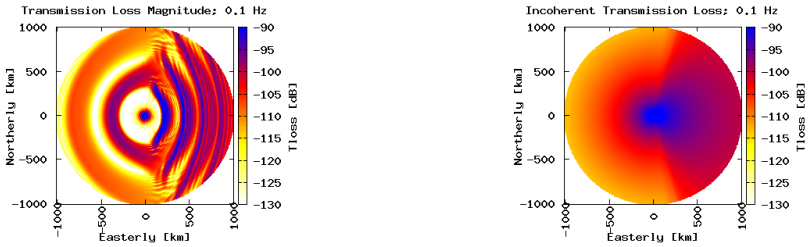


Figure 5: N by 2D transmission loss magnitude and incoherent transmission loss obtained with **Modess** for ground-to-ground propagation in the NCPA toy model atmosphere from a source at the origin to all azimuths.

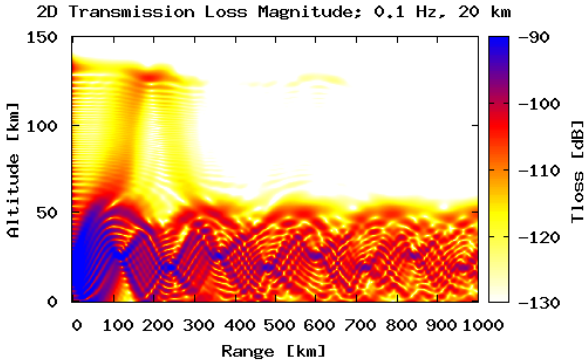


Figure 6: 2D transmission loss magnitude for propagation from an elevated source at 20 km altitude to the ground, obtained with **Modess** for propagation in the NCPA toy atmosphere.

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 **Wmod** is taken into account using first order perturbation theory. More details can be found in [1].

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 --help` sends the following help page to the screen:

```
By default the program computes the 1D transmission loss (TL) at the ground or
the specified receiver height 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 directions i.e. (N by 2D mode) by using the option --multiprop.

The options below can be specified in a colon-separated file "wmod.param" or at
the command line. Command-line options override file options.

Options Control:
  -h, --help             Prints help test
  --paramfile            Parameter file name [modess.param]
  --printparams          Print parameter summary to screen

Required Parameters:
  --atmosfile            Atmospheric profile filename
  --freq                 Frequency of analysis (Hz)

Optional Parameters [default]:
  --maxheight_km         Maximum height of analysis in km [150.0]
  --zground_km           Ground height [take from Z0 parameter in atmosfile, or
                         0.0]
  --Nz_grid              Number of vertical grid points to use [20000]
  --sourceheight_km      Source height in km [0.0]
  --receiverheight_km    Receiver height in km [0.0]
  --maxrange_km          Maximum range in km to use for modeling [1000.0]
  --Nrng_steps           Number of range steps to use [1000]
  --ground_impedence_model
                         Impedence model to use. Currently only "rigid" is
                         supported. [rigid]
  --use_attn_file        File name containing attenuation, to override default
                         Sutherland/Bass [n/a]. Columns are
```

```
                        Height(km) Attenuation(np/m)
  --dispersion_file     Filename to output the dispersion information
    --append_dispersion_file
                        Append results to dispersion file rather than
                        overwriting

Modes of Operation:
  --singleprop          Single azimuth propagation. Requires --azimuth
    --azimuth           Azimuth of propagation, in degrees CW from North
                        [0,360)
  --multiprop           Multiple azimuth propagation. Requires --azimuth_start,
                        --azimuth_end, and --azimuth_step
    --azimuth_start     Starting azimuth, in degrees CW from North [0,360)
    --azimuth_end       Ending azimuth, in degrees CW from North [0,360)
    --azimuth_step      Azimuth step, in degrees CW from North [0,360)

Flags:
  --write_2d_tloss      Output 2-D transmission loss to tloss2D.nm
  --write_phase_speeds  Output phase speeds to phasespeeds.nm
  --write_modes         Output modes to mode_###.nm. Also implies
                        --write_speeds
  --write_atm_profile   Output atmospheric profile to atm_profile.nm
  --Lamb_wave_BC        Use admittance = -1/2*dln(rho)/dz
  --turnoff_WKB         Turn off the WKB least phase speed estimation
  --wvnum_filter        Use wavenumber filter by phase speed. Requires --c_min
                        and --c_max
    --c_min             Minimum phase speed to keep
    --c_max             Maximum phase speed to keep



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:
  Nby2D_wtloss_1d.lossless.nm:  r, theta, 4*PI*Re(P), 4*PI*Im(P), (incoherent TL)
  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), rho(z_src), rho(z_rcv)
                               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 --singleprop --atmosfile profile_noheader.dat \
         --atmosheaderfile sampleheader.dat --azimuth 90 --freq 0.1

    ../bin/WMod --singleprop --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
         --azimuth 90 --freq 0.1 --write_2d_tloss --sourceheight_km 60 \
         --receiverheight_km 60

    ../bin/WMod --multiprop --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
```

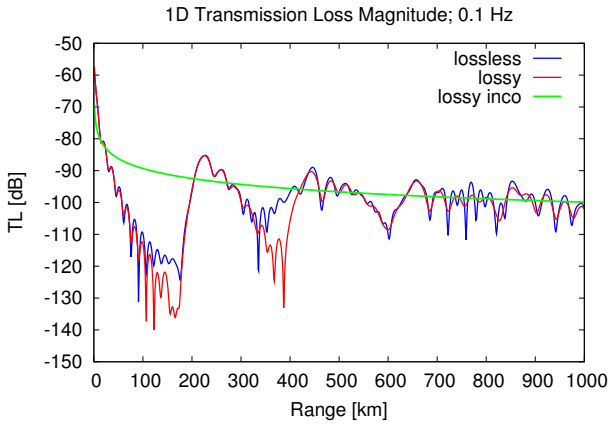


Figure 7: 1D transmission loss magnitude at 0.1 Hz obtained with **WMod** for eastward ground-to-ground propagation in the NCPA toy atmosphere shown in Figure 1. Shown are the lossless transmission loss magnitude, the lossy transmission loss magnitude and the lossy incoherent transmission loss.

```
--freq 0.1 --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 --singleprop --atmosfile profile_noheader.dat \
            --atmosheaderfile sampleheader.dat --azimuth 90 --freq 0.1
```

In this example the four required options are set, and an external header file is specified. 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 `profile_noheader.dat` that is included in the `samples` directory, and the profile metadata is provided in the additional text file `sampleheader.dat` rather than as a header in the same file.

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.

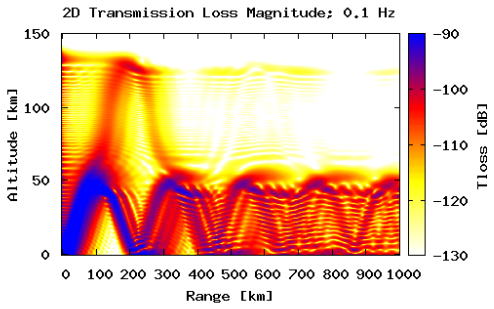


Figure 8: 2D transmission loss magnitude at 0.1 Hz obtained with **WMod** for eastward propagation in the NCPA toy model. The source is placed on the ground at the origin.

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

```
../bin/WMod --singleprop --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
            --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 above example the profile file has 7 columns in the order `zuvwtdp` (refer to section 3.2 for an explanation of atmospheric specifications).

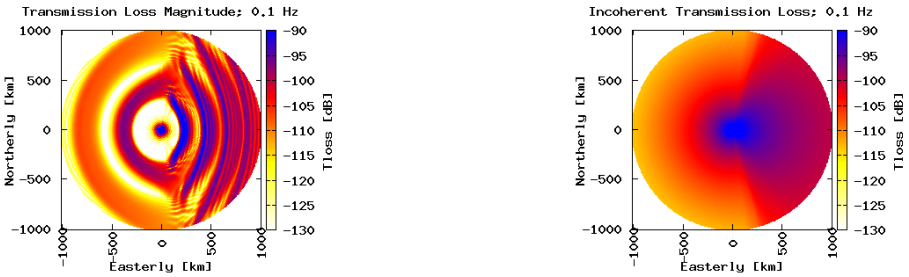


Figure 9: N by 2D transmission loss magnitude and incoherent transmission loss obtained with **WMod** for ground-to-ground propagation in the NCPA toy model atmosphere from a source at the origin to all azimuths.

In the third example the `--multiprop` 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 --multiprop --atmosfile NCPA_canonical_profile_zuvwtdp.dat \
```

```
--freq 0.1 --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 and the resulting transmission loss magnitude is plotted in Figure 9.

6 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.

6.1 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, [10]). 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 $m = 0, 1, 2, \ldots, N$,

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

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

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

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.

6.2 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

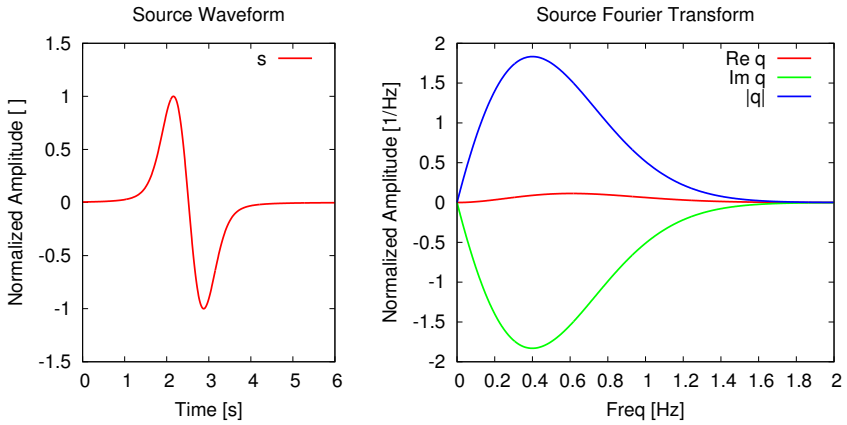


Figure 10: The built-in initial pulse with maximum Fourier transform frequency of 0.4 Hz and maximum frequency of 2 Hz. The waveform is shown in the left panel and the Fourier transform; real part, imaginary part, and magnitude; in the right panel.

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 10. 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 --help` sends the following help page to the screen:

```
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 (as in
WMod). Modess is faster but is accurate for launch angles less than 30 degrees
and low wind speeds. WMod extends the validity to higher angles and high Mach
numbers, but is slower.
To propagate a pulse, two steps must be completed:
1. A dispersion file must be calculated using the option --dispersion .
2. Pulse propagation is calculated for a selected source type:
  --source impulse : Delta function
  --source pulse1 : Built-in pulse type 1
  --source pulse2 : Built-in pulse type 2
  --source spectrum --source_file <filename> : User-supplied spectrum file
  Format: Freq Re[ spec(f) ] Im[ spec(f) ]
  --source waveform --source_file <filename> : User-supplied waveform file
  Format: Time Amplitude

The options below can be specified in a colon-separated file "modbb.param" or at
the command line. Command-line options override file options.

Options Control:
  --help                 Prints help test
  --paramfile            Parameter file name [modbb.param]
  --printparams          Print parameter summary to screen

Modes of Operation:
  --dispersion           Calculate dispersion at multiple frequencies
    --method             Calculation method. Currently supported: { modess, wmod
                         } [required]
    --dispersion_file    File to write dispersion results to [required]
    --azimuth            Azimuth of propagation, in degrees CW from North
                         [0,360) [required]
    --f_min              Minimum frequency in Hz [required]
    --f_max              Maximum frequency in Hz [required]
    --f_step             Frequency interval in Hz [required]
    --atmosfile          Atmospheric profile filename [required]
    --maxheight_km       Maximum height of analysis in km [150.0]
    --zground_km         Ground height [take from Z0 parameter in atmosfile, or
                         0.0]
    --Nz_grid            Number of vertical grid points to use [20000]
    --sourceheight_km    Source height in km [0.0]
    --receiverheight_km  Receiver height in km [0.0]
    --maxrange_km        Maximum range in km to use for modeling [1000.0]
    --Nrng_steps         Number of range steps to use [1000]
    --ground_impedence_model
                         Impedence model to use. Currently only "rigid" is
                         supported. [rigid]
    --Lamb_wave_BC       Use admittance = -1/2*dln(rho)/dz
    --write_atm_profile  Output atmospheric profile to atm_profile.nm
    --use_attn_file      File name containing attenuation, to override default
                         Sutherland/Bass [n/a]. Columns are
                         Height(km) Attenuation(np/m)
    --use_zero_attenuation
                         Set attenuation to zero.
    --wvnum_filter       Use wavenumber filter by phase speed. Requires --c_min
```

```
                        and --c_max
      --c_min           Minimum phase speed to keep
      --c_max           Maximum phase speed to keep
  --propagation         Calculate propagated waveform using dispersion results
    --input_dispersion_file
                        File to read dispersion results from [required]
    --output_waveform_file
                        File to write resultant waveform to [required]
    --source            Source type. Options include:
                        {impulse,pulse1,pulse2,spectrum,waveform} [impulse]
      --source_file     File containing the source spectrum or waveform, if
                        applicable
      --f_center        Center frequency for pulse1 or pulse2 options. Must be
                        <= f_max/5 [f_max/5]
    --nfft              Number of FFT points [4*f_max/f_step]
    --max_celerity      Maximum celerity for calculation [340.0]
    --receiver          Receiver type {single,multiple} [single]
      --range_km        Propagation range in km for a single receiver
      --start_range_km  Starting propagation range in km for multiple receivers
      --end_range_km    Ending propagation range in km for multiple receivers
      --range_step_km   Propagation range step in km for multiple receivers



OUTPUT Files: Format description (column order):
  <dispersion file>       Contains one line per frequency with entries:
                          freq, (# of modes), rho(z_src),
                          followed for each mode 'i' by quadruples:
                          real(k(i)), imag(k(i)), Mode(i)(z_src), Mode(i)(z_rcv)
  <waveform file>         t  P  (single receiver)
                          r  t  P  (multiple receivers)
Examples (run from 'samples' directory):
    ../bin/ModBB --dispersion --dispersion_file myDispersionFile.dat \
         --atmosfile NCPA_canonical_profile_zuvwtdp.dat --azimuth 90 --f_min \
         0.001953125 --f_step 0.001953125 --f_max 0.5 --method modess
 --or--
    ../bin/ModBB --dispersion --dispersion_file myDispersionFile.dat \
         --atmosfile NCPA_canonical_profile_zuvwtdp.dat --azimuth 90 --f_min \
         0.001953125 --f_step 0.001953125 --f_max 0.5 --method wmod

 --then--
    ../bin/ModBB --propagation --input_dispersion_file myDispersionFile.dat \
         --range_km 240 --output_waveform_file mywavf.dat --receiver single \
         --source impulse

    ../bin/ModBB --propagation --input_dispersion_file myDispersionFile.dat \
         --output_waveform_file mywavf.dat --receiver multiple \
         --start_range_km 240 --end_range_km 300 --range_step_km 20 --source \
         waveform --source_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.

If either source or reveiver is on the ground the minimum required phase speed is estimated using the WKB approximation (see Fig. 2). The option `--turnoff_WKB` forces the dispersion file to be written using the minimum of the effective sound speed as the minimum modal phase speed, rather than using the WKB approximation to estimate the lowest relevant phase speed. The option `--use_zero_attn` sets the attenuation to zero. In the current release this is done by setting the imaginary part of the wave numbers to zero, rather than by setting the attenuation to zero prior to writing the dispersion file. Since the imaginary part of the wave number is extimated in first order perturbation theory in Modess and Wmod setting the imaginary part to zero is equivalent to having set the attenuation coefficient to zero.

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.

6.3 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 --dispersion --dispersion-file myDispersionFile.dat \
```

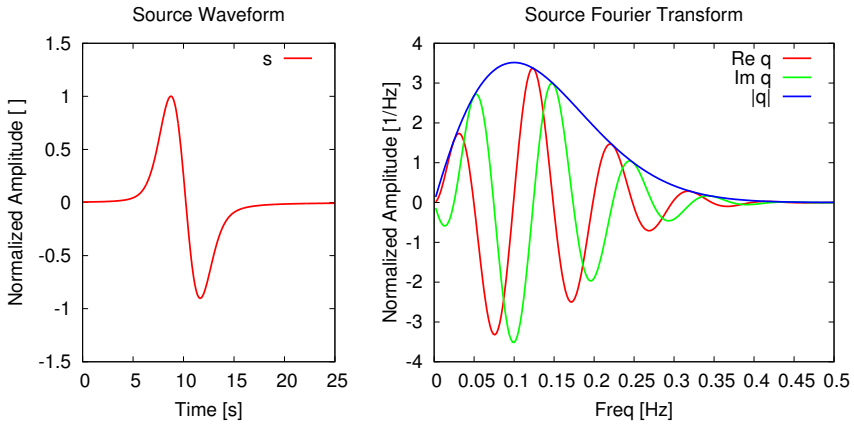


Figure 11: The built-in source waveform and Fourier transform with peak at 0.1 Hz.

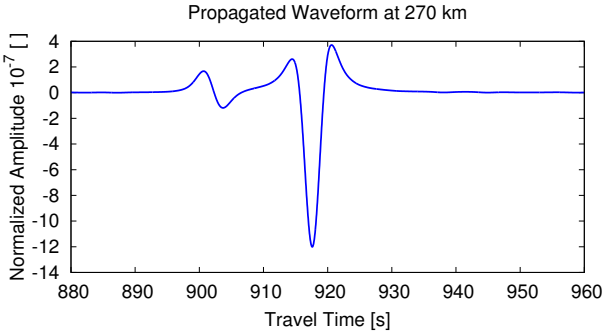


Figure 12: Eastward propagation in the NCPA toy atmosphere. Propagated pulse to 270 km vs time.

```
--atmosfile NCPA_canonical_profile_zuvwtdp.dat \
--azimuth 90 --method modess \
--f_min 0.002 --f_step 0.002 --f_max 0.5
```

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 11. The propagated waveform is plotted in Figure 12.

```
../bin/ModBB --propagation --input_dispersion_file myDispersionFile.dat \
             --range_km 270 --output_waveform_file mywavf.dat \
             --source pulse1 --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 13.

```
../bin/ModBB --propagation --input_dispersion_file myDispersionFile.dat \
             --receiver multiple --output_waveform_file mywavf_grid.dat \
```

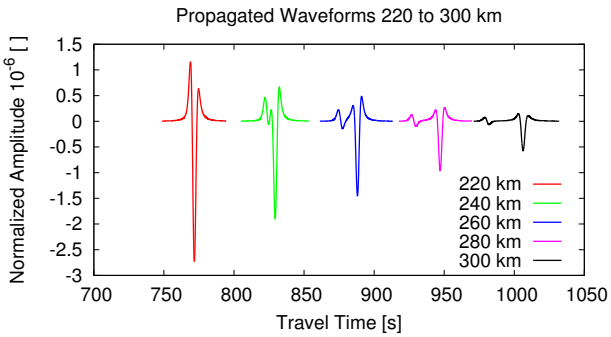


Figure 13: Eastward propagation in the NCPA toy atmosphere. Propagated pulse from 220 to 300 km vs time in increments of 20 km.

```
--start_range_km 220 --range_step_km 20 --end_range_km 300 \
--source pulse1 --max_celerity 320
```

7 ePape - Effective Sound Speed Padé Parabolic Equation Code

ePape implements a "Parabolic Equation" (PE)[4] one-way algorithm, originally developed for ocean acoustic applications, with the influence of the winds approximated using the effective sound speed approximation. As such, propagation is restricted to a single range/altitude plane specified by an azimuth. The influence of cross winds is not modeled. The functions of differential operators that arise are approximated using a Padé expansion whose order can be specified. The differential operators are discretized using a finite difference scheme. The Padé approximation provides numerical accuracy at high elevation angles; however, this can not be considered a high angle PE because the effective sound speed approximation limits the elevation angle, as well as the allowable wind speeds. Attenuation by the atmosphere is implemented via the addition of an imaginary part to the wave number. Both stratified and range dependent atmospheres are supported. Ground impedance boundary conditions are also supported; however, no impedance models have been included yet. Propagation over ground topography is not supported in this release.

7.1 Mathematical Background

The PE algorithms operate in the frequency domain and are thus approximate solutions to the convective Helmholtz equation. It's natural to work in cylindrical coordinates range r, azimuth θ, and altitude z. **ePape** uses the effective sound speed approximation[15] in which the thermodynamic sound speed, c, is replace by $c_{\text{eff}}(\theta, z) = c(z) + u(\theta, z)$, where u is the horizontal component of the wind along a given azimuth θ. In this approximation the influence of cross winds is not modeled, causing the propagation to be restricted to a single range/altitude plane, an $r-z$ plane. The plane is fixed by setting an azimuth, θ. In the effective sound speed approximation, the solution depends on azimuth only through the effective sound speed. Similarly, the locally stratified approximation is used, in which c and u can depend on range r, but the r dependence is very slowly varying on the scale of acoustic wavelengths so that changes in c and u can be treated by using the value of c_{eff} at the range step being considered, ignoring the r dependence, which is to say derivatives with respect to r, of c_{eff} at each range step.

Introduce the effective wave number (we will drop the explicit dependence on azimuth)

$$k(z) = \frac{\omega}{c_{\text{eff}}(z)}.$$

If $P(r, z)$ is the acoustic pressure, then

$$\Big(\frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \rho_0(z)\frac{\partial}{\partial z}\frac{1}{\rho_0(z)}\frac{\partial}{\partial z} + k(z)^2\Big)P(r, z) = 0 \tag{6}$$

where ρ_0 is the mean atmospheric density. The acoustic pressure P is scaled by both the $r^{-\frac{1}{2}}$ geometrical spreading factor and the $\sqrt{\rho_0}$ density factor,

$$P(r, z) = \sqrt{\frac{\rho_0(z)}{r}}\tilde{P}(r, z). \tag{7}$$

Let k_0 be a reference wave number, typically the wave number on the ground, and introduce the far field altitude operator

$$Q = \frac{1}{k_0^2}\Big(\frac{d^2}{dz^2} + \tilde{k}(z)^2 - k_0^2\Big),$$

with

$$\tilde{k}(z) = \sqrt{k(z)^2 - \frac{3}{4}\Big(\frac{\rho_0'(z)}{\rho_0(z)}\Big)^2 + \frac{1}{2}\frac{\rho_0''(z)}{\rho_0(z)}}. \tag{8}$$

Then, in the far field, ignoring terms of order $\frac{1}{r^2}$,

$$\Big(\frac{\partial^2}{\partial r^2} + k_0^2(1 + Q)\Big)\tilde{P}(r, z) = 0. \tag{9}$$

Formally factoring the differential operator

$$\frac{\partial^2}{\partial r^2} + k_0^2(1+Q) = \Big(\frac{\partial}{\partial r} + ik_0\sqrt{1+Q}\Big)\Big(\frac{\partial}{\partial r} - ik_0\sqrt{1+Q}\Big)$$

we see that solutions of

$$\Big(\frac{\partial}{\partial r} \pm ik_0\sqrt{1+Q}\Big)\tilde{P}(r,z) = 0$$

are one way solutions of Eq. 9 in the sense that they represent waves traveling in a single direction, ie. outward in the $-$ case and inward in the $+$ case. We will restrict to the outward going case.

Dividing out a carrier wave as well,

$$P = \frac{1}{\sqrt{r}}\tilde{P} = \frac{e^{ik_0 r}}{\sqrt{r}}p,$$

one finds that

$$\frac{\partial p}{\partial r} = ik_0\big(\sqrt{1+Q} - 1\big)p. \tag{10}$$

Here p is thought of as a vector valued function of r indexed by the altitude z. Note that Eq. 10 is the origin of the name "Parabolic Equation Method". In the earliest application of this approach $\sqrt{1+Q}-1$ was approximated by $\frac{\partial^2}{\partial z^2}$, resulting in a parabolic equation.

Eq. 10 has a formal solution,

$$p(r) = e^{ik_0(\sqrt{1+Q}-1)(r-r_0)}p(r_0)$$

where $r_0 < r$. One thinks of $p(r)$ as having been propagated out in range from r_0 to r. The PE algorithm is based on the formula (Trotter product formula [13])

$$p(r) = \lim_{N\to\infty}\Big(e^{ik_0(\sqrt{1+Q}-1)\frac{r-r_0}{N}}\Big)^N p(r_0) \tag{11}$$

so that letting

$$\Delta r = \frac{r - r_0}{N}$$

and q

$$r_j = r_0 + j\Delta r$$

one has the approximation

$$p(r_j) = e^{i\delta(\sqrt{1+Q}-1)}p(r_{j-1}) \tag{12}$$

with $\delta = k_0\Delta r$. This allows us to march out from r_0 to r if an efficient approximation for $e^{i\delta(\sqrt{1+Q}-1)}$ can be found. The atmospheric profile can be changed as required. Generally, r_0 is some small reference range and the starting pressure field, $p_0 = p(r_0)$, is called the starter. The starter is usually chosen in such a way as to mimic the signal at range r_0 from a point source. In this release, a point source starter has been implemented.

At high enough frequencies the ground can be treated as an impedance plane. This approximation ignores any signal propagation in the substrata and is implemented by imposing the boundary condition

$$\frac{\partial P(r,z)}{\partial z}\Big|_{z=0} = -\frac{i\omega\rho_0(0)}{Z(\omega)}P(r,0) \tag{13}$$

where $Z(\omega)$ is the effective ground impedance. This boundary condition influences the operator Q. Note that the density scaling effects the boundary condition. One finds that

$$\frac{\partial \tilde{P}(r,z)}{\partial z}\Big|_{z=0} = -\Big(\frac{i\omega\rho_0(0)}{Z(\omega)} + \frac{\rho_0'(0)}{2\rho_0(0)}\Big)P(r,0).$$

The additional density dependent terms provide the Lamb wave but are of the same order as the additional density dependent terms in $\tilde{k}$ and are usually ignored for frequencies above about 0.08 Hz where buoyancy is not significant.

The ground impedance is to be considered user input, similarly to the atmospheric profiles. It is generally a complex number. As $Z(\omega) \to \infty$ this becomes the rigid ground condition. This is expected as $\omega \to 0$. In common practice the ground impedance is set to zero for purposes of infrasound modeling, however, the low frequency behavior of the ground interaction is not well understood.

To implement Eq.12, regardless of Q, an effective approximation must be found for the action of the operator

$$\mathcal{F}(\delta, Q) = e^{i\delta(\sqrt{1+Q}-1)} \tag{14}$$

on functions of altitude. To this end a Padé approximant will be used. The Padé approximant is a best fit rational function of a given order. To approximate $\mathcal{F}(\delta, x)$, one defines

$$R(x) = \frac{a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n}{1 + b_1 x + b_2 x^2 + \cdots + b_m x^m} \tag{15}$$

and then chooses the coefficients a_j and b_k so that the Taylor expansions of R and $\mathcal{F}$ coincide up to order $N = n + m$. Explicitly,

$$\frac{d^j R}{dx^j}\Big|_{x=0} = \frac{d^j \mathcal{F}}{dx^j}\Big|_{x=0}$$

for $j = 0, 1, \ldots, N$. That gives $N + 1$ equations for as many unknown coefficients. If one restricts $m \geq n$ then $R(x)$ remains bounded for large x. One then approximates

$$\mathcal{F}(\delta, x) \approx R(x).$$

Approximation by a rational function is preferable to approximation by a polynomial, *ie* by a Taylor expansion, because polynomials always increase without bound for large argument. If we insist, as we will, that $m \geq n$ then the approximating function, $R(x)$, is bounded.

To determine the coefficients a_j and b_j we will need the Taylor coefficients, c_j, of $\mathcal{F}$,

$$\mathcal{F}(\delta, \xi) = \sum_{n=0}^{\infty} c_n(\delta)\xi^n$$

Thompson[11] has derived a recursion relation for the expansion coefficients c_n,

$$c_{n+1} = -\frac{2n-1}{2(n+1)} c_n - \frac{\delta^2}{4n(n+1)} c_{n-1}.$$

This makes the computation of the expansion coefficients very efficient. Note that $c_0 = 1$ and $c_1 = i\frac{\delta}{2}$. Then the a_j and b_j are determined by the condition that that the Taylor expansion of the function

$$\mathcal{F}(\delta, x) - R(x) = \sum_{j=0}^{\infty} c_j(\delta) x^j - \frac{\sum_{l=0}^{n} a_l x^l}{1 + \sum_{k=1}^{m} b_k x^k}$$

begin at order x^{m+n+1}. This is equivalent to the coefficients in the power series

$$\Big(\sum_{j=0}^{\infty} c_j(\delta) x^j\Big)\Big(1 + \sum_{k=1}^{m} b_k x^k\Big) - \sum_{l=0}^{n} a_l x^l$$

be zero for powers less than or equal to $m + n$. This recursion leads to a sequence of equations. At index 0,

$$a_0 = c_0,$$

for $i = 1, 2, \ldots, n$,

$$\sum_{j=0}^{i-1} c_j b_{i-j} - a_i = -c_i,$$

for $i = n+1, n+2, \ldots, m$,

$$\sum_{j=0}^{i-1} c_j b_{i-j} = -c_i,$$

and, for $i = m+1, m+2, \ldots, N$,

$$\sum_{j=i-m}^{i-1} c_j b_{i-j} = -c_i.$$

Since the c_i are known, this gives a system of linear equations for the coefficients a_j and b_j. This linear system can be readily solved for the coefficients a_j and b_j. The approximation for Eq. 14 is given by

$$R(Q) = \frac{a_0 + a_1 Q + a_2 Q^2 + \cdots + a_n Q^n}{1 + b_1 Q + b_2 Q^2 + \cdots + b_m Q^m}.$$

In **ePape** m is fixed at $n+1$. The user is given an option to choose m.

Atmospheric attenuation is taken into account by adding an imaginary part, α_{att}, known as the attenuation coefficient, to the wave number k. Standard models for the attenuation coefficient are to be found in the literature and generally attributed to Sutherland and Bass[14]. An extension of the Sutherland-Bass model due to Godin takes into account the influence of the winds and can be used as well[6].

An artificial attenuation layer is introduced at the top of the computational domain to prevent spurious reflections. The absorbing layer comprises an additional imaginary part $\alpha_A(z)$ of $k(z)$ where α_A is zero as z approaches some fixed altitude z_T, then increases slowly enough that there are no significant signal returns from the layer, but becomes large enough to attenuate all signals. The form used in **ePape** is

$$\alpha_A(z) = \mu e^{\frac{z - z_T}{\lambda_T}}. \tag{16}$$

Currently, we have set $\mu = 0.1$, $z_T = z_{max} - \lambda_T$, and $\lambda_T = min(\frac{2\pi}{k_0}, 5km)$.

Rather than introducing a new symbol we will define

$$k(z) = \sqrt{\frac{\omega^2}{c_{\text{eff}}(z)^2} - \frac{3}{4}\big(\frac{\rho_0'(z)}{\rho_0(z)}\big)^2 + \frac{1}{2}\frac{\rho_0''(z)}{\rho_0(z)}} + i\alpha_{\text{att}}(z) + i\alpha_A(z) \tag{17}$$

To evaluate the action of Q^n on vertical vectors p the z axis is replaced with a uniform grid with spacing h and then, writing

$$p_j = p(jh)$$

we replace $\frac{d^2}{dz^2}$ with the centered finite difference operator,

$$\frac{d^2 p}{dz^2}\Big|_{z=jh} \approx \frac{p_{j+1} - 2p_j + p_{j-1}}{h^2}$$

leading to the tri-diagonal matrix

$$\Big(\frac{d^2}{dz^2}\Big)_{j,k} \approx \frac{\delta_{j,k+1} - 2\delta_{j,k} + \delta_{j,k-1}}{h^2};$$

This form is fine for $j > 1$. For $j = 1$ the boundary condition Eq. 13 must be implemented. Write it in the form

$$\frac{\partial \tilde{P}(r,z)}{\partial z}\Big|_{z=0} = -\mathcal{A}P(r,0)$$

and implement it to first order by introducing a $j = 0$ term such that

$$\frac{p_1 - p_0}{h} \approx -\mathcal{A}p_1$$

from which it follows that $p_0 = (1 + h\mathcal{A})p_1$ and then

$$\frac{p_2 - 2p_1 + p_0}{h^2} \approx \frac{p_2 - (1 - h\mathcal{A})p_1}{h^2}.$$

Thus, the centered finite difference version of $\frac{d^2}{dz^2}$ incorporating the impedance boundary condition is

$$D = \frac{1}{h^2}\begin{pmatrix} -1+h\mathcal{A} & 1 & & & & & \\ 1 & -2 & 1 & & & & \\ & 1 & -2 & 1 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & 1 & -2 & 1 & \\ & & & & 1 & -2 \end{pmatrix}. \tag{18}$$

If we introduce the diagonal matrix K with matrix elements

$$K_{jk} = \delta_{j,k}\Big(k(jh)^2 - k_0^2)$$

then the centered finite difference version of Q is

$$Q = \frac{1}{k_0^2}(D + K) \tag{19}$$

Note that the rigid ground condition is obtained with $\mathcal{A} = 0$ and the pressure release condition with $p_0 = 0$ leading to $\mathcal{A} = -\frac{1}{h}$.

We have implemented a version of the Collins self-starter[2, 3]. The scaled pressure at range r_0 from a point source at altitude $z_0 = hj_0$ and $r = 0$ is approximated using

$$p(r, z) = \frac{1}{k_0}(1 + Q)^{-\frac{1}{2}} e^{ik_0 r(1+Q)^{\frac{1}{2}}} \delta j, j_0.$$

A Gaussian starter is included in **ePape**, but is not recommended as it imperfectly estimates a point source and introduces error relative to the modal models.

Ground topography will be modeled by extending the computational domain downwards and setting the density and soundspeed underground to approximate a rigid ground surface. The region underground will be called the basement. The sound speed in the basement will be assumed to be infinite and the densty will be denoted by ρ_b. Let $S(r)$ is the altitude of the ground surface relative to some reference altitude or depth. In practice it can be relative to the bottom of the computational domain, or basement, but will be considered sufficiently distant from the bottom that the depth of the basement may be considered infinite. For the wavenumber $k(z)$ we extend

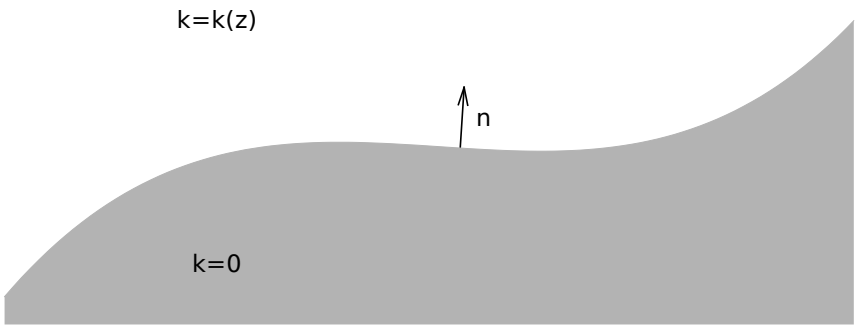
$$k(z) = \begin{cases} 0 & \text{if } z < S(r) \\ \frac{\omega^2}{c_{\text{eff}}(z - S(r))^2} + i\alpha_{\text{att}}(z - S(r)) + i\alpha_A(z - S(r)) & \text{if } S(r) < z \end{cases} \tag{20}$$

and for the density

$$\rho(z) = \begin{cases} \rho_b & \text{if } z < S(r) \\ \rho_0(z) & \text{if } S(r) < z. \end{cases}$$

The ground surface is the locus of points (r, z) satisfying

$$z - S(r) = 0.$$



Ground heights specified in the input atmospheric profiles using the Z0 scalar value are used to construct a cubic spline, with repeated endpoints to allow for slight overshoots. This spline returns ground heights interpolated between profiles, as well as their horizonal first and second derivatives.

Eq. 6 is valid for $z > S(r)$ while for $z < 0$

$$\Big(\frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{\partial^2}{\partial z^2}\Big)P(r,z) = 0.$$

These are augmented by the condition that pressure and the normal component of the velocity (or pressure gradient) be continuous at the air/ground interface. Let

$$\mathbf{r}_\pm = \begin{pmatrix} r \\ S(r) \end{pmatrix} \pm 0^+ \hat{\mathbf{n}}(r)$$

represent the limiting position on either side of the normal to the gound surface and note that the upward pointing unit normal at a point on the ground surface is given by

$$\hat{\mathbf{n}}(r) = \begin{pmatrix} n_r(r) \\ n_z(r) \end{pmatrix} = \frac{1}{\sqrt{1 + S'(r)^2}} \begin{pmatrix} -S'(r) \\ 1 \end{pmatrix}.$$

Then

$$P(\mathbf{r}_+) = P(\mathbf{r}_-) \tag{21}$$

and

$$\frac{1}{\rho_0\big(S(r)\big)}\hat{\mathbf{n}}(r)\cdot\nabla P|_{\mathbf{r}_+} = \frac{1}{\rho_b}\hat{\mathbf{n}}(r)\cdot\nabla P|_{\mathbf{r}_-}. \tag{22}$$

Note that

$$\hat{\mathbf{n}}(r)\cdot\nabla = \frac{1}{\sqrt{1 + S'(r)^2}}\Big(-S'(r)\frac{\partial}{\partial r} + \frac{\partial}{\partial z}\Big).$$

The pressure fields can be scaled as before. For $z > S(r)$ we define

$$P(r,z) = \sqrt{\frac{\rho_0(z)}{r}}\tilde{P}(r,z) = \sqrt{\frac{\rho_0(z)}{r}}e^{ik_0 r}p(r,z).$$

For $z < S(r)$ the density is constant, so the density scaling doesn't matter, we will do it just to keep the units the same and to make the density gradient zero, setting

$$P(r,z) = \sqrt{\frac{\rho_0\big(S(r)\big)}{r}}\tilde{P}(r,z) = \sqrt{\frac{\rho_0\big(S(r)\big)}{r}}e^{ik_0 r}p(r,z).$$

Using Eq. 10 one finds

$$\frac{\partial p}{\partial r} = ik_0\big(\sqrt{1+Q}-1\big)p$$

for $z > S(r)$ and

$$\frac{\partial p}{\partial r} = ik_0\big(\sqrt{1+Q_0}-1\big)p$$

for $z < S(r)$ where

$$Q_0 = \frac{1}{k_0^2}\Big(\frac{d^2}{dz^2} - k_0^2\Big).$$

The pressure continuity condition, Eq.21, becomes

$$p(\mathbf{r}_+) = p(\mathbf{r}_-).$$

Using

$$\begin{aligned}\frac{\partial}{\partial r}\sqrt{\frac{1}{r}}\; e^{ik_0 r}p(r,z) &= \sqrt{\frac{1}{r}}\; e^{ik_0 r}\Big(-\frac{1}{2r} + ik_0 + \frac{\partial}{\partial r}\Big)p(r,z)\\ &= \sqrt{\frac{1}{r}}\; e^{ik_0 r}\Big(-\frac{1}{2r} + ik_0\sqrt{1+Q}\Big)p(r,z),\end{aligned}$$

with Q replaced by Q_0 for $z < S(r)$, the condition Eq.22 becomes

$$\frac{1}{\rho_0(S(r))}\Big[ik_0 S'(r)\sqrt{1+Q} - \frac{\partial}{\partial z} - \frac{1}{2}\frac{\rho_0'(z)}{\rho_0(z)}\Big]p\Big|_{\mathbf{r}_+} = \frac{1}{\rho_b}\Big[ik_0 S'(r)\sqrt{1+Q_0} - \frac{\partial}{\partial z}\Big]p\Big|_{\mathbf{r}_-} \tag{23}$$

where the far field approximation has been used and terms proportional to $\frac{1}{r}$ have been dropped.

Begin with the case in which the interface is flat. Here $S(r)$ is a constant to be denoted by z_S. Let J be the smallest integer with $Jh > z_S$ and we will assume that $J - 1 < z_S$. Introduce the notation

$$\begin{aligned} p_S &= p(r, z_S)\\ J_S &= \frac{z_S}{h}\\ \rho_a &= \rho_0(z_S)\\ \Gamma &= \frac{1}{2}\frac{\rho_0'(z_S)}{\rho_0(z_S)}.\end{aligned} \tag{24}$$

We then approximate

$$\frac{\partial p}{\partial z}\big|_{\mathbf{r}_+} \approx \frac{p_J - p_S}{(J - J_S)h}$$

and

$$\frac{\partial p}{\partial z}\big|_{\mathbf{r}_-} \approx \frac{p_S - p_{J-1}}{(J_S - J + 1)h}.$$

The continuity of p implies that p_S is the pressure at the interface whether measured from above or below. In the case of a flat ground surface $S'(r) = 0$ and Eq.23 can be approximated as

$$\frac{1}{\rho_a}\Big(\frac{p_J - p_S}{(J - J_S)h} + \Gamma p_S\Big) \approx \frac{1}{\rho_b}\frac{p_S - p_{J-1}}{(J_S - J + 1)h}.$$

Solving for p_S one finds

$$\begin{aligned} p_S &= \frac{\frac{1}{\rho_b}\frac{p_{J-1}}{J_S - J + 1} + \frac{1}{\rho_a}\frac{p_J}{J - J_S}}{\frac{1}{\rho_b}\frac{1}{J_S - J + 1} + \frac{1}{\rho_a}\frac{1}{J - J_S} - h\Gamma}\\ &= \mathcal{A}p_J + \mathcal{B}p_{J-1},\end{aligned}$$

with

$$\mathcal{A} = \frac{\frac{1}{\rho_a}\frac{1}{J - J_S}}{\frac{1}{\rho_b}\frac{1}{J_S - J + 1} + \frac{1}{\rho_a}\frac{1}{J - J_S} - h\Gamma}$$

and

$$\mathcal{B} = \frac{\frac{1}{\rho_b}\frac{1}{J_S-J+1}}{\frac{1}{\rho_b}\frac{1}{J_S-J+1} + \frac{1}{\rho_a}\frac{1}{J-J_S} - h\Gamma}.$$

The finite difference approximation to the second derivatives just above and below the interface are then approximated by

$$\frac{\partial^2 p}{\partial z^2}\Big|_{\mathbf{r}_+} \approx \frac{p_{J+1} - 2p_J + p_S}{h^2(J - J_S)} \approx \frac{p_{J+1} + (-2 + \mathcal{A})p_J + \mathcal{B}p_{J-1}}{h^2(J - J_S)}$$

and

$$\frac{\partial^2 p}{\partial z^2}\Big|_{\mathbf{r}_-} \approx \frac{p_S - 2p_{J-1} + p_{J-2}}{h^2(J_S - J + 1)} \approx \frac{\mathcal{A}p_J + (-2 + \mathcal{B})p_{J-1} + p_{J-2}}{h^2(J_S - J + 1)}.$$

Note that these forms couple the basement and the atmosphere.

$$D = \frac{1}{h^2}\begin{pmatrix} \ddots & \ddots & \ddots & & & & & & \\ & 1 & -2 & 1 & & & & & \\ & & 1 & -2 & 1 & & & & \\ & & & \alpha & \beta & \gamma & & & \\ & & & & a & b & c & & \\ & & & & & 1 & -2 & 1 & \\ & & & & & & 1 & -2 & 1 \\ & & & & & & & \ddots & \ddots & \ddots \end{pmatrix} \tag{25}$$

where β is the $J-1, J-1$ entry and b the J, J entry and

$$\begin{aligned} \alpha &= \frac{1}{J_S - J + 1} \\ \beta &= \frac{-2 + \mathcal{B}}{J_S - J + 1} \\ \gamma &= \frac{\mathcal{A}}{J_S - J + 1} \\ a &= \frac{\mathcal{B}}{J - J_S} \\ b &= \frac{-2 + \mathcal{A}}{J - J_S} \\ c &= \frac{1}{J - J_S} \end{aligned}.$$

Using the specification Eq.20 for $k(z)$, the specification Eq.19 for the finite difference version of the operator Q, which, in this case, includes the interface condition, remains valid. The boundary conditions at the upper and lower limits of the domain have not been specified, but they have no effect.

As in the flat ground case, we look for a finite difference approximation to the operator Q that includes the interface condition. Assuming such an approximation has been found at a given range r, let J be the smallest integer with $Jh > S(r) = z_S$. In addition to the notation introduced in Eq.24, introduce the notation

$$\mathbf{M} = ik_0 S'(r)\sqrt{1 + Q}.$$

We will see that the resulting expression for $\mathbf{M}$ changes with each range step. Here we will interprate Q as the finite difference approximation for the current range step.

As before, approximate

$$\frac{\partial p}{\partial z}\Big|_{\mathbf{r}_+} \approx \frac{p_J - p_S}{(J - J_S)h},$$

and, if $(J-1)h \neq S(r)$ (the possibility that $(J-1)h = S(r)$ will have to be considered),

$$\frac{\partial p}{\partial z}\Big|_{\mathbf{r}_-} \approx \frac{p_S - p_{J-1}}{(J_S - J + 1)h}.$$

As before, the continuity of p implies that p_S is the pressure at the interface whether from above or below. Then Eq.23 becomes

$$\frac{1}{\rho_a}\Big(\frac{p_J - p_S}{(J-J_S)h} + \Gamma p_S - (\mathbf{M}p)_J\Big) \approx \frac{1}{\rho_b}\Big(\frac{p_S - p_{J-1}}{(J_S - J + 1)h} - (\mathbf{M}p)_{J-1}\Big).$$

Solving for p_S one finds

$$\begin{aligned} p_S &= \frac{\frac{1}{\rho_b}\frac{p_{J-1}}{J_S-J+1} + \frac{1}{\rho_a}\frac{p_J}{J-J_S} - h\Big((\mathbf{M}p)_J - (\mathbf{M}p)_{J-1}\Big)}{\frac{1}{\rho_b}\frac{1}{J_S-J+1} + \frac{1}{\rho_a}\frac{1}{J-J_S} - h\Gamma} \\ &= \mathcal{A}p_J + \mathcal{B}p_{J-1} - \sum_k \mathcal{M}_{J,k}p_k + \sum_k \mathcal{M}_{J-1,k}p_k \end{aligned}$$

with the matrix $\mathcal{M}$ defined by

$$\mathcal{M} = \frac{h}{\frac{1}{\rho_b}\frac{1}{J_S-J+1} + \frac{1}{\rho_a}\frac{1}{J-J_S} - h\Gamma} M.$$

The expression for p_S can now be substituted into the finite difference approximation for $\frac{\partial^2}{\partial z^2}$ giving

$$\frac{\partial^2 p}{\partial z^2}\Big|_{\mathbf{r}_+} \approx \frac{p_{J+1} + (-2+\mathcal{A})p_J + \mathcal{B}p_{J-1} - \sum_k (\mathcal{M}_{J,k} - \mathcal{M}_{J-1,k})p_k}{h^2(J-J_S)}$$

and

$$\frac{\partial^2 p}{\partial z^2}\Big|_{\mathbf{r}_-} \approx \frac{\mathcal{A}p_J + (\mathcal{B}-2)p_{J-1} + p_{J-2} - \sum_k (\mathcal{M}_{J,k} - \mathcal{M}_{J-1,k})p_k}{h^2(J_S-J+1)}.$$

These get substituted into the matrix for D. Eq. 25 remains unchanged except that the vectors

$$-\frac{1}{h^2(J-J_S)}(\mathcal{M}_{J,k} - \mathcal{M}_{J-1,k})$$

and

$$-\frac{1}{h^2(J_S-J+1)}(\mathcal{M}_{J,k} - \mathcal{M}_{J-1,k})$$

get added to the J^{th} and $(J-1)^{\text{st}}$ rows, respectively.

To proceed further we need an approximation for $\sqrt{1+Q}$. The Taylor expansion gives

$$\begin{aligned} \sqrt{1+Q} &= 1 + \frac{1}{2}Q - \frac{1}{8}Q^2 + \frac{1}{16}Q^3 - \frac{5}{128}Q^4 + \cdots + (-1)^n \frac{1\cdot 3\cdot \ldots \cdot (2n-3)}{n!2^n}Q^n + \cdots \\ &= 1 + c_1 Q + c_2 Q^2 + \ldots \end{aligned}$$

The general n, m^{th} order rational approximation is given by

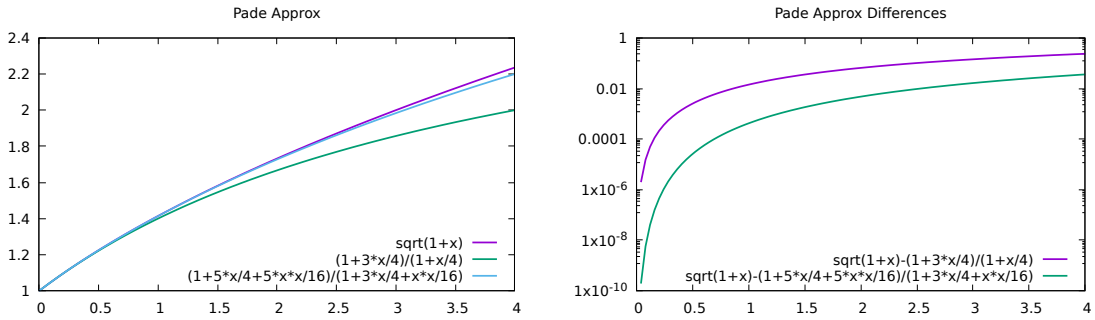
$$\sqrt{1+Q} \approx \frac{1 + a_1 Q + a_2 Q^2 + \cdots + a_n Q^n}{1 + b_1 Q + b_2 Q^2 + \cdots + b_m Q^m}$$

An order $1,1$ rational approximation gives

$$\sqrt{1+Q} \approx \frac{1 + \frac{3}{4}Q}{1 + \frac{1}{4}Q}$$

and an order $2,2$ gives

$$\sqrt{1+Q} \approx \frac{1 + \frac{5}{4}Q + \frac{5}{16}Q^2}{1 + \frac{3}{4}Q + \frac{1}{16}Q^2}$$



These approximations are shown in the figure and can be improved as needed. In the interval $(0, 1)$ the order $2, 2$ approximation is correct to a part in a thousand.

To use these rational approximations we need to manage the matrix division. Note that specific rows of $\mathbf{M}$ are required, in particular the J^{th} and $J + 1^{\text{st}}$ rows. In general, if $\mathbf{M}_K$ is the K^{th} row of $\mathbf{M}$, and if $\mathbf{e}^{(K)}$ is the vector that is zero except for the K^{th} entry, which is 1, then

$$\mathbf{M}_K = \mathbf{e}^{(K)\,\text{t}}\mathbf{M} = (\mathbf{M}^{\text{t}} e^{(K)})^{\text{t}}.$$

Thus, to obtain the K^{th} row, one solves the linear system

$$\mathbf{M}_K^{\text{t}} = \mathbf{M}^{\text{t}} e^{(K)}.$$

This means that all matrix divisions can be implemented using an LU decomposition or similar. For example, the 1,1 rational approximation gives

$$(1 + \frac{1}{4}Q^{\text{t}})\mathbf{M}_K^{\text{t}} = (1 + \frac{3}{4}Q^{\text{t}})e^{(K)}.$$

Currently, the 3,4 approximation is being used throughout.

7.2 Running ePape

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

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

on a command line. Generally, options are followed by values, either numbers, strings or filenames. Flags are not. Entering `ePape --help` sends the following help page to the screen:

```
By default the program computes both the 1D (at the ground) and 2D transfer
function and saves the data to 2 files:
    file tloss_1d.pe - transfer function at the ground
    file tloss_2d.pe - full 2-D transfer function field

The options below can be specified in a colon-separated file "epade_pe.param" or
at the command line. Command-line options override file options.

Options Control:
  --help                Prints help test
  --paramfile           Parameter file name [epade_pe.param]
  --printparams         Print parameter summary to screen
```

```
Atmosphere:
  --atmosfile            1-D atmospheric profile filename
  --atmosfile2d          2-D atmospheric summary filename (see manual)

Required Parameters:
  --freq                 Frequency of analysis (Hz)
  --starter              Starter type: one of { self, gaussian, user }
  --maxrange_km          Maximum range in km to use for modeling

Modes of Operation:
  --singleprop           Single azimuth propagation. Requires --azimuth
    --azimuth            Propagation azimuth ( degrees clockwise from north,
                         [0,360) )
  --multiprop            Multiple azimuth propagation. Requires --azimuth_start,
                         --azimuth_end, and --azimuth_step, disables
                         --atmosfile2d
    --azimuth_start      Starting azimuth, in degrees CW from North [0,360)
    --azimuth_end        Ending azimuth, in degrees CW from North [0,360)
    --azimuth_step       Azimuth step, in degrees CW from North [0,360)

Optional Parameters [default]:
  --atmosheaderfile      External header file, overrides internal header [None]
  --npade                Number of Pade coefficients to use [4]
  --dz_m                 Altitude resolution in meters [automatic]
  --maxheight_km         Maximum height of analysis in km [150.0, or max height
                         of atmosphere]
  --sourceheight_km      Source height in km [ground]
  --receiverheight_km    Receiver height in km [ground]
  --groundheight_km      Ground height in km [Z0 parameter in profile, or 0.0]
  --Nrng_steps           Number of range steps to use [decide internally]
  --ground_impedence_real
                         Real part of ground impedence [rigid ground]
  --ground_impedence_imag
                         Imaginary part of ground impedence [rigid ground]
  --starterfile          File name containing starter [n/a]. Columns are
                         Height(km) RealPart ImaginaryPart
  --attnfile             File name containing attenuation, to override default
                         Sutherland/Bass [n/a]. Columns are
                         Height(km) Attenuation(np/m)

Flags:
  --write_2d_tloss       Output 2-D transfer function to tloss_2d.pe
  --write_atm_profile    Output atmospheric profile to atm_profile.pe
  --write_starter        Output starter to starter.pe
  --write_topography     Output interpolated topography to topography.pe
  --lossless             Ignore atmospheric attenuation
  --topo                 Use topography. Requires presence of 'Z0' parameter in
                         atmospheric files



OUTPUT Files: Format description (column order):
  tloss_1d.pe:                 r (km), az (deg), TF (real), TF (imag)
  tloss_multiplot.pe           r (km), az (deg), TF (real), TF (imag)
```

```
  tloss_2d.pe:                 r, z, TF (real), TF (imag)
  atm_profile.pe:              z,u,v,w,t,d,p,c,c_eff
  starter.pe:                  z, starter (real), starter(imag)
  topography.pe:               az, r, z0

Examples (run from 'samples' directory):

    ../bin/ePape --singleprop --starter self --atmosfile \
         NCPA_canonical_profile_trimmed.dat --freq 0.1 --azimuth 90 \
         --maxrange_km 1000

    ../bin/ePape --singleprop --starter self --atmosfile2d \
         toy_profile_2d_summary.dat --freq 0.5 --azimuth 90 --maxrange_km 1800 \
         --write_2d_tloss

    ../bin/ePape --multiprop --starter self --atmosfile \
         NCPA_canonical_profile_trimmed.dat --freq 0.5 --azimuth_start 0 \
         --azimuth_end 360 --azimuth_step 2 --maxrange_km 1000

    ../bin/ePape --singleprop --topo --starter self --atmosfile2d \
         toy_profile_2d_summary_topo.dat --freq 0.5 --azimuth 90 --maxrange_km \
         1200 --write_2d_tloss --write_topography
```

The basic functionality of ePape is to produce a single frequency transfer function for signal propagation in the effective sound speed approximation. Explicitly, the long range propagated acoustic pressure amplitude at a single frequency produced by a unit point source at a reference range of 1 km is computed. The density scaled pressure, $\rho_0^{-\frac{1}{2}}P$, is written to files. The magnitude of the transfer function is referred to as the signal attenuation. To obtain the physical attenuation the output must be multiplied by $\rho_0^{-\frac{1}{2}}$; however, for purposes of visualization this is not advised since the physical acoustic pressure decreases dramatically with increasing altitude, making it difficult to visualize. The attenuation in dB relative to 1 is referred to as the transmission loss.

`Options Control:` Parameters and options can be sent to **ePape** either on the command line, or through a parameter file `epade_pe.param` by setting the option `--paramfile`. The option `--printparams` causes the current parameter set to be printed to the screen.

`Atmosphere:` The input of atmospheric profiles is currently through ascii files specified using `--atmosfile` or `--atmosfile2d`. The `--atmosfile` flag indicates that a stratified model atmosphere is being used, and a single file is specified. The `--atmosfile2d` flag indicates that a range dependent model atmosphere is being used and is specified through an atmospheric summary file. The summary file has two columns, range and filename; the files contain the atmospheric specifications valid at the indicated range. Binary format input is under development.

`Modes of Operation:` Using the `--singleprop` option the attenuation along a single azimuth is computed. By default, the attenuation along the ground is saved in `tloss_1d.pe` and the attenuation in the entire range/altitude plane is saved in `tloss_2d.pe`. With the `--multiprop` option the attenuation along a specified grid of azimuths is computed and stored in `tloss_multiplot.pe`. In `tloss_multiplot.pe` data is stored in line seperated blocks of constant range. `--multiprop` uses the functionality of `--singleprop` for each azimuth. For convenience, the column format in `tloss_1d.pe` is the same as `tloss_multiplot.pe` in that the azimuth is saved in the second column. Currently, only stratified atmospheres are supported for `--multiprop`. Support for range dependent atmospheres is under development. The `--topo` option causes **ePape** to introduce the high density basement with interface given by a cubic spline of from the `Z0` values in the atmospheric profile files. Note that `--topo` is intrinsically range dependent and requires a summary file referencing a directory of profile files. The flag `--write_topography` causes the interpolated ground

elevation to be written to a file.

`Optional Parameters:` Parameters can be changed from their default values. The `--npade` flag allows the user to choose the degree of the polynomial in the denomenator of the Padé approximant. The default is 4. Note that the numerator has degree one less than the denominator. The `dz_m` flag allows the user to choose the altitude resolution. The default is the closest value to 20 points per wavelength, computed on the ground, that respects a uniform grid. The user has control over the numnber of range steps through `--Nrng_steps`. The default range step is one wavelength, estimated using the nomunal sound speed 340 m/s. A ground impedence value can be input using the `--ground_impedence_real` and `--ground_impedence_imag` flags.

7.3 Running ePape: examples

The **ePape** help page ends with three examples designed to be run in the `samples` directory.

1) The first example is propagation a the stratified model atmosphere, the NCPA toy atmosphere, with source on the ground. The command line is

```
../bin/ePape --singleprop --starter self --atmosfile \
     NCPA_canonical_profile_trimmed.dat --freq 0.1 --azimuth 90 \
     --maxrange_km 1000
```

and the results are shown in Fig. 14. The 1d transmission loss for ground to ground propagation is shown on the left plotted as transmission loss versus range, and the 2d transmission loss in the full range-altitude plane is shown on the right plotted as altitude versus range with transmission loss indicated by color.

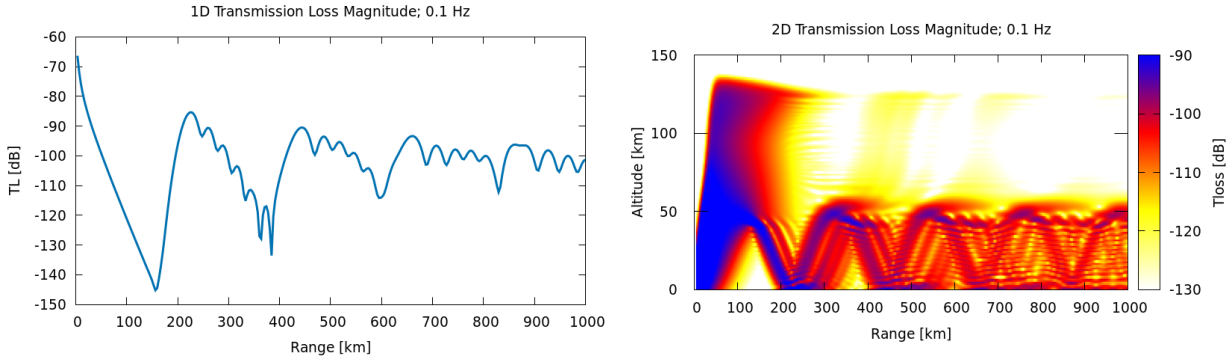


Figure 14: 1D and 2D transmission loss magnitudes at 0.1 Hz obtained with **ePape** for eastward ground-to-ground propagation in the NCPA toy atmosphere shown in Figure 1. Shown is the lossy transmission loss magnitude. The 1D transmission loss is shown in the left panel and the 2D transmission loss is shown in the right panel.

2) The second example shows how to input a range dependent atmosphere. The command line is

```
../bin/ePape --singleprop --starter self --atmosfile2d \
     toy_profile_2d_summary.dat --freq 0.5 --azimuth 90 --maxrange_km 1800 \
     --write_2d_tloss
```

The contents of the summary file `toy_profile_2d_summary.dat` are shown below.

```
0.0   toy_profiles/toy_soundspeed_jetstr0.dat
350.0  toy_profiles/toy_soundspeed_jetstr2.dat
500.0  toy_profiles/toy_soundspeed_jetstr4.dat
```

```
650.0  toy_profiles/toy_soundspeed_jetstr6.dat
800.0  toy_profiles/toy_soundspeed_jetstr8.dat
```

Each row contains a range (in km) and a file name. The range indicates where the model atmosphere given in the corresponding file is to commence. In this example, propagation is directly to the east. The model zonal winds and the corresponding effective sound speeds are shown in Fig. 15. In Fig. 16 the 1d transmission loss for ground to ground propagation is shown on the left plotted as transmission loss versus range, and the 2d transmission loss in the full range-altitude plane is shown on the right plotted as altitude versus range with transmission loss indicated by color.

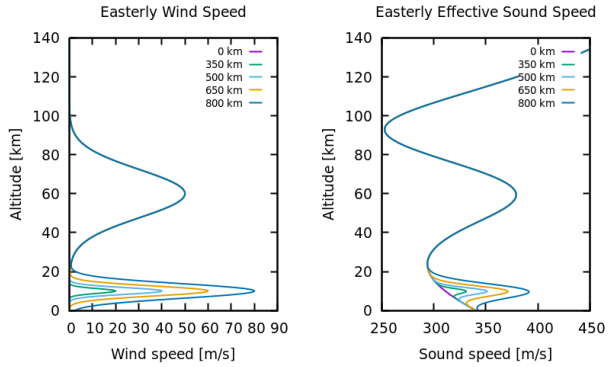


Figure 15: The zonal winds and associated effective sound speed profiles from the example range dependent atmosphere used in example 2.

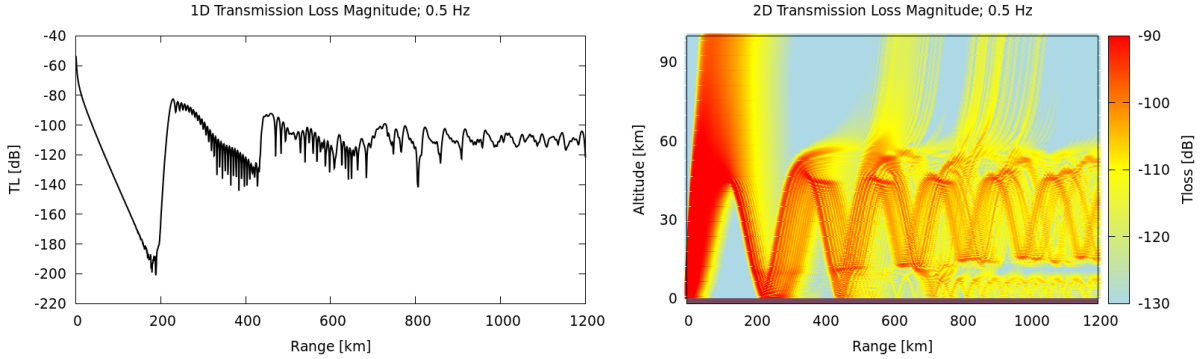


Figure 16: 1D and 2D transmission loss magnitudes at 0.5 Hz obtained with **ePape** for eastward ground-to-ground propagation in the range dependent atmosphere specified by the summary file `toy_profile_2d_summary.dat`. Shown is the lossy transmission loss magnitude. The 1d transmission loss on the left and the 2d on the right.

3) The third example shows how to use the `--multiplot` option for the NCPA toy atmosphere. The command line is

```
../bin/ePape --multiprop --starter self --atmosfile \
    NCPA_canonical_profile_trimmed.dat --freq 0.5 --azimuth_start 0 \
    --azimuth_end 360 --azimuth_step 2 --maxrange_km 1000
```

The lossless and lossy results are shown in Fig. 17.

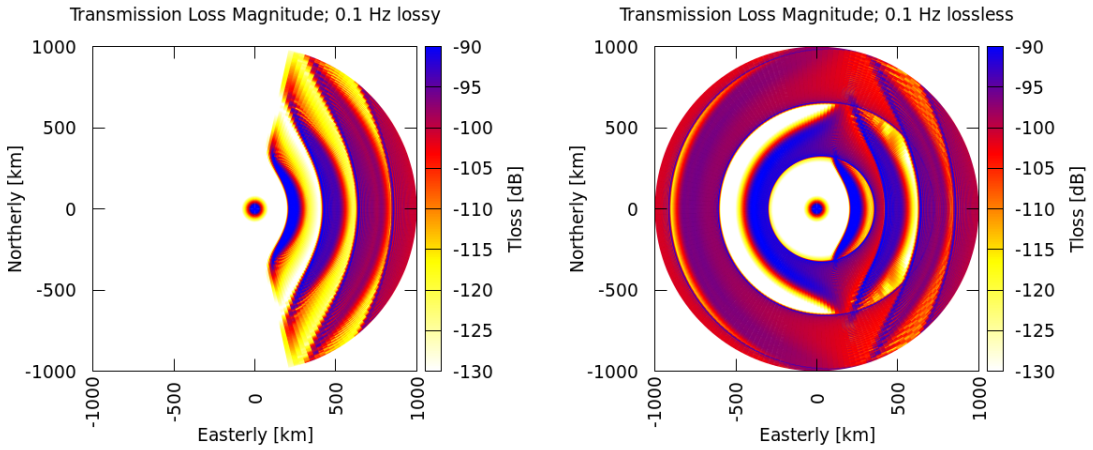


Figure 17: Multiplot option output. Shown on the left is the lossy transmission loss magnitude and on the right the lossless transmission loss magnitude.

4) The fourth example shows how to use the `--topo` option(s). Usage is similar to the use of **ePape** with range dependence. The command line is

```
../bin/ePape --singleprop --topo --starter self --atmosfile2d \
     toy_profile_2d_summary_topo.dat --freq 0.5 --azimuth 90 --maxrange_km \
     1200 --write_2d_tloss --write_topography
```

The profiles are the same as in 15 of example 2, except that there are non-zero values of `Z0` in some of the profile file headers.

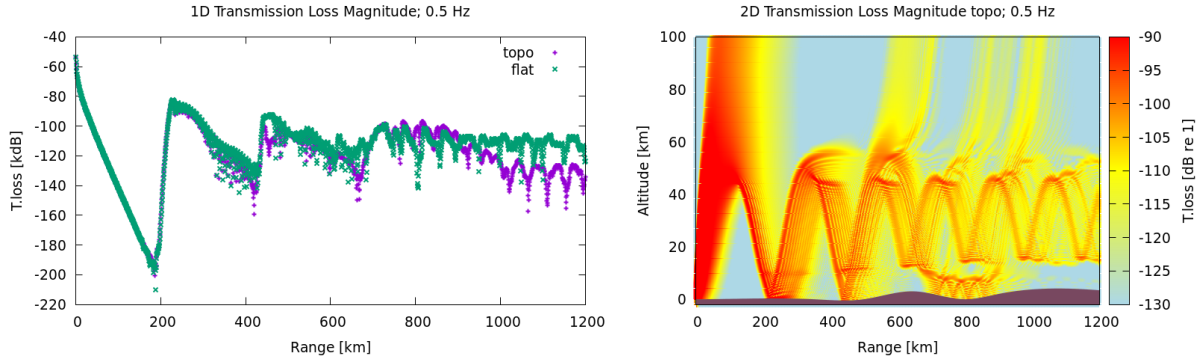


Figure 18: Topo option output for the atmosphere of example 2. Shown on the left is a comparison between ground-to-ground 1d transmission loss magnitudes over flat ground and over undulating ground. On the right is the 2d transmission loss magnitude over undulating ground. The topography is mapped out in the 2d plot.

References

[1] Jelle Assink. *Infrasound as upper atmospheric monitor*. PhD thesis, The University of Mississippi, 2012.

[2] Michael D Collins. A self-starter for the parabolic equation method. *The Journal of the Acoustical Society of America*, 92(4):2069–2074, 1992.

[3] Michael D Collins. The stabilized self-starter. *The Journal of the Acoustical Society of America*, 106(4):1724–1726, 1999.

[4] Michael D Collins and William L Siegmann. *Parabolic Wave Equations with Applications*. Springer, 2019.

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

[6] Oleg Godin. Dissipation of acoustic gravity waves: An asymptotic approach. *J. Acoust. Soc. Am.*, 2014.

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

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

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

[10] 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.

[11] David Roberts and David J Thomson. A numerically stable rational approximant for the split-step padé propagator. In *Proceedings of Meetings on Acoustics ICA2013*, volume 19, page 070076. Acoustical Society of America, 2013.

[12] J. Stoer and R. Bulirsch. *Introduction to Numerical Analysis*. Springer, N.Y. N.Y., 2 edition, 1991.

[13] Josef Stoer and Roland Bulirsch. *Introduction to numerical analysis*, volume 12. Springer Science & Business Media, 2013.

[14] 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.

[15] Roger Waxler and Jelle Assink. Propagation modeling through realistic atmosphere and benchmarking. In *Infrasound Monitoring for Atmospheric Studies*, pages 509–549. Springer, 2019.

[16] 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.