OpenSWPC: An Open-source Seismic Wave Propagation Code

User's Guide

Takuto Maeda

Version 4.0-E (2017-09-21)

Contents

Chapter 1

Set Up

1.1 System Requirements

Executing `OpenSWPC` requires a Fortran compiler that can handle (at least a part of) the Fortran 2003 standard and an MPI library. The program can be run on a single CPU or CPU core without parallelization; however, the MPI library is still required. In addition, the `NetCDF` library, compiled by the same Fortran compiler, is recommended to use the direct input/output of the `NetCDF`-formatted files.

The source code of SEISM almost strictly follows the language standard of Fortran2003. As an exception, system calls (the `system()` subroutine) are used. Note that this extension is supported by most available Fortran compilers. `OpenSWPC` uses stream I/O, which is part of the Fortran2003 standards. This functionality may not be implemented with older compilers.

This code was developed in the following environment:

- Apple OSX El Capitan
- GNU `gfortran` 6.1.0
- `OpenMPI` 1.10.2

In addition, the following computers were confirmed to work with `OpenSWPC`:

- EIC computer, ERI/UTokyo (ver. 2015; SGI Altix; intel fortran)
- JAMSTEC Earth Simulator (NEC SX-ACE; NEC compiler)
- AICS K computer (Fujitsu compiler)
- Nagoya University (FX10/Fx100; Fujitsu compiler)
- Linux Cent OS 6.6 (gfortran 4.9.2 & mpich)
- Linux Ubuntu 16.04LTS (gfortran 5.4 & OpenMPI)

1.2 Code directory tree

```
|-- doc : manuals
|-- bin : executable binaries (*.x)
|-- example : example input files
\-- src
```

Table 1.1: arch options for various environments.

arch name	target	NetCDF location
mac-intel	Mac OSX + Intel Compiler + Open MPI	${HOME}/local
mac-gfortran	Mac OSX + gfortran + Open MPI	/usr/local
eic	EIC (ERI, UTokyo) with the Intel Compiler	${HOME}/local
fx	Fujitsu FX10, FX100 and the K-computer	${HOME}/xlocal
es3	The Earth Simulator	Provided by the system
ubuntu-gfortran	Ubuntu 16.04LTS + gfortran + Open MPI	Installation by apt

```
|-- swpc_3d     : 3D problem
|-- swpc_psv    : 2D P-SV problem
|-- swpc_sh     : 2D SH problem
|-- tools       : Miscellaneous utility codes
\-- shared      : Modules commonly used in the above programs
```

1.3 Compilation and Execution

1.3.1 make

The directories `src/swpc_3d`, `src/swpc_psv`, `src/swpc_sh`, and `src/tools` contain `makefile`. Execute the `make` command in each directory to generate the executable binaries. An executable file (with a *.x extension) will be stored in the `bin` directory.

1.3.2 Specifying Compiler Options

In the makefiles, the following variables must be specified according to the environment:

`FC` compiler name

`FFLAGS` compiler option

`NCFLAG` `NetCDF` flag

`NCLIB` location of the `NetCDF` library directory

`NCINC` location of the `NetCDF` include directory

`NETCDF` linker option for `NetCDF`

If `NCFLAG = -D_NETCDF` is specified, make will try to compile `OpenSWPC` with `NetCDF`.

A set of the above variables under different computer environments is defined in `src/shared/makefile.arch` and `src/shared/makefile-tools.arch`. The former is for the compilation of FDM codes, and the latter is for the compilation of misc tools. The user can specify the `arch` option in make as shown below:

```
make arch=eic debug=true
```

The list of pre-defined architecture (`arch`) options is described in Table 1.1.

Figure 1.1: Area of JIVSM and eJIVSM. The colored area in the map is where the original JIVSM is defined. eJIVSM is extended to the gray-shaded area via an extrapolation. The surrounding graphs show the depth sections along the lines on the map of the model.

1.3.3 More about the `NetCDF` library

The `NetCDF` library consists of the following items:

`libnetcdf.*` `NetCDF` library file (static)

`libnetcdff.*` `NetCDF` Fortran library file (only `NetCDF` version 4 or later)

`netcdf.mod` Fortran module information file

The extension of the library files may be `*.a` (static library) or `*.so` (dynamic library), depending on the installation. All these files are necessary for successful compilation with `NetCDF`. In particular, the `netcdf.mod` file must be created by the same Fortran compiler as `OpenSWPC`. If `NetCDF` is installed using packaging tools such as `yum, apt`, or `homebrew`, the use of `gfortran` is implicitly assumed.

1.3.4 Preparing the Dataset

Subsurface Velocity Structure Model

In `OpenSWPC`, a 3D inhomogeneous medium can be represented as a set of velocity discontinuities using `NetCDF`-formatted files (see Section 2.7 for details). As an example of the velocity structure beneath the Japanese Archipelago, an automatic model generation script for the Japan Integrated Velocity Structure Model (JIVSM; Figure 1.1; *Koketsu et al.* (2012)), developed and originally distributed by the Headquarters for Earthquake Research Promotion in Japan, is included. An extension of JIVSM (eJIVSM), which covers a wider area, is also provided. These velocity structure models contain the ground surface (topography and bathymetry), subsurface soil, Moho, and oceanic crust of the two subducting plates. To generate these models, GMT version 4 is required. If the user does not use this model, the following processing steps may not be necessary.

First, download the original model files `lp2012nankai-e_str.zip` and `lp2012nankai-w_str.zip` and store them in `dataset/vmodel`. The URLs of these files can be found in the comments of the `gen_JIVSM.sh` script. To generate eJIVSM, the ETOPO1 (`ETOPO1_Bed_g_gmt4.grd`) topography data are also necessary.

Then, specify the Fortran compiler name to `FC` variables in the `params.sh` parameter file. The grid spacing (`dlon, dlat`) in the parameter file can be modified if necessary. Note that this spacing is not directly related to the grid width of the numerical simulations. At the time of the simulation, the `OpenSWPC` program automatically interpolates the velocity model data.

After these steps, execute the generation script:

```
./gen_JIVSM.sh
```

After a successful execution, 23 `NetCDF`-formatted files will be generated in the two model directories for `JIVSM` and `eJIVSM`. These files can be read and visualized in GMT by the `grdimage` or `grd2xyz` modules. The netcdf filename contains five integer numbers, which correspond to mass density (in kg/m^3), P wavespeed (m/s), S wavespeed (m/s), Q_P, and Q_S. They indicate the material information below the discontinuity defined in the file. List files of these NetCDF files (jivsm.lst and ejivsm.lst) for use in OpenSWPC will also be generated.

Station Lists

An example script to generate a station list file is stored in dataset/station/gen_stlst_hinet.sh. This script generates a formatted list of the high-sensitivity seismograph network Japan (Hi-net) provided by the National Research Institute for Earth Science and Disaster Resilience (NIED).

To use this script, first download the station csv list from the Hi-net website following the comments in the gen_stlst_hinet.sh file. Then, executing this bash script will result in the station list file for OpenSWPC.

1.3.5 On Embedding Parameters

Although most of the behavior of OpenSWPC is controlled dynamically by the input parameter file, several parameters are embedded in the source code to achieve high-computational performance, as described below. These parameters are defined in src/swpc_??/m_global.F90. If these parameters are modified, re-compilation is necessary.

UC = 1E-15 (1E-12 for 2D codes)
: A number to convert the simulation results into the SI unit system. Modifications may be necessary to use a different unit system.

MP = DP
: Precision of the finite-difference computation. By default (MP=DP), i.e., parts of the computation are performed in double precision, while other unnecessary variables are defined and calculated in single precision to save memory space and computation time. The user may change it to MP=SP to switch the entire computation to single precision, which will decrease the required memory up to 2/3 and allow faster computation speeds. However, in this case, a noisy seismic waveform might be observed, in particular, near the seismic source due to the overflow of floating point numbers.

NM = 3
: Number of generalized Zener viscoelastic bodies. If this number is larger than 1, it represents a nearly frequency-independent constant Q model in a specified frequency range. If this is set to zero, the simulation will be conducted with an elastic body without attenuation. Increasing this number enables the reproduction of a wider frequency range of constant Q; however, it may also result in a significant increase in the computational loads for 3D simulations.

1.3.6 Execution

To run the program, the MPI program is necessary, such that

```
> mpirun -np ${NP} ./bin/swpc_3d.x -i ${input}
```

where ${NP} is the number of MPI processes and ${input} is the name of the input file. Note that the mpirun command may be different for different computational systems.

If the program runs properly, the following message will appear in the standard error output. The result may be slightly different for different programs (3D/P-SV/SH) or execution modes.

```
------------------------------------------------------------------------------
 SWPC_3D (benchmark mode)
------------------------------------------------------------------------------

 Grid Size                :     384 x  384 x  384
 MPI Partitioning         :       4 x    6
 Total Memory Size        :          12.705  [GiB]
 Node Memory Size         :           0.529  [GiB]
 Stability  Condition c   :           0.980  (c<1)
 Wavelength Condition r   :           7.000  (r>5-10)
 Minimum velocity         :           3.500  [km/s]
 Maximum velocity         :           6.062  [km/s]
 Maximum frequency        :           1.000  [Hz]

------------------------------------------------------------------------------

 it=0000050, 1.877 s/loop, eta 000:29:43, ( 5.00E-05  5.00E-05  4.96E-05 )
 it=0000100, 1.887 s/loop, eta 000:28:18, ( 1.75E-05  1.75E-05  1.05E-05 )
 it=0000150, 1.932 s/loop, eta 000:27:22, ( 1.02E-05  1.02E-05  5.41E-06 )
 it=0000200, 1.943 s/loop, eta 000:25:54, ( 6.59E-06  6.59E-06  4.35E-06 )

         .
         .
         .

 it=0000950, 1.986 s/loop, eta 000:01:39, ( 4.89E-07  4.89E-07  1.81E-06 )
 it=0001000, 1.982 s/loop, eta 000:00:00, ( 1.65E-07  1.65E-07  1.54E-07 )

------------------------------------------------------------------------------

 Total time             :        1982.348 s

------------------------------------------------------------------------------
```

The first part of the message contains information such as the estimated memory usage, stability condition, and wavelength condition. As shown in the above example, a stability condition of $c < 1$ is mandatory to execute; if the specified parameter violates this condition, the program aborts immediately. In addition, the wavelength condition (the ratio of spatial grid size and minimum wavelength) is recommended to $r > 5-10$. During the computation, the computation speed, remaining time (eta; estimated time of arrival), and maximum velocity amplitude of the components are shown.

Chapter 2

Parameter Settings

2.1 Notation of the Parameter File

In the parameter files, one parameter is defined on each line in the following format.

```
variable_name = value
```

The description of the values should follow Fortran notation. For example, logical (Boolean) values are denoted as `.true.` or `.false.`.

Lines that do not contain an equal sign (=) will be neglected; in addition, lines starting with `!` or `#` are regarded as comment lines and will be skipped. Comments can be followed by variable definitions, that is, comments can be written on the same line as the parameter definition. For example, the following parameter line will work without errors.

```
 nx = 1024 ! number of grids
```

The order of the parameter definition can be changed freely. If a parameter is not specified, `OpenSWPC` may use a pre-defined default variable in some cases. In such a case, the use of the default-parameter will be included in the standard error output. Note that there are parameters that must be defined explicitly. Multiple definitions of the same parameters in a single parameter file are not recommended; however, in this case, the first definition may be adopted. It is acceptable to leave blanks before and after the equal sign; however, it is not permitted to have a blank space between the minus character and succeeding numbers (e.g., `'- 35.0'` is not allowed). It is recommended to use quotation marks around string parameters. Without them, the directory path character ('/') may be unexpectedly interpreted as a termination of the string parameter.

2.2 An Example Parameter File

The following is a full set of example parameters. In the following sections, detailed descriptions of each parameter will be given.

```

!! ---------------------------------------------------------------------- !!
!!
!!  SWPC input file
!!
!! ---------------------------------------------------------------------- !!
```

```
!! ---------------------------------------------------------------------- !!
!! Control
!!

title            = 'swpc'         !! exe title: used for output filenames
odir             = './out'        !! output directory
ntdec_r          = 50             !! screen report timing (1/cycle)


!! ---------------------------------------------------------------------- !!
!! Model/Grid Size and Area
!!

nproc_x          = 4              !! parallelization in x-dir
nproc_y          = 6              !! parallelization in x-dir
nx               = 384            !! total grid number in x-dir
ny               = 384            !! total grid number in y-dir
nz               = 384            !! total grid number in z-dir
nt               = 1000           !! time step number

dx               = 0.5            !! grid width in x-dir
dy               = 0.5            !! grid width in y-dir
dz               = 0.5            !! grid width in z-dir
dt               = 0.02           !! time step width

vcut             = 1.5            !! minimum velocity
                                  !- smaller velocities will be increased

xbeg             = -96.0          !! minimum in x-dir
ybeg             = -96.0          !! minimum in y-dir
zbeg             = -10.0          !! minimum in z-dir
tbeg             = 0.0            !! start time

clon             = 139.7604       !! center longitude
clat             = 35.7182        !! center latitude
phi              = 0.0            !! horizontal coordinate rotation
                                  !- measured clockwise from the north

fq_min           = 0.02           !! minimum freq. for Q-constant model
fq_max           = 2.00           !! maximum freq. for Q-constant model
fq_ref           = 1.0            !! ref. freq. for physical dispersion

!! ---------------------------------------------------------------------- !!
!! Snapshot Output
!!

snp_format       = 'netcdf'       !! snapshot format (native or netcdf)

xy_ps%sw         = .false.        !! P&S amp. for xy section
xz_ps%sw         = .true.         !! P&S amp. for xz section
yz_ps%sw         = .false.        !! P&S amp. for yz section
fs_ps%sw         = .false.        !! P&S amp. for free surface
ob_ps%sw         = .true.         !! P&S amp. for ocean bottom

xy_v%sw          = .false.        !! 3-comp. velocity for xy section
xz_v%sw          = .true.         !! 3-comp. velocity for xz section
yz_v%sw          = .false.        !! 3-comp. velocity for yz section
fs_v%sw          = .false.        !! 3-comp. velocity for free surface
ob_v%sw          = .true.         !! 3-comp. velocity for ocean bottom

```

```
xy_u%sw          = .false.          !! 3-comp. disp. for xy section
xz_u%sw          = .true.           !! 3-comp. disp. for xz section
yz_u%sw          = .false.          !! 3-comp. disp. for yz section
fs_u%sw          = .false.          !! 3-comp. disp. for free surface
ob_u%sw          = .true.           !! 3-comp. disp. for ocean bottom


z0_xy            =  7.0             !! depth for xy cross section
x0_yz            =  0.0             !! x-value for yz cross section
y0_xz            =  0.0             !! y-value for xz cross section

ntdec_s          = 5                !! time decimation of snapshot
                                    !- (specify 1 for no decimation)

idec             = 2                !! x-decimation for snapshot
jdec             = 2                !! y-decimation for snapshot
kdec             = 2                !! z-decimation for snapshot

!! ----------------------------------------------------------------------- !!
!! Waveform Output
!!

sw_wav_v         = .true.           !! velocity trace output at stations
sw_wav_u         = .false.          !! displacement trace output at stations
ntdec_w          = 5                !! time decimation of waveform output
st_format        = 'xy'             !! station format: 'xy' or 'll'
fn_stloc         = './example/stloc.xy'  !! station location file
wav_format       = 'sac'            !! 'sac' or 'csf'

!! ----------------------------------------------------------------------- !!
!! Earthquake Source
!!

!! Moment tensor source format:
!!   xymoij / xym0dc / llm0ij / llm0dc / xymwij / xymwdc / llmwij / llmwdc
!! Body force source format:
!!   xy or ll
stf_format       = 'xym0ij'

!! Basis source time function
!! 'boxcar' / 'triangle' / 'herrmann' / 'kupper' / 'cosine' / 'texp'
stftype          = 'kupper'

fn_stf           = "./example/source.dat"   !! Source grid file name

!! source depth correction
!! 'asis':use z value, 'bd{i}': i-th boundary (i=0...9)
sdep_fit         = 'asis'

  !! --------------------------------------------------------------------- !!
  !! Body force source mode
  !!
  bf_mode          = .false.


  !! --------------------------------------------------------------------- !!
  !! Plane wave source mode
  !!
  pw_mode          = .false.   !! plane wave input; neglects fn_stf
  pw_ztop          = 100.      !! top z-coordinate of the initial plane wave
  pw_zlen          = 30.       !! wavelength of the initial plane wave
  pw_ps            = 'p'       !! 'p' P-wave 's' S-wave
```

```
    pw_strike         = 0.0       !! strike direction of plane wave (deg.)
    pw_dip            = 0.0       !! dip of plane wave (deg.)
    pw_rake           = 0.0       !! rake of plane S-wave polarization (deg.)

  !! ----------------------------------------------------------------------- !!
  !! Absorbing Boundary Condition
  !!

  abc_type          = 'pml'            !! 'pml' or 'cerjan'
  na                = 20               !! absorbing layer thickness
  stabilize_pml     = .true.           !! avoid low-v layer in PML region

  !! ----------------------------------------------------------------------- !!
  !! Velocity model
  !!

  vmodel_type       = 'lhm'            !! velocity model type 'uni'/'grd'/'lhm'
  is_ocean          = .true.           !! topography z<0 is covered by ocean
  is_flatten        = .false.          !! Force topography variation to zero

    !! --------------------------------------------------------------------- !!
    !! For uniform velocity model 'uni'
    !!
    vp0               = 5.0              !! P-wave velocity [km/s]
    vs0               = 3.0              !! S-wave velocity [km/s]
    rho0              = 2.7              !! mass density    [g/cm^3]
    qp0               = 200              !! Qp
    qs0               = 200              !! Qs
    topo0             = 0                !! topography location

    !! --------------------------------------------------------------------- !!
    !! For GMT grid file input 'grd' (requires netcdf library)
    !!
    dir_grd           = '${DATASET}/vmodel/ejivsm'     !! directory for grd file
    fn_grdlst         = './example/grd.lst'            !! grd file list
    node_grd          = 0                              !! input MPI node

    !! --------------------------------------------------------------------- !!
    !! For layered homogeneous medium model ('lhm')
    !!
    fn_lhm            = 'example/lhm.dat'    !! 1D velocity structure

    !! --------------------------------------------------------------------- !!
    !! For random medium models
    !!
    dir_rmed          = './in/'             !! location of random medium file
    fn_grdlst_rmed    = './example/grd.lst' !! grd file list
    rhomin            = 1.0                 !! minimum density threshold

  !! ----------------------------------------------------------------------- !!
  !! Checkpoint/Restart
  !!
  is_ckp            = .false.          !! perform checkpoint/restart
  ckpdir            = './out/ckp'      !! output directory
  ckp_interval      = 1000000          !! interval for checkpoint (1/cycle)
  ckp_time          = 1000000.         !! checkpoint time
  ckp_seq           = .true.           !! sequential output mode

  !! ----------------------------------------------------------------------- !!
  !! Reciprocity Green's Function Mode
  !!
  green_mode        = .false.          !! reciprocity Green's function mode
```

Figure 2.1: (a) Partitioning of the computational domain for MPI. (b) Schematic of the data exchange by the MPI protocol (modified from *Maeda et al.*, 2013).

```
  green_stnm       = 'st01'            !! virtual station name from fn_stlst
  green_cmp        = 'z'               !! virtual source direction 'x', 'y', 'z'
  green_trise      = 1.0               !! rise time
  green_bforce     = .false.           !! also calc. body force Green's function
  green_maxdist    = 550.              !! horizontal limit of source grid
  green_fmt        = 'llz'             !! list file format: 'xyz' or 'llz'
  fn_glst          = 'example/green.lst'    !! Green's function grid point list

  !! ----------------------------------------------------------------------- !!
  !! MISC
  !!

  stopwatch_mode   = .true.            !! measure computation time at routines
  benchmark_mode   = .true.            !! benchmark mode

  ipad             = 0                 !! memory padding size for tuning
  jpad             = 0                 !! memory padding size for tuning
  kpad             = 0                 !! memory padding size for tuning
```

2.3 Controlling Parameters

`title`

Title of the computation to be used for the output filename.

`odir`

Name of output directory. This is a relative directory path from the location of the program execution. If this directory does not exist at the time of run, `OpenSWPC` will automatically create it.

`ntdec_r`

Number of **T**ime-step **DEC**imation factors for screen Reporting. The maximum amplitudes of the velocity components are reported in the standard error output every `ntdec_r` steps. This screen output is generally used to confirm that the model is working correctly. A cycle that is too short (this parameter is too small) may slow down the computation.

2.4 Coordinate System and Parallel Computation

For parallel computation, `OpenSWPC` performs 2D model partitioning for a 3D code (figure 2.1) and 1D partitioning for a 2D code, in the horizontal direction in both cases. The computation is performed in Cartesian coordinates. We adopt the computational coordinate system depicted in Figure 2.2. By default, the coordinate axes x, y, and z represent the north, east, and depth directions, respectively. They cover the region of `xbeg` $\leq x \leq$ `xend`, `ybeg` $\leq$ y $\leq$ `yend`, and `zbeg` $\leq z \leq$ `zend`. Note that the z-axis is defined as positive downward. Because the free surface is usually defined at $z = 0$, it is recommended to let `zbeg` be a negative value to include the free surface in the model.

Figure 2.2: Relation between computational coordinate and geographical coordinate systems.

The volume is discretized into nx, ny, and nz grids with spatial grid widths of dx, dy, and dz, respectively, in each direction. The parameter file must provide definitions of xbeg, ybeg, and zbeg and nx, ny, and nz; other parameters (xend, yend, and zend) are automatically computed from them. The center of the Cartesian coordinate ($x = 0$, $y = 0$) corresponds to the center longitude (clon) and latitude (clat). The geographical coordinate is projected onto the Cartesian coordinate by the Gauss–Krüger transform as follows (see Figure 2.2):

1. First generate an evenly spaced grid in Cartesian coordinates from the input parameters phi and those related to the x, y coordinates.
2. Project the grid location onto the geographical coordinate by using the Gauss–Krüger transform with a center location of (clon, clat).
3. Obtain the medium parameter at the grid location via a bicubic interpolation of the input velocity structure model.

If the specified area exceeds that of the input velocity model, the outermost value of the velocity structure is used for the extrapolation.

2.4.1 Staggered Grid

OpenSWPC adopts the staggered-grid coordinate system shown in Figure 2.3. The unit volume shown in the figure is defined as a "voxel" at the grid indices (I,J,K). A grid location x belongs to the voxel number

$$I = \left\lceil \frac{x - x_{\mathrm{beg}}}{\Delta x} \right\rceil, \tag{2.1}$$

and if the voxel number I is given, its center coordinate location is

$$x = x_{\mathrm{beg}} + \left(I - \frac{1}{2} \right) \Delta x, \tag{2.2}$$

where $\lceil \cdot \rceil$ is a ceiling function and x_{beg} is the minimum value of the x-coordinate. Note that x_{beg} is set to belong to the voxel I=1.

A voxel has a volume of

$$\begin{aligned} x_{\mathrm{beg}} + (I-1)\Delta x < x \le x_{\mathrm{beg}} + I \Delta x, \\ y_{\mathrm{beg}} + (J-1)\Delta y < y \le y_{\mathrm{beg}} + J \Delta y, \\ z_{\mathrm{beg}} + (K-1)\Delta z < z \le z_{\mathrm{beg}} + K \Delta z, \end{aligned} \tag{2.3}$$

The normal stress tensor components are defined at the center of the voxel, the shear stress is defined on the edge, and velocity vector components are defined on its surface (Figure 2.3) . Medium parameters are defined at the center of the voxel at

$$\begin{aligned} x_{\mathrm{beg}} + (I-1/2)\Delta x, \\ y_{\mathrm{beg}} + (J-1/2)\Delta y, \\ z_{\mathrm{beg}} + (K-1/2)\Delta z. \end{aligned} \tag{2.4}$$

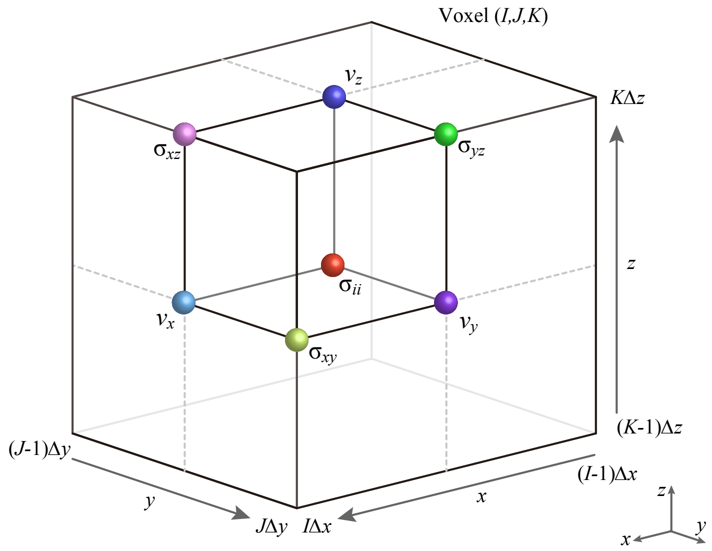

Figure 2.3: Staggered grid layout in 3D space for the case of `xbeg=ybeg=zbeg=0`.

If necessary, averaging will be performed between neighboring voxels.

The spatial grid width, Δx, Δy, and Δz, and the time step width, Δt, must satisfy the stability condition. The stability condition in N_D-dimensional space for the order of the finite difference method P is given by

$$\Delta t < \frac{1}{V_{\max}} \left(\sum_{i=1}^{N_D} \frac{1}{\Delta x_i^2} \right)^{-1/2} \left(\sum_{p=1}^{P/2} C_p \right)^{-1}, \tag{2.5}$$

where $V_{\max}$ is the maximum velocity of the medium, C_p are the coefficients of the finite difference formula, and Δx_i is the spatial grid width in the i-th direction. For the fourth-order formula of the finite difference method, which is used in the code, the coefficients are $C_1 = 9/8$ and $C_2 = 1/24$. For example, for the fourth-order finite difference with isotropic grid sizes ($\Delta x = \Delta y = \Delta z = h$) in three-dimensional space, the stability condition is reduced to

$$\Delta t < \frac{6}{7} \frac{1}{V_{\max} \sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}}} = \frac{6h}{7\sqrt{3} V_{\max}} \simeq 0.495 \frac{h}{V_{\max}}. \tag{2.6}$$

This condition can be interpreted as "the distance that the seismic wave propagates within a single time step must be much smaller than the spatial grid width." The numerical simulation will diverge immediately if this condition is not satisfied.

In addition, the minimum wavelength of the simulated seismic waves should be much longer than the spatial grid width. If the wavelength becomes relatively small compared to this condition, a fictitious numerical dispersion will appear and result into inaccurate later phases. Usually, the wavelength is taken to be longer than 5–10 times the spatial grid width to avoid this effect. Therefore, the minimum velocity (usually the S-wave velocity) in the velocity model should be selected carefully. One may specify a smaller spatial grid size to avoid this problem; however, in this case, the time-step size must also be shortened to satisfy the stability condition.

`nproc_x, nproc_y`

Number of partitions in the x- and y-directions (Figure 2.1). The total number of partitions will be `nproc_x` × `nproc_y` for the 3D case and `nproc_x` for the 2D case. This total number of partitions must be equal to the number of processes given in `mpirun`. These numbers can be 1. If `nproc_x=nproc_y=1`, this will become a serial (non-parallel) computation in practice.

`nx, ny, nz`

Total number of spatial grids in each direction. `nx` and `ny` do not need to be multiples of `nproc_x` and `nproc_y`, respectively.

`dx, dy, dz`

Spatial grid width in each direction in units of km. The total computational size in the physical domain will be `nx`×`dx`, `ny`×`dy`, and `nz`×`dz`. The grid widths in different directions do not necessarily need to be equal.

`nt`

Number of time steps.

`dt`

Length of the time step in seconds. The total (physical) simulation time will be `nt`×`dt`.

`xbeg, ybeg, zbeg`

Minimum value of the coordinates. If specifications of `xbeg` or `ybeg` are omitted, they will automatically be set to `xbeg = - nx × dx / 2` and `ybeg = - ny × dy / 2`. This setting is recommended to minimize distortion due to the map projection. The default value of `zbeg` is `-30`×`dz`.

`tbeg`

Starting time. Usually, it is set to zero.

`clon, clat`

Center longitude and latitude in degrees. The map projection will be performed with this location as a reference point.

`phi`

Horizontal rotation angle of the computational coordinate (see Figure 2.2). If `phi=0`, the x- and y-axes correspond to the north and east directions, respectively. Note that the output files (snapshot and waveform) will be rotated if this value is nonzero.

2.5 Viscoelastic Bodies

`OpenSWPC` adopts the generalized Zener body (GZB) as a model of the viscoelastic body. It consists of several viscoelastic Zener bodies with different relaxation times to attain nearly constant Q in a wide frequency range. As a consequence, it accompanies the frequency-dependent body wavespeed via physical dispersion (e.g., *Aki and Richards*, 2002); therefore, users should specify the reference frequency in which the velocity model is given.

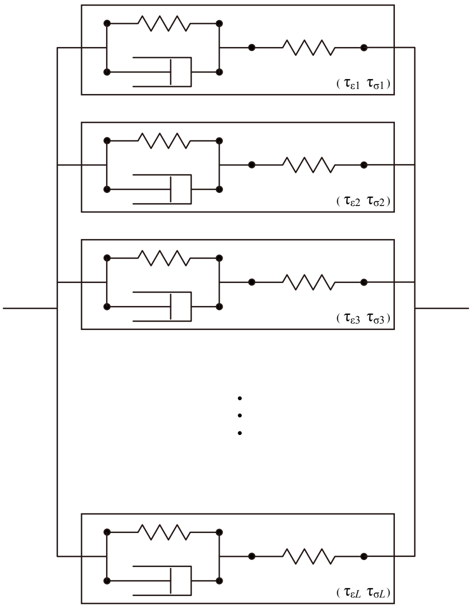

Figure 2.4: GZB.

GZB consists of N_M Zener bodies, as schematically shown in Figure 2.4. This viscoelastic body is described by the relaxation functions for an elastic moduli $\pi \equiv \lambda + \mu$ and μ, as

$$\begin{aligned} \psi_\pi (t) &= \pi_R \left(1 - \frac{1}{N_M} \sum_{m=1}^{N_M} \left(1 - \frac{\tau_m^{\varepsilon P}}{\tau_m^\sigma} \right) e^{-t/\tau_m^\sigma} \right) H(t) \\ \psi_\mu (t) &= \mu_R \left(1 - \frac{1}{N_M} \sum_{m=1}^{N_M} \left(1 - \frac{\tau_m^{\varepsilon S}}{\tau_m^\sigma} \right) e^{-t/\tau_m^\sigma} \right) H(t), \end{aligned} \tag{2.7}$$

where τ_m^σ is the relaxation time of the m-th body, $\pi_R \equiv \lambda_R + 2\mu_R$ and μ_R are the relaxed moduli, and $\tau_m^{\varepsilon P}$ and $\tau_m^{\varepsilon S}$ are creep times of the P- and S-waves, respectively. The wide frequency range of constant Q is achieved by connecting Zener bodies with different relaxation times. In addition, the intrinsic attenuations of the P- and S-waves (Q_P and Q_S, respectively) can be defined independently by choosing different creep times between the elastic moduli π and μ.

The constitutive equation between stress and strain (or particle velocity) is written as follows.

$$\begin{aligned} \dot{\sigma}_{ii}(t) &= \left(\dot{\psi}_\pi(t) - 2\dot{\psi}_\mu(t) \right) * \partial_k v_k(t) + 2\dot{\psi}_\mu(t) * \partial_i v_i(t) \qquad \text{(Do not take the sum over } i\text{.)} \\ \dot{\sigma}_{ij}(t) &= \dot{\psi}_\mu(t) * \left(\partial_i v_j(t) + \partial_j v_i(t) \right). \end{aligned} \tag{2.8}$$

The convolution appearing in the constitutive equation can be avoided by defining the memory variables (*Robertsson et al.*, 1994) and solving the auxiliary differential equations for them. We also adopt the τ-method of *Blanch et al.* (1994) to automatically choose the creep times that achieve a constant Q condition.

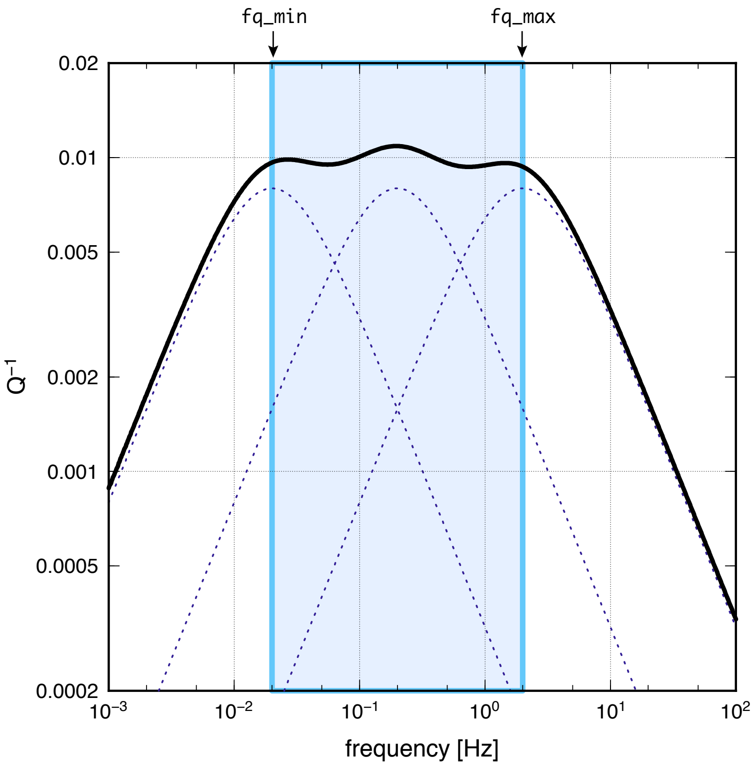

Figure 2.5: Example of frequency dependence of the intrinsic attenuation Q^{-1} for a GZB of `NM=3`. The thick solid line shows the attenuation of the entire model, while the dotted lines show the attenuation model for each constituent of the Zener body. The vertical lines show the specified minimum and maximum frequencies for the constant Q range.

fq_min
: Minimum frequency of the constant-Q model.

fq_max
: Maximum frequency of the constant-Q model.

fq_ref
: Reference frequency at which the velocity model is given.

The Q^{-1} value becomes nearly constant between the frequencies of fq_min and fq_max, as shown in Figure 2.5. Outside of the band, the attenuation becomes weaker with increasing/decreasing frequency. As shown in this figure, the nearly constant Q^{-1} is achieved using three different viscoelastic bodies. If one needs to make Q^{-1} constant over a wider frequency range, the hard-coded parameter NM should be increased. However, this leads to a significant increase in the memory usage and computational loads. The frequency dependence of Q^{-1} with the parameters specified above can be investigated using the program qmodel_tau.x (see Section 3.3.1).

2.6 Simulation Data Output

2.6.1 Output Datafile Format

OpenSWPC can export two types of data: spatiotemporal snapshots and the seismic waveform at stations.

For snapshot files, the user may choose from an originally defined binary format or a NetCDF file (recommended). The waveforms are usually exported in SAC format. The endian conversion is not performed at the time of the data output. However, the official libraries of NetCDF and SAC automatically detect the endian format and convert them if necessary. Therefore, users do not have to worry about the differences in endian formats between machines.

There is a utility program to read the original-formatted data. Note that the binary format may have slight differences for different versions of OpenSWPC. Because the format change is tracked, backward compatibility is always assured. It is recommended to use the same version of the simulation code. For SAC files, the header components described in Table 2.1 are automatically set. The units of SAC files are nm/s for velocity and nm for displacement, following the standard of SAC. While the earthquake source may be represented by multiple point sources, the header always represents the source listed in the first line of the source input file.

The snapshot file contains the header information listed in Table 2.2. These headers are commonly defined in either original format or NetCDF.

For NetCDF, these headers are set as global attributes. The other headers are set following the COARDS Conventions [1] and the CF Convention [2].

Note that the horizontal directions of the snapshot and waveforms are same as the coordinate of computation. The x- and y-axes correspond to the north and east directions only if phi=0. For the waveform, this angle, phi, is stored in the cmpaz header. The vertical-component waveform is defined as positive upward.

[1] http://ferret.wrc.noaa.gov/noaa_coop/coop_cdf_profile.html

[2] http://cfconventions.org

Table 2.1: SAC headers automatically set by OpenSWPC.

Header name	Description
kevnm	title of the parameter file
evlo,evla,evdp	The location of the event (in degrees for horizontal, in m for depth)
o	Origin time of the event listed in the first line of the source list
kzdate,kztime	Date and time of the execution of the simulation code
b	tbeg of the parameter file
delta	ntdec_w × dt
mag	The moment magnitude converted from the seismic moment
user0,..,user5	Moment tensor ($m_{xx}, m_{yy}, m_{zz}, m_{yz}, m_{xz}, m_{xy}$) of the first line of the source file
user6,user7,user8	clon, clat, phi of the parameter file
kstnm	stnm of the parameter file
stlo,stla,stdp	Station location (in degrees for horizontal, in m for depth)
kcmpnm	Vx, Vy, Vz for velocities or Ux, Uy, Uz for displacements
cmpinc,cmpaz	Station directions according to the coordinate specification
idep	7 for velocity, 7 for displacement

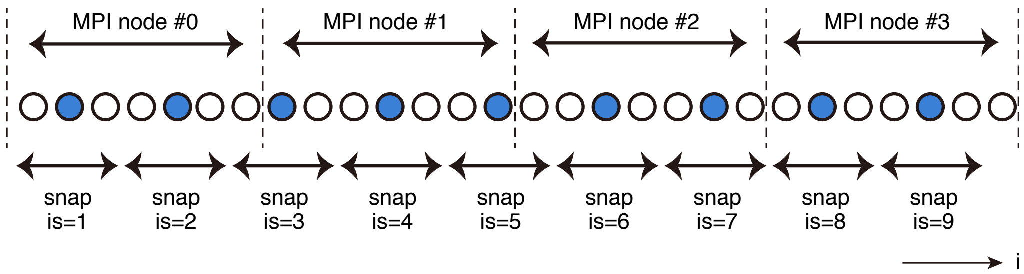

Figure 2.6: Schematic of the spatial decimation for the snapshot output. The vertical dotted lines show the borders of the MPI nodes. In this example, the data at the blue grids will be exported as the snapshot data.

2.6.2 Snapshot Data Output

Spatiotemporal snapshot output may be created along cross sections of xy, yz, and xz profiles on the topography (fs) and/or on the bathymetry (ob). There are three types of snapshots: divergence and rotation of the velocity (ps), velocity (v), and displacement (u).

The use of spatial and temporal decimations is recommended to reduce the I/O load and export data size. Decimation in time is specified by ntdec_s starting from it=0 (before starting the computation). In space, the decimations are performed by factors of idec, jdec, and kdec, and OpenSWPC tries to export the center of the decimation window, as schematically shown in Figure 2.6. The numbers of exporting grids in each MPI node do not necessarily need to be the same for each node. The amplitudes of these snapshot points will be gathered to specific nodes (see Table 2.3) and exported as single files.

snp_format

Datafile format of the snapshot files: "native" (original binary format) or "netcdf".

Table 2.2: Snapshot headers set by `OpenSWPC`.

Var	Type	Description
`bintype`	character(8)	Fixed to "STREAMIO"
`codetype`	character(8)	"SWPC_3D", "SWPC_PV", or "SWPC_SH" depending on the code
`hdrver`	integer	Header version
`title`	character(80)	`title` in the parameter file
`exedate`	integer	Date and time of the execution in POSIX time
`coordinate`	character(2)	Snapshot cross section: 'xy', 'xz', 'yz', 'fs', or 'ob'
`datatype`	character(2)	Data type: 'ps', 'v2', or 'v3'
`ns1,ns2`	integer	Number of data samples along the first and second axes
`beg1,beg2`	real	Coordinate value at the first data point of the axes
`ds1,ds2`	real	Snapshot grid spacing
`dt`	real	Time step width of the snapshot
`na1,na2`	real	Grid numbers of the absorbing boundary layer in the snapshot
`nmed`	integer	Number of stored medium parameters
`nsnp`	integer	Number of snapshots per time step
`clon,clat`	real	`clon, clat` in the parameter file
`v1,v2,v3`	real	Currently not being used

Although the `NetCDF` file format is recommended for convenience in data handling, the use of this format may lead to a slight (~ 10 %) increase in computation time.

`xy_ps%sw, xz_ps%sw, yz_ps%sw, fs_ps%sw, ob_ps%sw`
: Flags for exporting snapshot files of the PS files (`.true.` or `.false.`). If they are set to `.true.`, the divergence and rotation vector of the particle velocity will be exported.

`xy_v3%sw, xz_v3%sw, yz_v3%sw, fs_v3%sw, ob_v3%sw`
: Flags for exporting snapshot files of the velocities.

`xy_u3%sw, xz_u3%sw, yz_u3%sw, fs_u3%sw, ob_u3%sw`
: Flags for exporting snapshot files of the displacements.

`z0_xy`
: Depth (km) of the snapshot cross section.

`x0_yz`
: X-coordinate value (km) of the snapshot cross section.

`y0_xz`
: Y-coordinate value (km) of the snapshot cross section.

`ntdec_s`
: Temporal decimation factor of the snapshot output. Snapshots will be exported every `ntdec_s` time steps.

`idec, jdec, kdec`
: Spatial decimation factor of the snapshot output for the x, y, and z directions.

2.6.3 Seismic Waveform Output

Seismic velocity and/or displacement records at specified stations can be obtained as SAC-formatted files by setting the parameters `sw_wav_v` and/or `sw_wav_u` to `.true.`. Displacement

Table 2.3: MPI node number for exporting snapshot files.

Section	Type	Node
yz	PS	0
xz	PS	mod(1, nproc)
xy	PS	mod(2, nproc)
fs	PS	mod(3, nproc)
ob	PS	mod(4, nproc)
yz	V	mod(5, nproc)
xz	V	mod(6, nproc)
xy	V	mod(7, nproc)
fs	V	mod(8, nproc)
ob	V	mod(9, nproc)
yz	U	mod(10, nproc)
xz	U	mod(11, nproc)
xy	U	mod(12, nproc)
fs	U	mod(13, nproc)
ob	U	mod(14, nproc)

Table 2.4: Format of the station location file.

Type	Format				
'xy'	x	y	z	name	zsw
'll'	lon	lat	z	name	zsw

records are calculated before the decimation, and therefore, they are expected to be more accurate than performing a numerical integration of the output velocity records. The traces are stored in the memory during the computation and are exported at the end.

Station locations are given in Cartesian coordinates (`xy`) or geographical coordinates (`ll`), as in Table 2.4. In the station list, lines starting with # will be ignored.

The depth of the station can be changed depending on the variable zsw in the station list, as shown in Table 2.5. This operation is important because the station near the free surface is occasionally located above the approximated ground surface in air due to the staircase approximation of the topography and bathymetry. Usually, it is recommended to set `zsw='obb'`; this setting locates stations one-grid level below the ground surface (or seafloor).

Multiple stations can be specified in the station list file. There is no fixed limit on the number of stations. The number of stations is automatically counted, and only the stations inside the computational region will be exported.

`sw_wav_v, sw_wav_u`

Output velocity (`sw_wav_v`) and displacement (`sw_wav_v`) traces.

`ntdec_w`

Decimation factor of the waveform output. For `ntdec_w=1`, traces at every computational time step will be exported.

Table 2.5: Station depth specifications.

zsw	Station depth setting
'dep'	Calculate the station location from the given station depth
'fsb'	One grid below the free surface (for oceanic areas, the sea surface)
'obb'	One grid below the ocean bottom (seafloor) or ground surface
'oba'	One grid above of the ocean bottom (seafloor) or ground surface
'bdi' (i=0, ···, 9)	Internal velocity discontinuity specified by the velocity model

Table 2.6: csf headers.

Header name	Description
nvhdr	Format version numbers. Always zero.
ntrace	Number of traces in the file.
npts	Number of time samples in the trace.

st_format
Format of the station list file. See Table 2.4.

fn_stloc
Station location filename.

wav_format
Station file format: 'sac' (usual, recommended) or 'csf'.

The csf format

Because the SAC format is defined to express the data at one component of one station in a single file, the number of files may become extraordinarily large. In this case, data transfer between computers will become very inefficient. For OpenSWPC version 3.0 or later, users can choose a concatenated SAC format (csf) for the data output by specifying wav_format='csf'. This is a set of SAC binary files connected to a single file, with headers as in Table 2.6. The header consists of three of four-byte floating-point numbers. After the header, SAC-formatted trace records are repeated ntrace times. If this format is selected, the csf files are created at every computation node with a node number in the filename for every component of the traces.

2.6.4 Output Filename Conventions

Output data names are determined by the following rules:

- Snapshot (odir)/(title).(section).(type).snp
- Waveform (odir)/wav/(title).(stnm).(component).sac
- Computation time (odir)/wav/(title).tim

In the above rules, (section) takes a cross section such as xy or yz. (type) takes v or ps depending on the snapshot data type. (component) takes Vx, Vy, or Vz for the velocity or Ux, Uy, or Uz for the displacement.

2.7 Velocity Model

2.7.1 Choice of Velocity Model Type

Users can choose a uniform (`uni`), a layered homogeneous medium (`lhm`), or a `NetCDF` (grd) file input (`grd`) velocity type. In addition, a randomly inhomogeneous medium calculated by an external program can be overlaid onto the model.

`vmodel_type`
: Specify the input velocity model. Choose from one of the following.

  `'uni'` Homogeneous medium with a free surface. The following additional parameters are required:

  - `vp0` P-wave velocity [km/s] in the uniform model.
  - `vs0` S-wave velocity [km/s] in the uniform model.
  - `rho0` Mass density [g/cm^3] in the uniform model.
  - `qp0` Q_P of the uniform model.
  - `qs0` Q_S of the uniform model.
  - `topo0` Topography depth in the uniform model. If this value is greater than zero, seawater is filled from $z = 0$ to this depth.

  `'lhm'`
  Layered Homogeneous Medium. The one-dimensional velocity structure file should be specified as below.

  `fn_lhm`
  Medium specification file. Each line specifies the depth of the top of the layer, density, P-wave velocity, S-wave velocity, Q_P, and Q_S below that depth. They must be separated by space(s) (see following example). Lines starting with # will be neglected.

```
# depth   rho(g/cm^3)   vp(km/s)    vs(km/s)      Qp        Qs
# ---------------------------------------------------------------
      0         2.300       5.50        3.14      600       300
      3         2.400       6.00        3.55      600       300
     18         2.800       6.70        3.83      600       300
     33         3.200       7.80        4.46      600       300
    100         3.300       8.00        4.57      600       300
    225         3.400       8.40        4.80      600       300
    325         3.500       8.60        4.91      600       300
    425         3.700       9.30        5.31      600       300
```

  `'grd'`
  Velocity model input from `NetCDF` (GMT grd) files. The compilation of `OpenSWPC` should be performed in accompaniment with the use of the `NetCDF` library. The following parameters are required.

  `dir_grd`
  Directory of the velocity structure (`NetCDF`) files.

`fn_grdlst`

List file that specifies the grd files and the associated medium. Each line contains the grd filename (with a single or double quotation mark; recommended), density, P-wave velocity, S-wave velocity, Q_P, Q_S, and the layer number integers (0-9) separated by spaces (see following example). Lines starting with # will be neglected. The layer number is used to specify the source or station depth fit to the layer depth. The first `NetCDF` file will be treated as the ground surface. If the depth of the ground surface is deeper than zero, the depth range from $z = 0$ to the surface is assumed to be an ocean layer. The grid above the free surface is treated as an air column.

```
# grd filename            rho     vp     vs     QP    QS   sw
# -------------------------------------------------------
'eJIVSM_01_TAB_.grd'    1.80   1.70   0.35    119    70   0
'eJIVSM_02_BSM_.grd'    1.95   1.80   0.50    170   100   0
'eJIVSM_03_BSM_.grd'    2.00   2.00   0.60    204   120   0
'eJIVSM_04_BSM_.grd'    2.05   2.10   0.70    238   140   0
'eJIVSM_05_BSM_.grd'    2.07   2.20   0.80    272   160   0
'eJIVSM_06_BSM_.grd'    2.10   2.30   0.90    306   180   0
'eJIVSM_07_BSM_.grd'    2.15   2.40   1.00    340   200   0
'eJIVSM_08_BSM_.grd'    2.20   2.70   1.30    442   260   0
'eJIVSM_09_BSM_.grd'    2.25   3.00   1.50    510   300   0
'eJIVSM_10_BSM_.grd'    2.30   3.20   1.70    578   340   0
'eJIVSM_11_BSM_.grd'    2.35   3.50   2.00    680   400   0
'eJIVSM_12_BSM_.grd'    2.45   4.20   2.40    680   400   0
'eJIVSM_13_BSM_.grd'    2.60   5.00   2.90    680   400   0
'eJIVSM_14_BSM_.grd'    2.65   5.50   3.20    680   400   0
'eJIVSM_15_UPC_.grd'    2.70   5.80   3.40    680   400   0
'eJIVSM_16_LWC_.grd'    2.80   6.40   3.80    680   400   0
'eJIVSM_17_CTM_.grd'    3.20   7.50   4.50    850   500   0
'eJIVSM_18_PH2_.grd'    2.40   5.00   2.90    340   200   1
'eJIVSM_19_PH3_.grd'    2.90   6.80   4.00    510   300   0
'eJIVSM_20_PHM_.grd'    3.20   8.00   4.70    850   500   0
'eJIVSM_21_PA2_.grd'    2.60   5.40   2.80    340   200   2
'eJIVSM_22_PA3_.grd'    2.80   6.50   3.50    510   300   0
'eJIVSM_23_PAM_.grd'    3.40   8.10   4.60    850   500   0
```

`node_grd`

MPI node to input the `NetCDF` data. All `NetCDF` files are first read by this node, and then, transferred to all nodes via MPI data communication.

`is_ocean`

In the default (`.true.`), the depth from $z = 0$ to the set topography will be treated as an ocean layer. If this parameter is set to `.false.`, the seafloor will be used as a free surface and no seawater will be used.

`'user'`

A user subroutine defined in `src/swpc_*/m_vmodel_user.F90` is used for defining velocity model. Recompilation of the code is necessary if this Fortran file is modified. Please refer the comments in the file for input/output of the user subroutine.

`vcut`

Cut-off velocity. For the `'lhm'` or `'grd'` models, a velocity slower than this value will be overwritten by the `vcut` value. This parameter is used to avoid wavelengths that are too short and violate the wavelength condition (the wavelength is recommended to be longer than 5-10 grids). This substitution will not be performed in the oceanic area.

On Treatments of Air and Seawater Layer

In `OpenSWPC`, the air column has a mass density of $\rho = 0.001$ [g/cm^3], velocities of $V_P = V_S = 0$ [km/s], and intrinsic attenuation parameters of $Q^P = Q^S = 10^{10}$. In the ocean column, $\rho = 1.0$ [g/cm^3], $V_P = 1.5$ [km/s], $V_S = 0.0$ [km/s], and $Q^P = Q^S = 10^6$ are assumed. The air column is treated as a vacuum with no seismic wave propagation (with zero velocities). However, the mass density must not be zero to avoid division by zero. In the free surface and seafloor, the reduced order of the finite difference is performed according to (*Okamoto and Takenaka*, 2005; *Maeda and Furumura*, 2013). These discontinuities are automatically detected as boundaries that change μ and λ from zero to a finite value.

2.7.2 Small-Scale Random Inhomogeneity

Users may overlay small-scale velocity inhomogeneities with specified power-law spectra on the background velocity models of `'uni'`, `'lhm'`, and `'grd'`. The small-scale velocity inhomogeneity ξ is defined by external files. From the average velocities V_{P0}, V_{S0}, and ρ_0, the fluctuated velocities and density are given as

$$
\begin{aligned}
V_P &= V_{P0}\,(1+\xi) \\
V_S &= V_{S0}\,(1+\xi) \\
\rho &= \rho_0\,(1+\nu\xi)\,,
\end{aligned}
\tag{2.9}
$$

where $\nu = 0.8$ is a scaling parameter based on a laboratory experiment (Birch's law; *Sato et al.*, 2012).

Velocity models having this small-scale inhomogeneity are specified by appending `_rmed` to the original velocity models: `vmodel_type='uni_rmed'`, `'lhm_rmed'`, or `'grd_rmed'`. For random media generation, the readers are referred to Section 3.4.1.

`dir_rmed`
: A directory name for storing the random media data files.

The random media are given as two- or three-dimensional `NetCDF` files. At each grid location, the velocity fluctuation $\xi(I, J, K)$ is defined. The code automatically reads the corresponding volume from the file; It is not necessary to decompose the `NetCDF` files into parts for parallel computation. If the computational size (`Nx,Ny,Nz`) is larger than the random media file size, the media is used repeatedly by applying a circular boundary condition. The simulation codes do not care if the grid sizes of the simulation and the input random media file are identical.

Parameters for `uni_rmed`

The following parameter is required in addition to the parameters used in `vmodel='uni'`.

`fn_rmed0`
: Name of the random medium file.

In this model, the average velocity will be fluctuated based on the input `fn_rmed0`.

Parameters for lhm_rmed

In this model, the small-scale velocity fluctuation is applied to every layer defined by `vmodel='lhm'`. It is possible to assign different random velocity models at different layers.

The following parameter is substituted in `fn_lhm`:

`fn_lhm_rmed`
: List file of the velocity structure.

The list file has a similar format to `fn_lhm`; it contains the filenames of the random media files in the rightmost column as in the following example.

```
# depth   rho(g/cm^3)  vp(km/s)   vs(km/s)     Qp      Qs      fn_rmed
# ----------------------------------------------------------------------
      0        2.300      5.50       3.14     600     300     rmedia1.nc
      3        2.400      6.00       3.55     600     300     rmedia1.nc
     18        2.800      6.70       3.83     600     300     rmedia2.nc
     33        3.200      7.80       4.46     600     300     rmedia2.nc
    100        3.300      8.00       4.57     600     300     -
    225        3.400      8.40       4.80     600     300     -
    325        3.500      8.60       4.91     600     300     -
    425        3.700      9.30       5.31     600     300     -
```

In this example, the layers starting from depths of 0 km and 3 km have fluctuations defined in `rmedia1.nc`, and the layers from 18 km and 33 km are defined in `rmedia2.nc`. For the layer deeper than 100 km, a dummy filename (-) is given. In this case (i.e., there is no file found), a fluctuation will not be given. The dummy filename is mandatory in this case.

Parameters for grd_rmed

When overlaying the random fluctuations to the layers defined by the model of `vmodel='grd'`, it is possible to assign different random media to different layers. The starting depth of the velocity fluctuation can be either the free surface or depths defined by a layer.

The filename of the velocity fluctuation is given by the following parameter:

`fn_grdlst_rmed`
: A list file that specifies the velocity layer and the random fluctuation files for each layer.

The list file has two additional columns at the right: the filename of the random medium and the reference layer number.

```
# grd filename           rho    vp    vs    QP    QS   sw  fn_rmed        ref
# ---------------------------------------------------------------------------
'eJIVSM_01_TAB_.grd'    1.80  1.70  0.35   119    70   0 'rmed3d_1.nc' 0
'eJIVSM_02_BSM_.grd'    1.95  1.80  0.50   170   100   0 'rmed3d_1.nc' 0
'eJIVSM_03_BSM_.grd'    2.00  2.00  0.60   204   120   0 'rmed3d_1.nc' 0
'eJIVSM_04_BSM_.grd'    2.05  2.10  0.70   238   140   0 'rmed3d_1.nc' 0
'eJIVSM_05_BSM_.grd'    2.07  2.20  0.80   272   160   0 'rmed3d_1.nc' 0
'eJIVSM_06_BSM_.grd'    2.10  2.30  0.90   306   180   0 'rmed3d_1.nc' 0
'eJIVSM_07_BSM_.grd'    2.15  2.40  1.00   340   200   0 'rmed3d_1.nc' 0
'eJIVSM_08_BSM_.grd'    2.20  2.70  1.30   442   260   0 'rmed3d_1.nc' 0
'eJIVSM_09_BSM_.grd'    2.25  3.00  1.50   510   300   0 'rmed3d_1.nc' 0
'eJIVSM_10_BSM_.grd'    2.30  3.20  1.70   578   340   0 'rmed3d_1.nc' 0
'eJIVSM_11_BSM_.grd'    2.35  3.50  2.00   680   400   0 'rmed3d_1.nc' 0
```

```
'eJIVSM_12_BSM_.grd'   2.45  4.20  2.40   680   400   0 'rmed3d_1.nc' 0
'eJIVSM_13_BSM_.grd'   2.60  5.00  2.90   680   400   0 'rmed3d_1.nc' 0
'eJIVSM_14_BSM_.grd'   2.65  5.50  3.20   680   400   0 'rmed3d_1.nc' 0
'eJIVSM_15_UPC_.grd'   2.70  5.80  3.40   680   400   0 'rmed3d_1.nc' 0
'eJIVSM_16_LWC_.grd'   2.80  6.40  3.80   680   400   0 'rmed3d_3.nc' 0
'eJIVSM_17_CTM_.grd'   3.20  7.50  4.50   850   500   0 'rmed3d_3.nc' 0
'eJIVSM_18_PH2_.grd'   2.40  5.00  2.90   340   200   1 'rmed3d_2.nc' 18
'eJIVSM_19_PH3_.grd'   2.90  6.80  4.00   510   300   0 'rmed3d_2.nc' 18
'eJIVSM_20_PHM_.grd'   3.20  8.00  4.70   850   500   0 'rmed3d_3.nc' 18
'eJIVSM_21_PA2_.grd'   2.60  5.40  2.80   340   200   2 'rmed3d_2.nc' 21
'eJIVSM_22_PA3_.grd'   2.80  6.50  3.50   510   300   0 'rmed3d_2.nc' 21
'eJIVSM_23_PAM_.grd'   3.40  8.10  4.60   850   500   0 'rmed3d_3.nc' 21
```

The reference layer number defines the reference depth plane of the random media. If this number is zero, the depth grid number of the computational model is directly used to assign the random media. This is exactly the same as the behavior of the `uni_rmed` or `lhm_rmed` models. If the nonzero value of the reference layer number `NR` is specified, the depth of the `NR` layer is treated as the base plane. The depth grid of the random medium is measured from this depth. Introducing this reference plane, the inclined random media according to the velocity discontinuity (such as the plate boundary) can be specified. In the above example, the 18th and 21st layers are treated as the references of 18–20th and 21–23th layers, respectively.

Truncation of Velocity Fluctuations

If the magnitude of the velocity fluctuation becomes too large, there can be a spot with non-physical velocity, such as negative velocity or a velocity too large for the Earth medium. The simulation may be unstable under the following conditions:

1. The fluctuated velocity $V = (1 + \xi)V_0$ exceeds the stability condition for cases with $\xi > 0$.
2. The velocity has unrealistic negative values for cases with $\xi < -1.0$.
3. The mass density has negative values for cases with $\xi < -1.25$.

To avoid such situations, `OpenSWPC` automatically limits the range of the fluctuated velocity to `vcut`$\leq v \leq 0.95 \times v_{\max}$, where `vcut` is an input parameter and $v_{\max}$ is the maximum possible velocity derived from the stability condition.

In addition, the following parameter controls the minimum density.

`rhomin`
: Minimum mass density in g/cm^3. (1.0 g/cm^3 by default.)

2.8 Earthquake Source Specification

2.8.1 Moment Rate Function

This section describes the moment rate functions, $\dot{M}(t)$, that can be used in `OpenSWPC` by choosing the parameter `stftype`. In the following, all moment rate functions have a duration (or characteristic time) T_R and are normalized so that the total moment is 1.

Box-car function (`boxcar`)	$\dot{m}^{R}(t) = \dfrac{1}{T_R}$	$(0 \le t \le T_R)$	(2.10)
Triangle function (`triangle`)	$\dot{m}_R^{T}(t) = \begin{cases} 4t/T_R^2 \\ -4(t - T_R)/T_R^2 \end{cases}$	$\begin{matrix} (0 \le t \le T_R/2) \\ (T_R/2 < t \le T_R) \end{matrix}$	(2.11)
Herrmann function (`herrmann`)	$\dot{m}^{H}(t) = \begin{cases} 16t^2/T_R^3 \\ -2(8t^2 - 8tT_R + T_R^2)/T_R^3 \\ 16\,(t - T_R)^2\,/T_R^3 \end{cases}$	$\begin{matrix} (0 \le t \le T_R/4) \\ (T_R/4 < t \le 3T_R/4) \\ (3T_R/4 < t \le T_R) \end{matrix}$	(2.12)
Cosine function (`cosine`)	$\dot{m}^{C}(t) = \dfrac{1}{T_R}\left[1 - \cos\left(\dfrac{2\pi t}{T_R}\right)\right]$	$(0 \le t \le T_R)$	(2.13)
Küpper Wavelet (`kupper`)	$\dot{m}^{K}(t) = \dfrac{3\pi}{4T_R}\sin^3\left(\dfrac{\pi t}{T_R}\right)$	$(0 \le t \le T_R)$	(2.14)
$t - \exp$ type (`texp`)	$\dot{m}^{E}(t) = \dfrac{(2\pi)^2 t}{T_R^2}\exp\left[-\dfrac{2\pi t}{T_R}\right]$	$(t \ge 0)$	(2.15)

Figure 2.7 shows each moment rate function and its Fourier spectrum. The moment rate functions have a roll off of f^{-1}–f^{-4} at frequencies of $f \gg 1/T_R$. To avoid numerical dispersion, the source spectrum should be sufficiently small at the highest target frequency. As this maximum frequency, we adopt $f_{\max} = 2/T_R$ for all types of source time functions (the red dotted line in Figure 2.7). If the parameter is appropriately set so that numerical dispersion does not occur at frequencies below $f_{\max}$, the result should not be contaminated by numerical dispersion. In addition, the uppermost frequency, where the spectrum response of the source time function becomes flat in the frequency domain, is approximately $f \le 1/(2T_R)$ (the blue dotted line in Figure 2.7).

2.8.2 Moment Tensor Source

The source mechanisms of the faulting are given by a six-component moment tensor or by three parameters of a double couple source (`strike`, `dip`, `rake`). The source locations can be given either by their computational or geographical coordinates. Therefore, there are eight possible formats to describe the source (see Table 2.7). In the program, sources are given as a stress-drop source by using the moment rate function. The moment rate function is chosen from the given six functions (Figure 2.7). They require parameters in the source list file for their starting time T_0, duration T_R, and total moment M_0.

`OpenSWPC` can accept multiple point sources as multiple lines in the source list file. There is no fixed limit to the number of sources (in practice, this is determined by the memory size). By gradually changing the starting time and source location, a finite fault rupture can be mimicked. In the source list file, lines starting with # will be ignored. By setting `sdep_fit`, the source depth can be changed so that it fits the medium's velocity boundary.

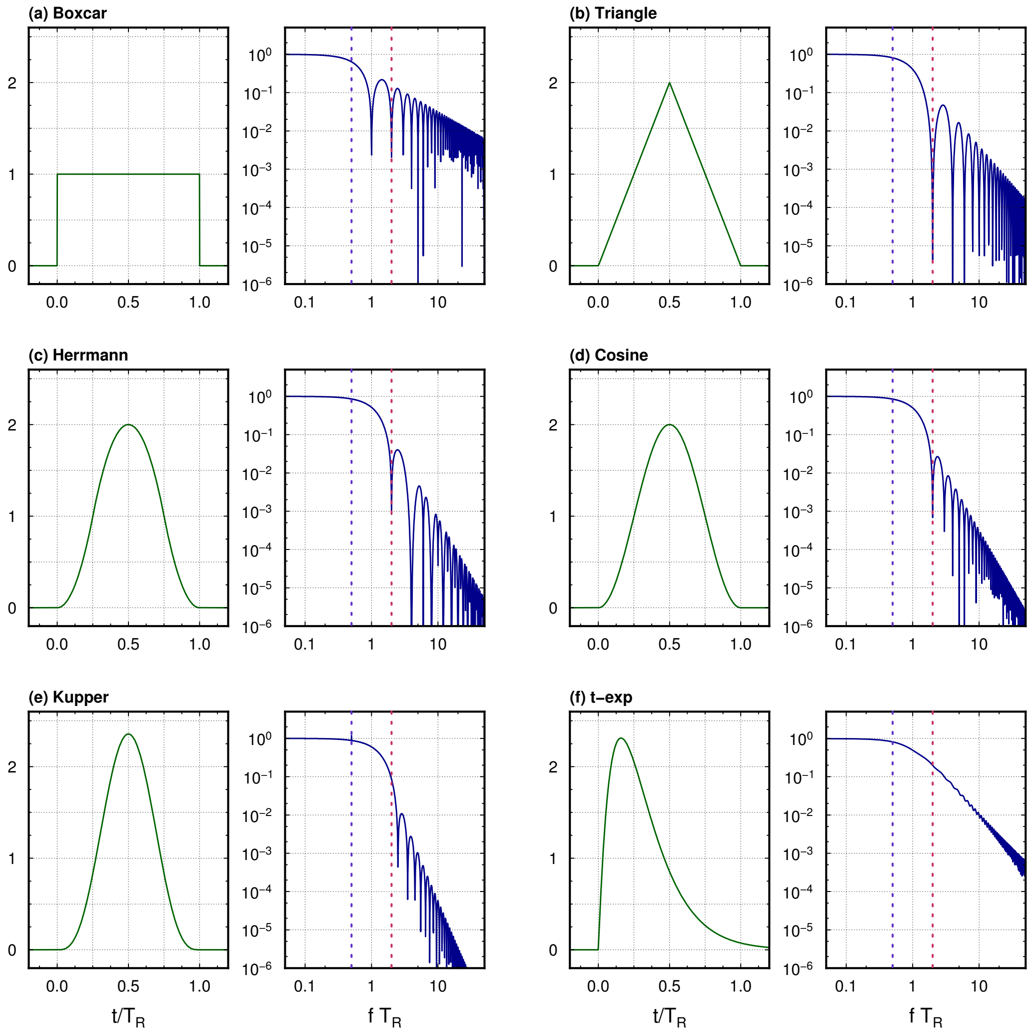

Figure 2.7: Moment rate functions $\dot{M}(t)$ (left) and their Fourier spectra (right).

Table 2.7: Format of the source list file.

Type	Format											
'xym0ij'	x	y	z	T_0	T_R	M_0	m_{xx}	m_{yy}	m_{zz}	m_{yz}	m_{xz}	m_{xy}
'xym0dc'	x	y	z	T_0	T_R	M_0	strike	dip	rake			
'llm0ij'	lon	lat	z	T_0	T_R	M_0	m_{xx}	m_{yy}	m_{zz}	m_{yz}	m_{xz}	m_{xy}
'llm0dc'	lon	lat	z	T_0	T_R	M_0	strike	dip	rake			
'xymwij'	x	y	z	T_0	T_R	M_W	m_{xx}	m_{yy}	m_{zz}	m_{yz}	m_{xz}	m_{xy}
'xymwdc'	x	y	z	T_0	T_R	M_W	strike	dip	rake			
'llmwij'	lon	lat	z	T_0	T_R	M_W	m_{xx}	m_{yy}	m_{zz}	m_{yz}	m_{xz}	m_{xy}
'llmwdc'	lon	lat	z	T_0	T_R	M_W	strike	dip	rake			

In this case, the depth in the source list file will be ignored. The layer number should be specified in the `fn_grd` or `fn_grd_rmed` list files.

`stf_format`
: Format of the source list file. Choose from `'xym0ij'` or `'llmwdc'` for example. See Table 2.7 for the complete list.

`stftype`
: Choice of the source time function. Select from `'boxcar'`, `'triangle'`, `'herrmann'`, `'kupper'`, `'cosine'`, and `'texp'`. See Figure 2.7 for these functions.

`fn_stf`
: Filename of the source list.

`sdep_fit`
: Flag to fit the source depth to the velocity discontinuity. `'asis'`: do not fit (default). `'bd{i}'`(i=1,2,$\cdots$9): fits to the `i`-th boundary specified in the rightmost column of `fn_grdlst`.

2.8.3 Body Force Mode

A body force source can be used instead of a moment tensor source. In this mode, the three-component force vector (f_x, f_y, f_z) should be specified. The force vector is assumed to have a bell-shaped source time function, as in the case of the moment tensor source. Although there is no restriction on the number of body force elements, it is not possible to use both a moment tensor and a body force at the same time.

`bf_mode`
: Flag for the body force mode. If this is `.true.`, the following parameters are used for the body force and the moment tensor source is ignored.

`stf_format`
: Format of the source file. See Table 2.8.

`stftype`
: Choice of the source time function. Same as the case with a moment tensor source.

Table 2.8: File formats of the body force files.

Type	Format							
'xy'	x	y	z	T_0	T_R	f_x	f_y	f_z
'll'	lon	lat	z	T_0	T_R	f_x	f_y	f_z

`fn_stf`
: Filename of the source list file. The format is described in Table 2.8.

`sdep_fit`
: Flag to fit the source depth to a specified velocity discontinuity. Same as the case with a moment tensor source.

2.8.4 Plane Wave Mode

A plane wave incident from the bottom can be used as an input source instead of the moment tensor or body force sources. In `OpenSWPC`, plane wave incidence is achieved by setting the velocity vector and stress tensor components based on the analytic solution of a plane wave propagating upward as the initial condition.

The specification of the initial conditions includes the depth of the initial plane wave (`pw_ztop`) and its characteristic length (`pw_zlen`; corresponding to the wavelength), the strike and dip angle of the plane wave (`pw_strike`, `pw_dip`), and the polarization direction (rake angle) in the case of an S-wave (`pw_rake`). See Figure 2.8 for the geometry. The definitions of the strike, dip, and rake parameters follow those of the earthquake source fault geometry of *Aki and Richards* (2002). For three-dimensional space, `pw_strike=0` results in the plane dip toward the y-direction (east for `phi=0`). A rake angle of `pw_rake=0`° or `pw_rake=180`° will result in pure SH waves whose polarization is parallel to the free surface.

The initial plane wave occupies a depth range of `pw_zlen` (km) starting at depths of $z =$`pw_ztop` at the center of the horizontal coordinate. The depth dependence of the wave amplitude is determined by the source time functions used in the moment rate function as a function of space (Figure 2.8). Via the definition of the source time function, the integration of the initial plane wave along the propagation direction will be normalized to 1.

`pw_mode`
: Flag to use the plane-wave mode. If it is `.true.`, all point-source locations (body force or moment tensor source) will be ignored.

`pw_ztop`
: z-value of the top of the initial plane wave at $x = y = 0$.

`pw_zlen`
: Characteristic spatial scale of the initial plane wave.

`pw_ps`
: Plane wave type. Choose from `'p'` or `'s'`

`pw_strike`
: Strike direction of initial plane wave in degrees measured from the x-axis.

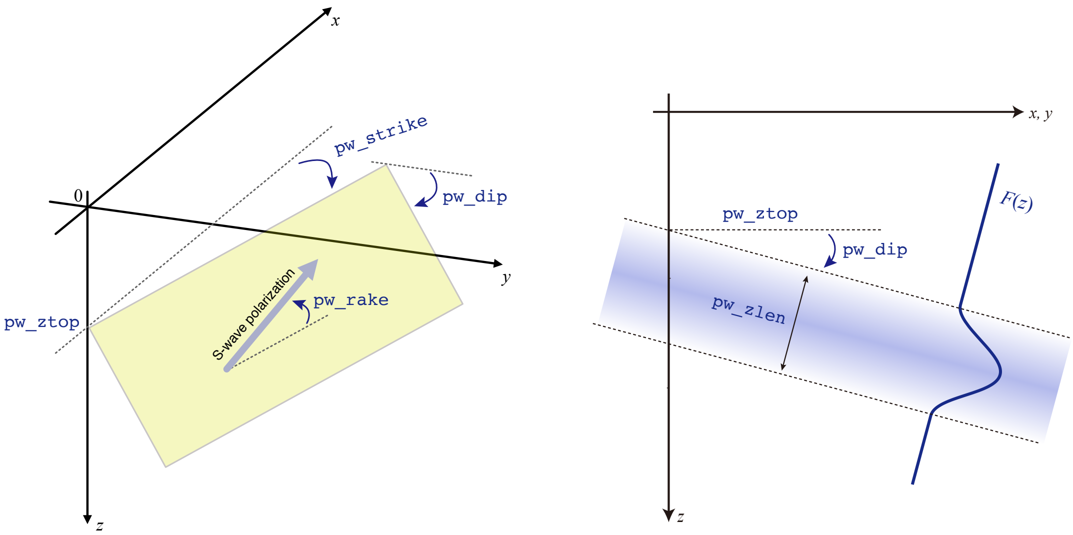

Figure 2.8: Geometry of the plane wave specification. (Left) The specification of the uppermost plane and the polarization direction. (Right) The depth cross section of the initial plane wave.

`pw_dip`
: Dip angle of the initial plane wave in degrees. The initial plane wave propagates vertically if this angle is zero.

`pw_rake`
: Polarization direction of initial plane S-wave in degrees measured from the horizontal plane.

`stftype`
: Source time function type. Same as the cases with the moment tensor or body force sources.

The use of the PML absorbing boundary condition (`abc_type='pml'`; see Section 2.9) is strongly recommended for the case of plane wave incidence. The simple Cerjan's (`abc_type='cerjan'`) condition always causes significant contamination by artificial reflections (Figure 2.9). Even when using the PML boundary, the tilted plane wave incidence (with nonzero `pw_dip` angle) causes some amount of artificial reflections. It is highly recommended that the boundary effect be confirmed with snapshot visualization when using this plane wave mode.

2.9 Absorbing Boundary Conditions

Users can choose an absorbing boundary condition from the auxiliary differential equation, the complex frequency-shifted perfectly matched layer (ADE CFS-PML *Zhang and Shen*, 2010), and Cerjan's sponge condition (*Cerjan et al.*, 1985).

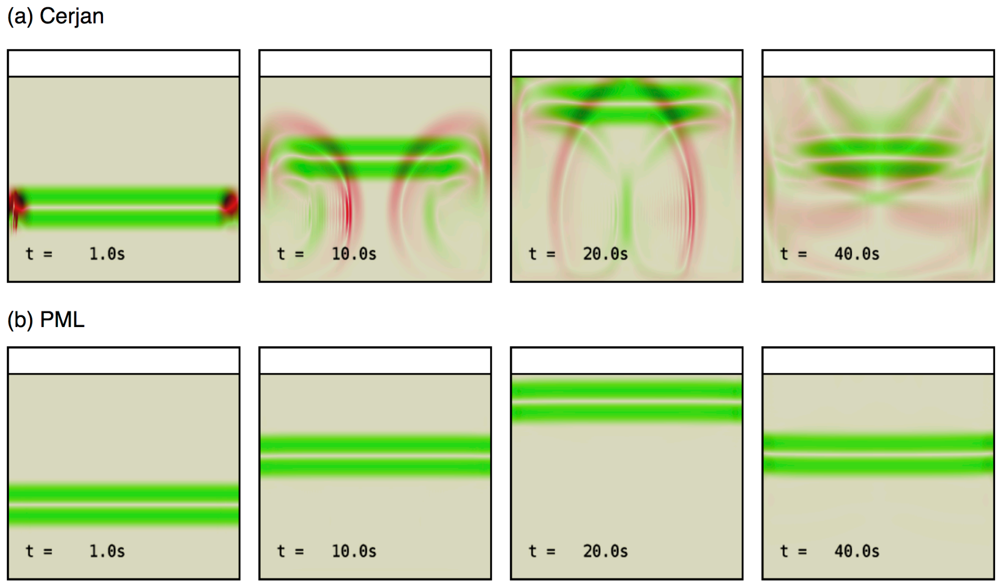

Figure 2.9: Snapshots of the absolute values of divergence (red) and rotation (green) for the case of vertical plane S-wave incidence with (a) Cerjan's condition and (b) PML boundary conditions.

The entire computational domain is separated into interior and exterior regions by the thickness of the absorber `na`, as shown in Figure 2.10. Because this program assumes the existence of a free surface and ignores acoustic waves in the air column, the waves in the top boundary will not be absorbed. At a given horizontal grid location `(I,J)`, the depth grid deeper than `kbeg_a` will be used as the attenuator.

For computational efficiency in the PML boundary condition, `OpenSWPC` does not solve the viscoelastic constitutive equation in the absorber. Note that, in the case of a medium having very small Q values, this may lead to a velocity gap between the interior and exterior regions due to physical dispersion.

For Cerjan's absorbing condition, the parameters suggested by *Cerjan et al.* (1985) are embedded in the source code. However, these parameters are scaled according to the width of the absorber `na`.

The PML absorber is usually far superior to Cerjan's sponge in its efficiency in avoiding artificial reflection from the boundaries. However, PML occasionally results in numerical instabilities, particularly for a medium with a strong velocity contrast and after several time steps. In such cases, Cerjan's sponge always gives a very stable result.

`abc_type`
: Type of the absorbing boundary condition. Choose from `'pml'` or `'cerjan'`.

`na`
: Thickness of the absorbing layer in numbers of grids. Usually, 10-20 grids are chosen.

`stabilize_pml`
: The low velocity layer is removed if this flag is `.true.`, to stabilize PML.

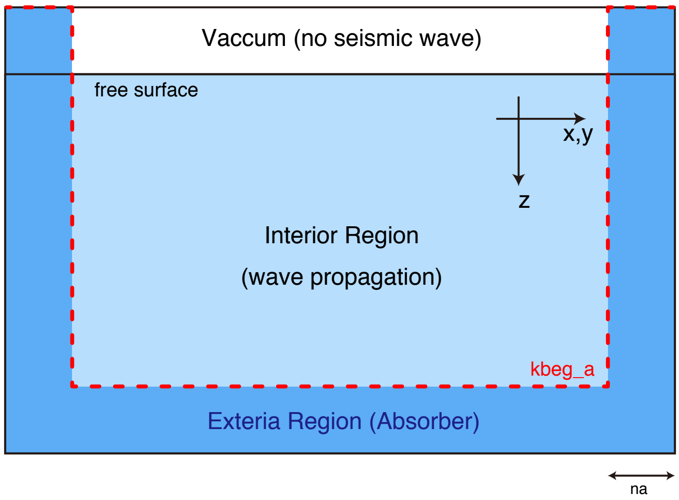

Figure 2.10: Schematic of the definition of the absorber region. The red dotted line indicates the location of `kbeg_a(I,J)`.

2.10 Checkpointing and Restarting

Some large-scale computers limit the computational time of a single job. To achieve long-duration computation, `OpenSWPC` can export all memory contents to files at specific times (checkpointing), and then continue the simulation as another job (restarting).

If this function is turned on, `OpenSWPC` will terminate the computation after an elapsed time of `ckp_time` (in seconds) and will export all memory images.

For the next job, `OpenSWPC` first tries to find the directory `cdir` to locate the checkpointing file. If there are checkpointing files, `OpenSWPC` reads them to continue the simulation. Otherwise, `OpenSWPC` starts the simulation from scratch.

After finishing the computation of all time steps, `OpenSWPC` removes most of the contents of the checkpointing files. However, it does not delete the checkpointing files. This is to avoid unexpectedly starting the computation from the beginning and overwriting the output files.

This function is only available for the three-dimensional simulation code (`swpc_3d.x`).

`is_ckp`
: The flag to use checkpointing/restarting.

`cdir`
: Output directory name of the checkpointing file. At restart, the checkpointing files are assumed to be in this directory.

`ckp_time`
: Checkpointing time in seconds.

`ckp_interval`
: Investigate if the computation time exceeds `ckp_time` periodically at this interval. Setting this interval step as too small may affect the performance of the computation.

`ckp_seq`
: Sequential output mode. If this flag is `.true.`, the I/O of the checkpointing files is sequentially performed from the zero-th MPI node. If the file system is shared by several computational nodes, this flag effectively improves the I/O performance.

2.11 Reciprocity Mode

This mode excites the seismic wave at a specified station location and exports the velocity and/or strain velocity of multiple virtual source locations. Based on the reciprocity theorem, this result corresponds to the body force and/or moment tensor response from virtual source locations observed at specified stations. If the time duration of the source time function is sufficiently short, they can be treated as Green's functions.

If we denote the Green's tensor, from the virtual source $\boldsymbol{\xi}$ to the receiver $\boldsymbol{r}$, as $G_{ij}(\boldsymbol{r}, t; \boldsymbol{\xi})$, this mode simulates the convolution of the spatial derivatives of Green's tensor with the source time function $s(t)$ as

$$
\begin{aligned}
G_i^{M1}(\boldsymbol{r}, t; \boldsymbol{\xi}) &\equiv \frac{\partial G_{ix}(\boldsymbol{r}, t; \boldsymbol{\xi})}{\partial \xi_x} * s(t) = \frac{\partial G_{ix}(\boldsymbol{\xi}, t; \boldsymbol{r})}{\partial \xi_x} * s(t) \\
G_i^{M2}(\boldsymbol{r}, t; \boldsymbol{\xi}) &\equiv \frac{\partial G_{iy}(\boldsymbol{r}, t; \boldsymbol{\xi})}{\partial \xi_y} * s(t) = \frac{\partial G_{iy}(\boldsymbol{\xi}, t; \boldsymbol{r})}{\partial \xi_y} * s(t) \\
G_i^{M3}(\boldsymbol{r}, t; \boldsymbol{\xi}) &\equiv \frac{\partial G_{iz}(\boldsymbol{r}, t; \boldsymbol{\xi})}{\partial \xi_z} * s(t) = \frac{\partial G_{iz}(\boldsymbol{\xi}, t; \boldsymbol{r})}{\partial \xi_z} * s(t) \\
G_i^{M4}(\boldsymbol{r}, t; \boldsymbol{\xi}) &\equiv \left(\frac{\partial G_{iy}(\boldsymbol{r}, t; \boldsymbol{\xi})}{\partial \xi_z} + \frac{\partial G_{iz}(\boldsymbol{r}, t; \boldsymbol{\xi})}{\partial \xi_y} \right) * s(t) = \left(\frac{\partial G_{iy}(\boldsymbol{\xi}, t; \boldsymbol{r})}{\partial \xi_z} + \frac{\partial G_{iz}(\boldsymbol{\xi}, t; \boldsymbol{r})}{\partial \xi_y} \right) * s(t) \\
G_i^{M5}(\boldsymbol{r}, t; \boldsymbol{\xi}) &\equiv \left(\frac{\partial G_{ix}(\boldsymbol{r}, t; \boldsymbol{\xi})}{\partial \xi_z} + \frac{\partial G_{iz}(\boldsymbol{r}, t; \boldsymbol{\xi})}{\partial \xi_x} \right) * s(t) = \left(\frac{\partial G_{ix}(\boldsymbol{\xi}, t; \boldsymbol{r})}{\partial \xi_z} + \frac{\partial G_{iz}(\boldsymbol{\xi}, t; \boldsymbol{r})}{\partial \xi_x} \right) * s(t) \\
G_i^{M6}(\boldsymbol{r}, t; \boldsymbol{\xi}) &\equiv \left(\frac{\partial G_{ix}(\boldsymbol{r}, t; \boldsymbol{\xi})}{\partial \xi_y} + \frac{\partial G_{iy}(\boldsymbol{\xi}, t; \boldsymbol{r})}{\partial \xi_x} \right) * s(t) = \left(\frac{\partial G_{ix}(\boldsymbol{r}, t; \boldsymbol{\xi})}{\partial \xi_y} + \frac{\partial G_{iy}(\boldsymbol{\xi}, t; \boldsymbol{r})}{\partial \xi_x} \right) * s(t),
\end{aligned}
\tag{2.16}
$$

which corresponds to the moment tensor response. Optionally, the body-force response

$$
\begin{aligned}
G_i^{B1}(\boldsymbol{r}, t; \boldsymbol{\xi}) &\equiv G_{ix}(\boldsymbol{r}, t; \boldsymbol{\xi}) * s(t) = G_{ix}(\boldsymbol{\xi}, t; \boldsymbol{r}) * s(t) \\
G_i^{B2}(\boldsymbol{r}, t; \boldsymbol{\xi}) &\equiv G_{iy}(\boldsymbol{r}, t; \boldsymbol{\xi}) * s(t) = G_{iy}(\boldsymbol{\xi}, t; \boldsymbol{r}) * s(t) \\
G_i^{B3}(\boldsymbol{r}, t; \boldsymbol{\xi}) &\equiv G_{iz}(\boldsymbol{r}, t; \boldsymbol{\xi}) * s(t) = G_{iz}(\boldsymbol{\xi}, t; \boldsymbol{r}) * s(t)
\end{aligned}
\tag{2.17}
$$

can be calculated.

To use this mode, the users should specify the station name `green_stnm` of the receiver. This station name should be contained in the station list file. `OpenSWPC` radiates the seismic wave by an excitation force with a direction specified by the `green_cmp` parameter and a source time function of the rise time, `green_trise`. To obtain the full response of all components, three independent simulations with `green_cmp='x'`, `'y'`, and `'z'` are necessary.

The virtual source location should be given in the Cartesian or geographical coordinates and depth (the format is described in Table 2.9) with unique integer ID numbers (`gid`). Multiple virtual source locations can be specified in the simulation. The `gid`s do not need to be sequential.

The output file is stored in the directory `(odir)/green/(gid)` in the `SAC` format with the name convention `(title)__(green_cmp)__mij__.sac` (for the moment tensor response) or `(title)__(green_cmp)__fi__.sac` (for the body force response).

Table 2.9: Virtual source location format for the reciprocity mode.

Type	Format			
'xyz'	x	y	z	gid
'llz'	lon	lat	z	gid

The amplitudes of the output files are multiplied by 10^9 to compare the SAC-formatted files in nm or nm/s units. The vertical component of the output file is changed to be positive upward. However, the derivative with respect to depth is performed according to the original definition of positive downward.

green_mode
: Flags to turn the reciprocity mode on. If this is `.true.`, the other earthquake source parameters will be ignored.

green_stnm
: Name of the virtual station. This name must be included in the station list.

green_cmp
: Component at the virtual receiver. Choose from 'x', 'y', or 'z'.

green_trise
: Rise time of the source time function convolved with the simulated Green's function.

green_bforce
: If `.true.`, calculate the body force response as well as the moment tensor response. The default setting is `.false.`.

green_fmt
: Format specification of the virtual source location. Choose from 'xyz' (Cartesian coordinate; default) or 'llz' (longitude, latitude, and depth).

green_maxdist
: The reciprocity wave will only be calculated if the horizontal distance is shorter than this parameter. Specify in units of km.

fn_glst
: Name of the virtual source location file.

stftype
: Source time function type. Same as the case with the moment tensor source.

ntdec_w
: Temporal decimation factor of the output waveforms. Same as the case with the normal waveform output.

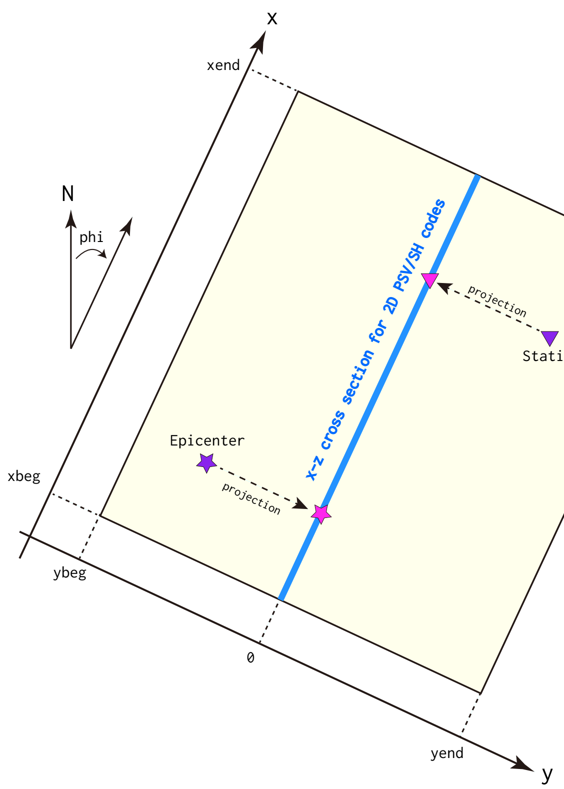

Figure 2.11: Cross section for the calculation in the 2D codes on the horizontal (x–y) plane. All stations and epicenters are projected onto the x–z cross section.

2.12 About Two-Dimensional Codes

OpenSWPC contains P-SV (swpc_psv) and SH (swpc_sh) codes, which work with the same parameter file. In these 2D codes, the simulation will be performed along the $x - -z$ cross section of $y = 0$. The parameters related to the y-direction will be omitted. The MPI partition will, therefore, be 1D, only in the x-direction. Note that all stations and sources outside the cross section will be projected onto the cross section, as schematically shown in Figure 2.11. For plane wave incidence, pw_strike and pw_rake will be fixed according to the type of code. Only the dip angle (pw_dip) can be changed.

2.13 Other Parameters

stopwatch_mode
: Measure the computation times at major subroutines and export the accumulated times to (odir)/(title).tim. This function is used for benchmarking and performance tuning.

`benchmark_mode`
: If this flag is `.true.`, the fixed homogeneous medium and single-point moment tensor source will be selected irrespective of the parameter specification. This is used for validation and performance measurements.

`ipad, jpad, kpad`
: Expand the Fortran array sizes along the x-, y-, and z-directions. In some computer architectures, the computation speed is very sensitive to the array size. In such cases, slightly changing the array size using these parameters may improve the performance. The expanded array will not be used for the simulation. Therefore, the simulation result is not affected by changing this option.

Chapter 3

Related Tools

3.1 Snapshot data handling

3.1.1 read_snp.x

Snapshot files in both NetCDF and the originally defined binary format can be extracted or visualized by the program read_snp.x.

```
read_snp.x -i snapfile [-h] [-ppm|-bmp] [-pall]
           [-mul var | -mul1 var -mul2 var ...] [-bin] [-asc] [-skip n]
```

-h

Print the header information defined in the snapshot, as in the following example.

```
> ../bin/read_snp.x -i swpc_3d.xz.ps.snp -h

[binary type]   : STREAMIO
[code type]     : SWPC_3D
[header version]:         3
[title]         : swpc_3d
[date generated]: 1408015126
                  2014-08-14T11-18-46
[coordinate]    : xz
[data type]     : ps
[ns1]           :       256
[ns2]           :       256
[beg1]          :      -63.87500
[beg2]          :       -9.87500
[ds1]           :        0.25000
[ds2]           :        0.25000
[dt]            :        0.05000
[na1]           :        20
[na2]           :        20
[nmed]          :         3
[nsnp]          :         2
[clon]          :      143.50000
[clat]          :       42.00000
```

`-ppm/-bmp`
: Visualize and export the image files in ppm or bmp format. The ppm or bmp directory will be automatically created in the current directory and image files with sequential numbers will be stored there. If the snapshot file is displacement or velocity, the absolute values of the vertical and horizontal amplitudes will be colored red and green, respectively. For the PS file, the absolute values of the divergence and rotation vector will be colored similarly. If the absolute value option is specified, the black-red-yellow-white color palette (similar to the "hot" color palette in GMT) will be adopted. For cross sections along the surface (ob, fs), the topography color map will be overlaid. For other cross sections, the velocity structure in the section will be overlaid.

`-pall`
: Visualize including the absorbing boundary region. This option works only if it is used with `-ppm/-bmp`. By default, the absorbing boundary region will be clipped.

`-mul var|-mul1 var -mul2 var ...`
: Multiply `var` by the amplitude for visualization. Adjust the visualized color by changing this value. Optionally, by specifying `-mul1` or `-mul2`, for example, one may change the weight of the amplitude by component.

`-abs`
: Visualize the absolute value of the vector. This only works with the velocity or displacement snapshots.

`-bin|asc`
: Export the snapshot data to binary (`-bin`) or ascii (`-asc`) files. The data file will be created in the automatically created `bin` or `asc` directories. The binary formatted data can be directly used in `GMT` with the `xyz2grd` module by appending the `-bis` option.

`-skip` *n*
: Skip the first *n* snapshots for visualization or data exports.

3.1.2 `diff_snp.x`

This program takes the difference between two snapshots and exports it to another snapshot file.

```
> diff_snp.x snap1 snap2 diffile
```

The output file format (`NetCDF` or binary) depends on the input file format.

3.2 Supporting Parameter Settings

3.2.1 `fdmcond.x`

The grid width in space and time in the finite difference method is controlled by the stability condition. The wavelength condition will affect the allowed maximum frequency radiated from the source.

The tool `fdmcond.x` can help determine these parameters to satisfy the conditions. After the user specifies several parameters, such as the grid width, maximum frequency (`fmax`), rise time (`Tr`), and minimum and maximum velocities in the medium (`vmin, vmax`), the program can suggest the other parameters.

Example

```

>  ./fdmcond.x

----------------------------------------------------------------------
                           FDM CONDITION
----------------------------------------------------------------------


 Model Dimension? --> 3
   2) 2D
   3) 3D


 Source Type? --> 3
   1) Triangle
   2) Herrmann
   3) Kupper


 Parameter Combination? --> 5
   1) dh   (space grid),  fmax (max freq.),  vmax (max vel.)
   2) dh   (space grid),  Tr   (rise time),  vmax (max vel.)
   3) dh   (space grid),  fmax (max freq.),  dt (time grid)
   4) dh   (space grid),  Tr   (rise time),  dt (time grid)
   5) dh   (space grid),  vmin (min vel.),   vmax (max vel.)
   6) dh   (space grid),  vmin (min vel.),   dt (time grid)
   7) fmax (max freq.) ,  vmax (max vel.),   dt (time grid)
   8) Tr   (rise time) ,  vmax (max vel.),   dt (time grid)
   9) vmin (min vel.)  ,  vmax (max vel.),   dt (time grid)


 Assumed Parameters:
   dx     =   0.25
   dy     =   0.25
   dz     =   0.25
   vmin   =   0.3
   vmax   =   8.0

 Derived Parameters:
   dt     <=   0.01546
   fmax   <=   0.17143
   Tr     >=  13.41667
```

3.2.2 `mapregion.x`

The geographical region of the simulation will be automatically determined by the parameters `clon`, `clat`, `phi`, `xbeg`, `ybeg`, `nx`, `ny`, `dx`, and `dy`. The `mapregion.x` program reads the parameter file and exports the outer edge of the region in longitude and latitude.

```
 > mapregion.x -i input.inf -o region.dat
```

If the option `-o` is omitted, the result will be printed to the standard output on the screen. This program will also estimate the total memory usage in the standard error output.

3.2.3 mapregion.gmt4, mapregion.gmt5

These scripts use `mapregion.x` to visualize the region by using GMT4 or GMT5. By default, these scripts plot only the region around the Japanese Islands.

3.3 Velocity Structure

3.3.1 qmodel_tau.x

Calculate the frequency dependence of Q^{-1} and the body wave dispersion from the input parameter file.

```
> qmodel_tau.x -nm [nm] -i [prm_file] -f0 [min_freq] -f1 [max_freq] -nf [ngrid]
```

This discretizes the frequency range from `min_freq` to `max_freq` into `ngrid` and exports $Q^{-1}(f)$ and physical dispersion. The latter is normalized to 1 at the reference frequency. The parameters related to the viscoelastic body are read from the input parameter file; however, the number of bodies `nm` should be specified separately because it is hard-coded into the program.

3.3.2 grdsnp.x

From the input parameter file, calculate and print the discontinuity of the input `NetCDF` file in Cartesian coordinates for the simulation (x, y, depth) in the standard output. This program is used to confirm the coordinate transformation and the detailed digital model, and to visualize the model in the computational domain.

```
> grdsnp.x -i [prm_file] -g [grd_file]
```

3.4 Generation of Random Media

3.4.1 gen_rmed3d.x

Generate a three-dimensional random medium file.

```
gen_rmed3d.x  [-o outfile] [-nx nx] [-ny ny] [-nz nz] [-epsil epsilon] [-kappa kappa]
[-dx dx] [-dy dy] [-dz dz] [-ax ax] [-ay ay] [-az az]  [-ptype ptype] {-seed seed_number}
```

`-o outfile`
: Name of the output file.

`-nx nx -ny ny -nz nz`
: Number of grids in the x-, y-, and z-directions. They must be a power of 2.

`epsil epsilon`
: Root mean square (RMS) of the velocity fluctuation ε.

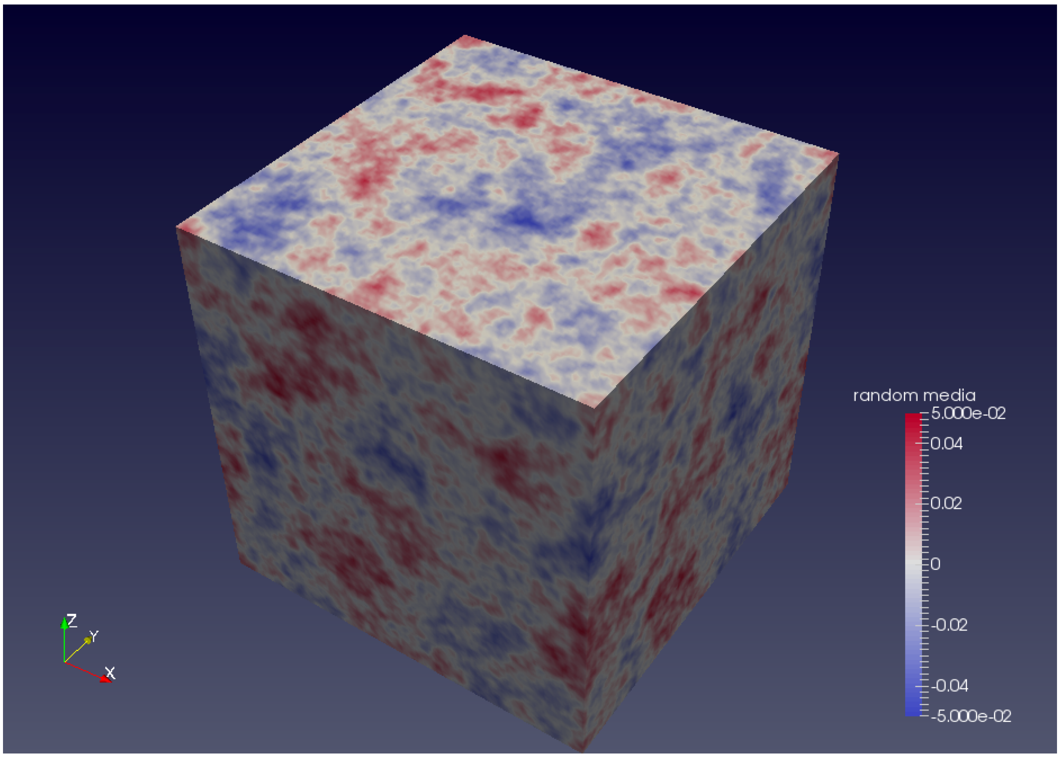

Figure 3.1: Example of the visualization of a 3D random medium using `ParaView`.

`-ax ax -ay ay -az az`
: Characteristic scales in the x-, y-, and z-directions in units of km.

`-dx dx -dy dy -dz dz`
: Grid width in the x-, y-, and z-directions. They should be identical to the simulation parameters.

`-ptype ptype`
: Choice of power spectrum density functions (PSDFs) of the random media model in wavenumber space: 1 for Gaussian, 2 for Exponential, and 3 for von Kármán.

`-kappa kappa`
: The parameter κ for the von Kármán-type PSDF.

`-seed seed_number`
: Specify the seed number of the random variable generation (optional). If this option is not specified, the seed number is automatically generated based on the execution date and time.

The random media file will be stored in the `NetCDF` format. Various software, such as `ParaView`[1] (Figure 3.1) and `Panoply`[2], can be used for visualization.

[1] http://www.paraview.org
[2] http://www.giss.nasa.gov/tools/panoply

3.4.2 gen_rmed2d.x

Generate a 2D random media file. Its usage is same as that of gen_rmed3d.x, with parameters related to the y-direction omitted.

3.5 Miscellaneous Tools

3.5.1 timvis scripts

Four scripts, timvis.gmt4, /timvis.gmt5, timvis_abs.gmt4, and /timvis_abs.gmt5, are used to visualize the elapsed time of the computation obtained with the input parameter stopwatch_mode = .true. by using GMT versions 4 and 5.

3.5.2 Geographic Coordinate Conversion

The Fortran programs ll2xy.x and xy2ll.x can project and inversely project the geographic and Cartesian coordinates with the same algorithm as OpenSWPC. These tools are provided for OpenSWPC version 3.0 or later.

Chapter 4

Additional Materials

4.1 Hints for Parameter Settings

The 3D simulation is bounded by the total memory size. The code requires

$$m_{\mathrm{MP}} = 116 + 24\mathtt{NM} = 188 \quad (\mathtt{NM} = 3) \quad \text{bytes} \tag{4.1}$$

of memory for the case of mixed precision (`MP=DP`) with a GNZ viscoelastic body of `NM=3`. Note that this is a coarse estimate excluding the effect of an absorbing boundary.

The computation time can be roughly estimated by the parameter $\mathtt{n_G}$, which is defined as the number of spatial and/or temporal grids that one CPU can process in a second. This value depends on the CPU, as shown in Table 4.1. The total computation time can be estimated by

$$t_{\mathrm{comp}} = \frac{\mathtt{nx} \times \mathtt{ny} \times \mathtt{nz}}{\mathtt{n_G} \times \mathtt{ncore}} \times \mathtt{nt} \quad [\mathrm{s}], \tag{4.2}$$

where `ncore` is the number of CPU cores used in the computation. If the estimated time exceeds that provided by the computer system, it is recommended to make the model size smaller and/or to use checkpointing/restarting.

4.2 Hints for Modifying the Code

4.2.1 Defining Your own Velocity Model

The velocity structure is defined by the subroutine `vmodel_*`, called by the module `m_medium.F90`. These subroutines commonly have the input/output parameters defined in Table 4.2. By creating a Fortran subroutine that returns the medium parameters `rho, lam, mu, qp` and

Table 4.1: Performance parameter $\mathtt{n_G}$.

Architecture Name	CPU	#core/CPU	$\mathtt{n_G}$
Mac Pro 2010	Xeon X5670 2.93GHz	6	6.7×10^6
EIC (ERI, UTokyko)	Xeon E5-2680 v3 2.5 GHz	12	7.0×10^6
The Earth Simulator (3rd gen.)	NEC SX-ACE	4	57×10^6

Table 4.2: Input/output specification of subroutines for velocity models.

Variable name	In/Out	Type	Description
`io_prm`	in	int	I/O number of the input parameter file
`i0,i1`	in	int	Start/end indices of arrays in x-direction
`j0,j1`	in	int	Start/end indices of arrays in y-direction
`k0,k1`	in	int	Start/end indices of arrays in z-direction
`xc(i0:i1)`	in	real	x grid locations
`yc(i0:i1)`	in	real	y grid locations
`zc(i0:i1)`	in	real	z grid locations
`vcut`	in	real	Cut-off velocity
`rho(k0:k1,i0:i1,j0:j1)`	out	real	Mass density [g/cm^3]
`lam(k0:k1,i0:i1,j0:j1)`	out	real	Lame coefficient λ [g/cm^3]
`mu(k0:k1,i0:i1,j0:j1)`	out	real	Lame coefficient μ [g/cm^3]
`qp(k0:k1,i0:i1,j0:j1)`	out	real	Q_P
`qs(k0:k1,i0:i1,j0:j1)`	out	real	Q_S
`bddep(i0:i1,j0:j1,0:NBD)`	out	real	Discontinuity boundary depths [km]

`qs` at locations given in the input of the subroutines `xc`, `yc`, and `zc`, it is easy to add a new velocity model.

The topography and bathymetry are automatically investigated in the `m_medium` module after calling the `vmodel_*` routine. To make this investigation work properly, the medium parameter `mu` must be zero in the air and ocean columns and `lam` must be zero in the air column.

The variables `bddep(:,:,0)` are assumed to be the topography, and are used for the snapshot output. The other values of `bddep(:,:,1:NBD)` are used to fit the source and/or station location to the discontinuity depths. Providing dummy values of these functions is not necessary.

4.2.2 Defining Your own Source Time Function

The source time function is called by the `source__momentrate` `Fortran` function in `m_source.F90` based on the choice of `stftype`. The definitions of the source time functions are given in `share/m_fdtool.F90`. It is easy to add a new source time function here and to add the call to the new function in the `m_source` module.

All of the pre-defined source time functions take two time parameters, `tbeg` and `trise`. In the source code, they are stored in the array variable `srcprm(:)`. If the new source time function requires more than three parameters, the user can expand the array `srcprm(:)` to store them.

4.2.3 Appending New Control Parameters

In many `Fortran` modules, the first set-up is performed by subroutines called `(modulename)__setup` during the first computation. Some of the setup modules read parameters from the input parameter file. These parameters are read by the subroutine `readini`, which is defined in `shared/m_readini.F90`.

Bibliography

Aki, K., and P. G. Richards (2002), *Quantitative Seismology: Theory and Methods*, 2nd eidtion ed., University Science Books.

Blanch, J. O., J. O. Robertsson, and W. W. Symes (1994), Modeling of a constant Q: methodology and algorithm for an efficient and optimally inexpensive viscoelastic technique, *Geophysics*, *60*, 176–184, doi:10.1111/j.1365-246X.2004.02300.x.

Cerjan, C., D. Kosloff, R. Kosloff, and M. Reshef (1985), A nonreflecting boundary condition for discrete acoustic and elastic wave equations, *Geophysics*, *50*(4), 705–708.

Koketsu, K., H. Miyake, and H. Suzuki (2012), Japan Integrated Velocity Structure Model Version 1, *Proceedings of the 15th World Conference on Earthquake Engineering*, p. Paper No.1773.

Maeda, T., and T. Furumura (2013), FDM Simulation of Seismic Waves, Ocean Acoustic Waves, and Tsunamis Based on Tsunami-Coupled Equations of Motion, *PAGEOPH*, *170*(1-2), 109–127, doi:10.1007/s00024-011-0430-z.

Maeda, T., T. Furumura, S. Noguchi, S. Takemura, S. Sakai, M. Shinohara, K. Iwai, and S.-J. Lee (2013), Seismic- and tsunami-wave propagation of the 2011 Off the Pacific Coast of Tohoku Earthquake as inferred from the tsunami-coupled finite-difference simulation, *Bulletin of the Seismological Society of America*, *103*(2B), 1456–1472.

Okamoto, T., and H. Takenaka (2005), Fluid-solid boundary implementation in the velocity-stress finite-difference method, *Zisin*, *57*, 355–364.

Robertsson, J. O., J. O. Blanch, and W. W. Symes (1994), Viscoelastic finite-difference modeling, *Geophysics*, *59*(9), 1444–1456, doi:10.1190/1.1443701.

Sato, H., M. C. Fehler, and T. Maeda (2012), *Seismic Wave Propagation and Scattering in the Heterogeneous Earth: Second Edition*, Springer Berlin Heidelberg, Berlin, Heidelberg, doi:10.1007/978-3-642-23029-5.

Zhang, W., and Y. Shen (2010), Unsplit complex frequency-shifted PML implementation using auxiliary differential equations for seismic wave modeling, *Geophysics*, *75*(4), T141–T154, doi:10.1190/1.3463431.

Acknowledgments

This project was supported by the Collaborative Research Program of the Earthquake Research Institute at the University of Tokyo (2015-B-01), the Core-to-Core Collaborative Research Program of the Earthquake Research Institute at the University of Tokyo, and the Disaster Prevention Research Institute at Kyoto University (2016-K-06). The code was developed through research collaborations with Takashi Furumura, Shunsuke Takemura, Masaru Todoriki, Futoshi Mori, Nana Yoshimitsu, Hiroyuki Kumagai, Hanae Morioka, Aitaro Kato, Issei Doi, Nozomi Kanaya, and Takehi Isse. The author would like to thank Enago for the English language review.

Revision History

2015-06-04 First closed version for the ERI/UT joint usage program.

2015-06-10 Revision for the new Earth Simulator.

2015-06-29 Added random media.

2015-07-14 MPI/OpenMP hybrid parallel simulation mode.

2015-12-04 Text revision.

2016-01-14 Body force and reciprocity modes.

2016-02-03 Output in `NetCDF` format.

2016-05-05 (v1.0) Official open-source release,

2016-06-19 (v2.0) Hybrid parallel simulation for 2D codes.

2016-08-21 (v3.0) Improved reciprocity mode, geographic projection tools, and csf waveform format.

2017-09-21 (v4.0) Minor bugfixes, new binary output for waveform, updated references.

Copyright & License

This software is provided under the MIT license.

Copyright (c) 2013-2017 Takuto Maeda

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF, OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

The author requests that the user cite (at least one of) the following papers in any publications that result from the use of this software, although this is not an obligation.

Accompanying Paper

- Maeda, T., S. Takemura, and T. Maeda (2017), OpenSWPC: An open-source integrated parallel simulation code for modeling seismic wave propagation in 3D heterogeneous viscoelastic media, *Earth Planets Space*, *69*, 102, doi:10.1186/s40623-017-0687-2.

Related Papers

- Furumura, T. and L. Chen (2004), Large scale parallel simulation and visualization of 3D seismic wavefield using the Earth Simulator, *Computer Modeling of Engineering and Sciences*, *6*(2), 153-168.

- Furumura, T. and L. Chen (2005), Parallel simulation of strong ground motions during recent and historical damaging earthquakes in Tokyo, Japan, *Parallel Computing*, *31*, 149-165.

- Furumura, T. Hayakawa, M. Nakamura, K. Koketsu, and T. Baba (2008), Development of long-period ground motions from the Nankai Trough, Japan, earthquakes: Observations and computer simulation of the 1944 Tonankai (Mw8.1) and the 2004 SE Off-Kii Peninsula (Mw7) Earthquakes, *Pure Appl. Geophys.*, *165*, 585-607.

- Furumura, T. and T. Saito (2009), An integrated simulation of ground motion and tsunami for the 1944 Tonankai earthquake using high-performance super computers, *J. Disast. Res.*, *4*(2), 118-126.

- Maeda, T., and T. Furumura (2013), FDM simulation of seismic waves, ocean acoustic waves, and tsunamis based on tsunami-coupled equations of motion, *Pure Appl. Geophys.*, *170*(1-2), 109-127, doi:10.1007/s00024-011-0430-z.

- Noguchi, S., T. Maeda, and T. Furumura (2013), FDM simulation of an anomalous later phase from the Japan Trench subduction zone earthquakes, *Pure Appl. Geophys.*, *170*(1-2), 95-108, doi:10.1007/s00024-011-0412-1.

- Maeda, T., T. Furumura, S. Noguchi, S. Takemura, S. Sakai, M. Shinohara, K. Iwai, S. J. Lee (2013), Seismic and tsunami wave propagation of the 2011 Off the Pacific Coast of Tohoku Earthquake as inferred from the tsunami-coupled finite difference simulation, *Bull. Seism. Soc. Am.*, *103*(2b), 1456-1472, doi:10.1785/0120120118.

- Maeda, T., T. Furumura, and K. Obara (2014), Scattering of teleseismic P-waves by the Japan Trench: A significant effect of reverberation in the seawater column, *Earth Planet. Sci. Lett.*, *397*(1), 101-110, doi:10.1016/j.epsl.2014.04.037.

- Noguchi, S., T. Maeda, and T. Furumura (2016), Ocean-influenced Rayleigh waves from outer-rise earthquakes and their effects on durations of long-period ground motion, *Geophys. J. Int.*, *205*(*2*), 1099-1107, doi:10.1093/gji/ggw074.

- Takemura, S., T. Maeda, T. Furumura, and K. Obara (2016), Constraining the source location of the 30 May 2015 (Mw 7.9) Bonin deep-focus earthquake using seismogram envelopes of high-frequency P waveforms: occurrence of deep-focus earthquake at the bottom of a subducting slab, *Geophys. Res. Lett.*, *43*, 4297-4302, doi:10.1002/2016GL068437.

- Yoshimitsu, N., T. Furumura, and T. Maeda (2016), Geometric effect on a laboratory-scale wavefield inferred from a three-dimensional numerical simulation, *J. Appl. Geophys.*, *132*, 184-192, doi:10.1016/j.jappgeo.2016.07.002.
  (A computational study on seismic wave propagation in the scale of $\sim$ 10 cm)

- Maeda, T., K. Nishida, R. Takagi, and K. Obara (2016), Reconstruction of a 2D seismic wavefield by seismic gradiometry, *Prog. Earth Planet. Sci.*, *3*, 31. doi:10.1186/s40645-016-0107-4.
  (A proposal of a seismic wavefield analysis, with verification via a numerical simulation)

- Todoriki, M., T. Furumura, and T. Maeda (2017), Effects of seawater on elongated duration of ground motion as well as variation in its amplitude for offshore earthquakes, *Geophys. J. Int.*, *208*, 226-233, doi:10.1093/gji/ggw388.

- Toya, M., A. Kato, T. Maeda, K. Obara, T. Takeda, and K. Yamaoka (2017), Down-dip variations in a subducting low-velocity zone linked to episodic tremor and slip: a new constraint from *ScSp* waves, *Scientific Reports*, *7*, 2868, doi:10.1038/s41598-017-03048-6.

- Morioka, H., H. Kumagai, and T. Maeda (2017), Theoretical basis of the amplitude source location method for volcano-seismic signals, *J. Geophys. Res.*, *122*, 6538-6551. doi:10.1002/2017JB013997.