USER GUIDE

(version March 6, 2014)

FDSim3D

The Fortran95 Code
for Numerical Simulation
of Seismic Wave Propagation
in 3D Heterogeneous Viscoelastic Media

by

Jozef KRISTEK and Peter MOCZO

Faculty of Mathematics, Physics and Informatics
Comenius University in Bratislava
Mlynska dolina F1
842 48 Bratislava
Slovak Republic

kristek@fmph.uniba.sk, moczo@fmph.uniba.sk

Purpose: The program FDSim3D is designed for computation of seismic wavefields in 3D heterogeneous surface geological structures with a planar free surface. A wavefield can be excited by point double-couple sources or by a vertically incident plane wave.

Terms of use: In any publication in which a user includes results obtained with the computer code, reference has to be made to

Kristek, J., Moczo, P., 2014. FDSim3D - The Fortran95 Code for Numerical Simulation of Seismic Wave Propagation in 3D Heterogeneous Viscoelastic Media. www.cambridge.org/moczo

and

Moczo, P., Kristek, J., Gális, M. 2014. The Finite-Difference Modelling of Earthquake Motions: Waves and Ruptures. *Cambridge University Press*.

CONTENTS

1 Introduction

Program FDSim3D is designed for the finite-difference (FD) simulation of seismic wave propagation and earthquake ground motion in 3D local surface heterogeneous viscoelastic structures with a planar free surface.

The computational algorithm is based on the explicit heterogeneous FD scheme solving equations of motion for the heterogeneous viscoelastic medium with material interfaces. The velocity-stress FD scheme is 4th-order accurate in space and 2nd-order accurate in time. The scheme is constructed on a staggered finite-difference grid.

The computational region is a volume of a parallelepiped with the top side representing a planar free surface, and bottom, rear, front, left and right sides representing either rigid boundaries or non-reflecting boundaries. The perfectly matched layer (PML) can be optionally combined with one of several types of the nonreflecting boundaries.

The discontinuous spatial grid is used to cover the computational region. The upper part of the grid has REFINE times smaller grid spacing than the lower part, where REFINE is an arbitrary odd number. Each part of the grid is a uniform Cartesian grid.

The rheology of the medium corresponds to the generalized Maxwell body in a definition equivalent to the generalized Zener body. This makes it possible to account both for spatially varying quality factors of the P and S waves and for arbitrary $Q(\omega)$ law.

A wavefield is excited either by a set of double-couple point sources or by a vertically impinging plane wave.

A computational grid model can be prepared using the computer code FDMod3D. The FD simulation is performed using the computer code FDSim3D. Both codes can be executed in a serial or parallel mode.

In the following description, abbreviation MKG2014 is used for the book

Moczo, P., Kristek, J., Gális, M. 2014. The Finite-Difference Modelling of Earthquake Motions: Waves and Ruptures. *Cambridge University Press*.

FDSim3D cannot be used as a black-box tool – similarly as other complicated seismological codes. A user is strongly advised to learn at least necessary basics of the finite-difference method.

2 Compilation

The FORTRAN compiler is necessary for the successful compilation. The minimum level is FORTRAN 95. The `fpp` FORTRAN pre-processor is necessary because the source code uses a few macros for conditional compilation. The conditional compilation makes it possible to choose either serial or parallel version of the executable program. No non-standard library is required.

Because the source code consists of 126 files, the compilation is performed using the `make` utility. The `Make` utility automatically builds an executable (target) program from the source code by reading a file called Makefile. Makefile specifies building of the executable program. Therefore, it is necessary to edit the provided Makefile before the first compilation. Examples how to edit the Makefile are given in the next subsections.

The building of executable program is invoked by statement `make` or `make –f Makefile`. After successful compilation and linking the executable program `FDSim3D` should be created.

2.1 Serial program

The only part which should be edited in the Makefile, is the part defining the FORTRAN compiler and flags for compilation and linking. In the Makefile there are these three lines:

```
FC = ifort
FFLAGS = -fpp –Ofast
LDFLAGS = -Ofast
```

This example is valid for the Intel FORTRAN Compiler `ifort`. In case of other compiler, the compiler name should be assigned to variable FC, compilation flags to variable FFLAGS and linking flags to LDFLAGS. The better performance could be achieved using highly optimized compilation (flags like –O3 or –Ofast), but the executable program should be always checked against a low-level optimized version.

2.2 Parallel (MPI) program

The source code uses the Message Passing Interface libraries for parallelisation. A suitable implementation of MPI should be installed. The code was tested with OpenMPI and MPICH but all implementations with standard MPI 1.2 should work.

The only part, which should be edited in the Makefile, is the part defining the FORTRAN compiler and flags for compilation and linking. In the Makefile there are these three lines:

```
FC = mpif90
FFLAGS = -fpp –Ofast –DUSE_MPI
LDFLAGS = -Ofast
```

This example is valid for an MPI implementation compiled with the Intel FORTRAN Compiler `ifort`. In case of other compiler, the MPI compiler name should be assigned to variable FC, compilation flags to variable FFLAGS and linking flags to LDFLAGS. The better performance could be achieved using highly optimized compilation (flags like –O3 or –Ofast), but the executable program should be always checked against low-level optimized version. The flag –DUSE_MPI defines macro which switches on the parallel version of the code.

If the MPI implementation requires use of statement USE MPI instead of statement INCLUDE 'mpif.h', then it is necessary to define also macro –DMPI2. The Makefile then reads:

```
FC = mpif90
FFLAGS = -fpp –Ofast –DUSE_MPI –DMPI2
LDFLAGS = -Ofast
```

Please, contact your system administrator for advice and help if necessary.

3 Data files

FDSim3D requires two types of input data files:

- Manually prepared input data files
    - ➢ an auxiliary file containing only the name of the current computation
    - ➢ input data file with the first set of controlling parameters for the computation
    - ➢ input data file containing names and positions of the receivers
    - ➢ input data file containing parameters specifying a type of the wavefield excitation
    - ➢ one or several input data files containing the source time function(s)

- Input data files created by the model preparation code FDMod3D
    - ➢ input data file with the second set of controlling parameters for the computation
    - ➢ input data file containing elastic parameters and densities describing types of material cells
    - ➢ input data file containing anelastic parameters describing types of material cells
    - ➢ one or two input data files containing spatial distribution of material cell types

FDSim3D generates several output data files depending on the input parameters:

- ➢ a log file containing the input parameters (as read by the program) and error messages
- ➢ a file containing particle-velocity values at specified receivers
- ➢ a file containing particle-velocity values at specified horizontal grid planes
- ➢ a file containing values of the total energy in the computational model

Note
In the following description of the input data *default* means that the variables need not to be specified.

3.1 Manually prepared input data files

Manually prepared input data files are

- an auxiliary file containing only the name of the current computation
- input data file with the first set of controlling parameters for the computation
- input data file containing names and positions of the receivers
- input data file containing parameters specifying a type of the wavefield excitation
- one or several input data files containing the source time function(s)

3.1.1 HF_FDSim3D

Auxiliary input data file. The file type is ASCII and consists of the following variable:

Name of Variable	*Type*	*Description*
JOBNAME	A17	The name of the current computation. This name is taken as a basis for constructing (by the program) names of other input and output files.

3.1.2 ***JOBNAME*.IN**

Input data file specifying several controlling variables. The file type is ASCII and consists of the following variables:

NAMELIST /CONTROLDATA/ TPML, PROCX, PROCY, PROCZ, STRESS_IMAGING

Name of Variable	*Type*	*Description*
TPML	real	The thickness of the perfectly matched layer (PML). If set to zero then no PML is assumed. *default:* TPML=0
PROCX, PROCY, PROCZ	real	The numbers of subregions of the whole computational region in the x- (easting), y- (northing) and z- (vertical) directions, respectively. Each subregion is processed by one process during a parallel computation. PROCX * PROCY * PROCZ has to be equal to the required number of processes during the run of the code. These parameters are not applicable during a serial computation.

Name of Variable	*Type*	*Description*
STRESS_IMAGING	logical	.TRUE.: The free-surface condition is approximated by the stress-imaging technique. .FALSE.: The free-surface condition is approximated by the AFDA technique. *default:* STRESS_IMAGING = .FALSE.

NAMELIST /CONTROLDATA1/ TW, DT, R, TAU , POW, SF, WC, TAU_EPS

Name of Variable	*Type*	*Description*
TW	real	The time window for the computation in seconds.
DT	real	Time step Δt in seconds. It has to satisfy the stability condition for the (2,4) staggered-grid VS FD scheme, i.e. $\Delta t \leq \frac{6}{7\sqrt{3}} \Upsilon$. Here Υ is the minimum ratio $\frac{h}{v}$ in the model, where v is the local P-wave velocity and h local grid spacing.
R, TAU, POW, SF, WC	real	The parameters of PML. *default:* R = 0.001, TAU = 1.5, POW = 2, SF = 0, WC = 0
TAU_EPS	real	The parameter of Rayleigh damping. *default:* TAU_EPS = 0. (no Rayleigh damping)

NAMELIST /NAMES_SR_REC/ SR_FILE_NAME, REC_FILE_NAME

Name of Variable	*Type*	*Description*
SR_FILE_NAME	A20	The name of the ASCII file containing the information on the source(s). See subsection 3.1.3
REC_FILE_NAME	A20	The name of the ASCII file containing the position of the receivers. See subsection 3.1.6

NAMELIST /KEYS/ KEY_TLV, KEY_SNV, KEY_EN

Name of Variable	*Type*	*Description*
KEY_TLV	logical	.TRUE.: Output files in ASCII format are generated. They contain values of particle velocities at specified receivers. See subsection 4.1.2. .FALSE.: The output files are not generated. *default:* KEY_TLV = .FALSE.
KEY_SNV	logical	.TRUE.: Output files in binary format are generated. They contain snapshots of the particle velocities in specified grid planes. See subsection 4.1.3. .FALSE.: The output files are not generated. *default:* KEY_SNV = .FALSE.
KEY_EN	logical	.TRUE.: Output file “ENERGY.DAT” in ASCII format is generated. The file contains values of total energy in the whole computational model. This is especially suitable for testing and checking performance of PML. See subsection 4.1.4. .FALSE.: The output file is not generated. *default:* KEY_EN = .FALSE.

NAMELIST /NONREF/ OMG, WB, KTLE, KTRI, KTFR, KTRE, KTBO

Name of Variable	*Type*	*Description*
OMG	real	Dominant frequency in Hz at which artificial reflections should be suppressed more than at other frequencies. Applicable in case of the Higdon, Peng & Toksöz and Liu-Archuleta boundaries. *default:* OMG = FREF
WB	real	Weight coefficient b for the Higdon and Liu-Archuleta types of nonreflecting boundary. It has to satisfy condition $0 \le b \le 0.4$. *default:* WB = 0.4
KTLE, KTRI, KTRE, KTFR, KTBO	integer	Key determining type of the left, right, rear, front and bottom boundary of the grid: = 0: rigid boundary = 1: Higdon

Name of Variable	*Type*	*Description*
		= 2: Reynolds = 3: Peng & Toksöz, maximum attenuation set for the P waves for perpendicular particle-velocity components = 4: Emmerman & Stephen, max. attenuation set for the P waves for perpendicular particle-velocity components = 5: Clayton & Engquist A1 = 6: Liu-Archuleta (original) = 7: Liu-Archuleta (maximum attenuation set for the P waves for perpendicular particle-velocity components) = 8: Liu-Archuleta with a better approximation of the A1 conditions of Clayton & Engquist = 9: Peng & Toksöz, maximum attenuation set for the S waves for all particle-velocity components

NAMELIST /SNAP/ IPAS2, MSNP

Included only if KEY_SNV = .TRUE.

Name of Variable	*Type*	*Description*
IPAS2	integer	If IPAS2 = 1, then the particle-velocity values at each time level are stored. If IPAS2 = 2 (3,...), then the particle-velocity values at each second (third,...) time level are stored.
MSNP	integer	The number of planes for which snapshots are stored.

MSNP integer numbers at the end of the file specify grid indices of the horizontal planes for which snapshots are stored. Index 0 means the free surface. The numbers are given in the free format.

NAMELIST /TXT/ TEXT

Name of Variable	*Type*	*Description*
TEXT	A20	An arbitrary alphanumeric text (e.g., describing the computation).

NAMELIST /REC/ MR

Name of Variable	*Type*	*Description*
MR	integer	The number of receivers.

3.1.3 ***SR_FILE_NAME***

Input data file specifying parameters of source(s). The file type is ASCII and consists of the following variables:

NAMELIST /SOURCE/ NPS, KEY_ONE_STF, PWINC

Name of Variable	*Type*	*Description*
NPS	integer	The number of point sources. In case of PWINC > 0, NPS could have an arbitrary value.
KEY_ONE_STF	logical	.TRUE. – only one source-time function is used for all point sources, i.e. only file SRC_001.DAT is read in and used. .FALSE. – each point source has different source-time function, i.e. files SRC_001.DAT, SRC_002.DAT, … SRC_nps.DAT are read in and used. In case of PWINC > 0, KEY_ONE_STF could have an arbitrary value.
PWINC	Integer	Key determining type of the source. = 0: double-couple source(s) = 1: impinging plane S wave = 2: impinging plane P wave NOTE: The plane wave incidence is not possible if a discontinuous grid is used.

The other data is read in depending on the value of PWINC:

In case of PWINC = 0 (Double-couple source(s))

```
DO I = 1, NPS
  READ (10,*) XS(I), YS(I), ZS(I), TB(I), TE(I)
  READ (10,*) SFIS(I), SDEL(I), SLAM(I), M0(I)
END DO
```

Name of Variable	*Type*	*Description*
XS(I)	real	The x-coordinate (easting) of the I-th source in meters.
YS(I)	real	The y-coordinate (northing) of the I-th source in meters.
ZS(I)	real	The z-coordinate (vertical) of the I-th source in meters. (0 for sources at the free surface, but for the methodological reasons the sources at the free surface will be treated as sources at the depth of 1.5 times the grid spacing)
TB(I)	real	The start time of the I-th source.
TE(I)	real	The end time of the I-th source.
SFIS	real	The strike of the I-th source (in degrees). (The y-coordinate is northing).
SDEL	real	The dip of the I-th source (in degrees).
SLAM	real	The rake of the I-th source (in degrees).
M0	real	The scalar seismic moment of the I-th source (in Nm).

In case of PWINC > 0 (plane wave incidence)

```
READ (10,*) DPW
IF ( PWINC == 1 ) READ (10,*) ANGLE
```

Name of Variable	*Type*	*Description*
DPW	integer	Depth (in meters) at which the plane wave is excited using Alterman & Karal (1968) approach.
ANGLE	real	The ANGLE (in degrees) defines the polarization of S wave. ANGLE = 0° means S wave polarized in the north-south direction, ANGLE = 90° means S wave polarized in the east-west direction.

3.1.4 **SRC_*xxx*.DAT**

Input data file(s) specifying the **source time function of the double couple source(s)**. The file type is ASCII. The file consists of two columns: the first column is time in seconds, the second column is the normalized slip (usually some kind of a ramp function starting from zero and ending at value 1.)

The file SRC_001.DAT contains the source-time function of the first source, SRC_002.DAT the source-time function of the second source, etc.

If KEY_ONE_STF = .FALSE. then the number of SRC_xxx.DAT files should be the same as the value of NPS; otherwise only one SRC_001.DAT file is assumed for all sources.

3.1.5 **PW_STF.DAT**

Input data file specifying the **source-time function of a plane wave** impinging from depth DPW. The file type is ASCII and consists of two columns: the first column is time in seconds, the second column is the particle velocity (in m/s).

3.1.6 ***REC_FILE_NAME***

Input data file specifying positions of receivers. The file type is ASCII and consists of four columns. The first column is an alphanumeric name (6 characters); the alphanumeric name will make the first part of the output data file with the particle-velocity components. The second column is the x-coordinate (easting) of a receiver in meters, the third column is the y-coordinate (northing) of the receiver in meters, and the fourth column is the vertical coordinate of the receiver in meters (0 for receivers at the free surface, positive for borehole receivers).

3.2 Input data files created by the model preparation code FDMod3D

Input data files created by the model preparation code FDMod3D are

- input data file with the second set of controlling parameters for the computation
- input data file containing elastic parameters and densities describing types of material cells
- input data file containing anelastic parameters describing types of material cells
- one or two input data files containing spatial distribution of material cell types

3.2.1 ***JOBNAME*.INM**

Input data file specifying several controlling variables. The file type is ASCII and consists of the following variables:

NAMELIST /FILENAMES/ JMH_FILE_NAME, JMHF_FILE_NAME, MO_FILE_NAME, Q_FILE_NAME

Name of Variable	*Type*	*Description*
JMH_FILE_NAME	A20	Name of the file containing spatial distribution of material cell types in the coarser spatial grid.
JMHF_FILE_NAME	A20	Name of the file containing spatial distribution of material cell types in the finer spatial grid.
MO_FILE_NAME	A20	Name of the file containing elastic parameters and densities describing types of material cells.
Q_FILE_NAME	A20	Name of the file containing anelastic coefficients describing types of material cells.

NAMELIST /CONTROLDATA2/ MX, MY, MZ, LPAS, H, XBMIN, YBMIN, REFINE

Name of Variable	*Type*	*Description*
MX	integer	The total numbers of the grid cells in the **y-direction (northing)** in coarser grid. (The corresponding total numbers of the grid spacings is MX-1.)

Name of Variable	*Type*	*Description*
MY	integer	The total numbers of the grid cells in the **x-direction (easting)** in coarser grid. (The corresponding total numbers of the grid spacings is MY-1.) Note: MX for the y-direction and MY for the x-direction are correct – the notation has a “historic” reason.
MZ	integer	The total number of the horizontal grid planes (finer and coarser grids together).
LPAS	integer	The number of the horizontal grid planes of the finer grid. If LPAS = 0 then the computational grid consists only of the coarser grid. *default:* LPAS = 0
H	real	The spatial grid spacing in the coarser grid in meters.
XBMIN	real	x-coordinate (easting) of the left boundary of the computational model in meters. *default:* XBMIN = 0
YBMIN	real	y-coordinate (northing) of the front boundary of the computational model in meters. *default:* YBMIN = 0
REFINE	integer	The ratio between the grid spacings in the coarser and finer grids. *default:* REFINE = 1

NAMELIST /ATTEN/ FRJMIN, FRJMAX, FRANGE, FREF, N_FREQ

Name of Variable	*Type*	*Description*
FRJMIN	real	The lower limit of the frequency range in which the discrete grid model should correspond to the desired $Q(\omega)$ law. The value is in Hz. *default:* FRJMIN is determined from FRANGE
FRJMAX	real	The upper limit of the frequency range in which the discrete grid model should correspond to the desired $Q(\omega)$ law. It should be larger (less than one order) than the maximum

Name of Variable	*Type*	*Description*
		frequency f_{AC} up to which the computation should be sufficiently accurate. The value is in Hz.
FRANGE	real	This variable determines the frequency range for the attenuation in which the discrete grid model should correspond to the desired $Q(\omega)$ law. FRANGE = 3, e.g., means frequency range $\left\langle \text{FRJMAX} * 10^{-3}, \text{FRJMAX} \right\rangle$. The value is ignored if FRJMIN > 0
FREF	real	The reference frequency at which the values of wave speeds were prescribed for the grid model. The value is in Hz.
N_FREQ	real	The number of relaxation frequencies. It should be less or equal to 4. *default:* N_FREQ = 4

3.2.2 ***MO_FILE_NAME***

Input data file specifying the elastic parameters and densities describing types of material cells. The file type is binary. The data is read by

```
    READ ( 14 ) JMNUM
```

Name of Variable	*Type*	*Description*
JMNUM	integer	The number of material cell types.

```
     READ ( 14 )                                                &
          ( DENU (JM1), DENV (JM1), DENW (JM1),                 &
            L2MX (JM1), L2MY (JM1), L2MZ (JM1),                 &
            LAXY (JM1), LAXZ (JM1), LAYZ (JM1),                 &
            MUXY (JM1), MUXZ (JM1), MUYZ (JM1), JM1 = 1, JMNUM)
```

Name of Variable	*Type*	*Description*
DENU, DENV, DENW	real	The volume arithmetic average of the density [kg/m^3] at the grid position of the NS-, EW-, and vertical component of the particle velocity.
$\begin{pmatrix} \text{L2MX} & \text{LAXY} & \text{LAXZ} & 0 & 0 & 0 \\ \text{LAXY} & \text{L2MY} & \text{LAYZ} & 0 & 0 & 0 \\ \text{LAXZ} & \text{LAYZ} & \text{L2MZ} & 0 & 0 & 0 \\ 0 & 0 & 0 & \text{MUXY} & 0 & 0 \\ 0 & 0 & 0 & 0 & \text{MUXZ} & 0 \\ 0 & 0 & 0 & 0 & 0 & \text{MUYZ} \end{pmatrix}$	real	The effective material parameters (elasticity matrix for orthorhombic materials) in [Pa]

3.2.3 *Q_FILE_NAME*

Input data file specifying the anelastic parameters describing types of material cells. The file type is binary. The data is read by

```
READ ( 15 )                                                      &
       ( YL2MX (JM1, 1:4), YL2MY (JM1, 1:4), YL2MZ (JM1, 1:4),   &
         YLAXY (JM1, 1:4), YLAXZ (JM1, 1:4), YLAYZ (JM1, 1:4),   &
         YMUXY (JM1, 1:4), YMUXZ (JM1, 1:4), YMUYZ (JM1, 1:4),   &
                                                  JM1 = 1, JMNUM)
```

Name of Variable	*Type*	*Description*
$\begin{pmatrix} \text{YL2MX} & \text{YLAXY} & \text{YLAXZ} & 0 & 0 & 0 \\ \text{YLAXY} & \text{YL2MY} & \text{YLAYZ} & 0 & 0 & 0 \\ \text{YLAXZ} & \text{YLAYZ} & \text{YL2MZ} & 0 & 0 & 0 \\ 0 & 0 & 0 & \text{YMUXY} & 0 & 0 \\ 0 & 0 & 0 & 0 & \text{YMUXZ} & 0 \\ 0 & 0 & 0 & 0 & 0 & \text{YMUYZ} \end{pmatrix}$	real	The anelastic coefficients corresponding to effective material parameters.

3.2.4 ***JMH_FILE_NAME***

Input data file specifying spatial distribution of the material cell types in the **coarser** spatial grid whose grid spacing is H. The file type is binary. The data is read by

```
DO L = LPAS, MZ
  READ (13)  JM( 1:MX,1:MY, L)
END DO
```

Name of Variable	*Type*	*Description*
JM (I, K, L)	integer	The integer number specifying type of block in the (I,K,L)-th grid cell.

3.2.5 ***JMHF_FILE_NAME***

Input data file specifying spatial distribution of material cell types in the **finer** spatial grid whose grid spacing is H / REFINE. The file type is binary. The data is read by

```
MXF = (MX-1)*REFINE + 1
MYF = (MY-1)*REFINE + 1

DO L = 0, LPAS
  READ (33) JMF( 1:MXF,1:MYF, L)
END DO
```

Name of Variable	*Type*	*Description*
JMF (I, K, L)	integer	The integer number specifying type of block in the (I,K,L)-th grid cell.

3.3 Output data files

FDSim3D generates several output data files depending on the input parameters:

- a log file containing the input parameters (as read by the program) and error messages
- a file containing particle-velocity values at specified receivers in the ASCII format if KEY_TLV = .TRUE.
- a file containing particle-velocity values at specified horizontal grid planes in the binary format if KEY_SNV = .TRUE.
- a file containing values of the total energy in the computational model in the ASCII format if KEY_EN = .TRUE.

3.3.1 ***JOBNAME*.LOG**

Log file containing the input parameters read from the input data files. In case of certain error(s) in the computation the file also contains the error message. The file type is ASCII.

3.3.2 ***REC_NAME(J)*.asc**

Output data file(s) containing times and values of particle-velocity components The files are generated only if KEY_TLV = .TRUE. A filename consists of the alphanumeric name of the *J*th receiver (see subsection 3.1.6). The file type is ASCII and consists of four columns:

Time | *NS-component* | *EW-component* | *Vertical (up-down) component*

3.3.3 **SNAPppp_Vttttt.DAT**

Output data file containing values of the particle velocity at the specified horizontal grid plane **ppp** at the **ttttt** time level. The files are generated only if KEY_SNV = .TRUE. The file type is binary. If the grid plane **ppp** is in the finer grid, the file is written by

```
      TPMLF = TPML * REFINE

      WRITE ( 18 ) U(1-TPMLF:MXF+TPMLF,1-TPMLF:MYF+TPMLF,ppp), &
                   V(1-TPMLF:MXF+TPMLF,1-TPMLF:MYF+TPMLF,ppp), &
                   W(1-TPMLF:MXF+TPMLF,1-TPMLF:MYF+TPMLF,ppp)
```

f the grid plane **ppp** is in the coarser grid, the file is written by

```
      WRITE ( 18 ) U(1-TPML:MX+TPML,1-TPML:MY+TPML,ppp),          &
                   V(1-TPML:MX+TPML,1-TPML:MY+TPML,ppp),          &
                   W(1-TPML:MX+TPML,1-TPML:MY+TPML,ppp)
```

U, V and W are the NS, EW and vertical UD particle-velocity components.

3.3.4 ENERGY.DAT

Output data file containing time and value of total energy in the computational model. The file is generated only if KEY_EN = .TRUE. The file type is ASCII and consists of two columns:

Time | *Total energy*

4 Model preparation code FDMod3D

The FDMod3D is designed to generate an FD-grid model of a medium. The output data files make the input data files for the finite-difference program FDSim3D.

The coordinate system is Cartesian with x-coordinate easting, y-coordinate northing and z-coordinate positive upward. The free surface is set to z=0 m.

The medium is divided into 2 parts (see Figure 1)

- **3D heterogeneous** part depicted in the figure as **Sediments.** Material parameters can vary in all directions and several (NI) surfaces (material interfaces where material parameters change discontinuously) can be prescribed.
- **1D vertically heterogeneous** part depicted in the figure as **Rock** and **Bedrock.** Material parameters can vary only in the vertical direction.

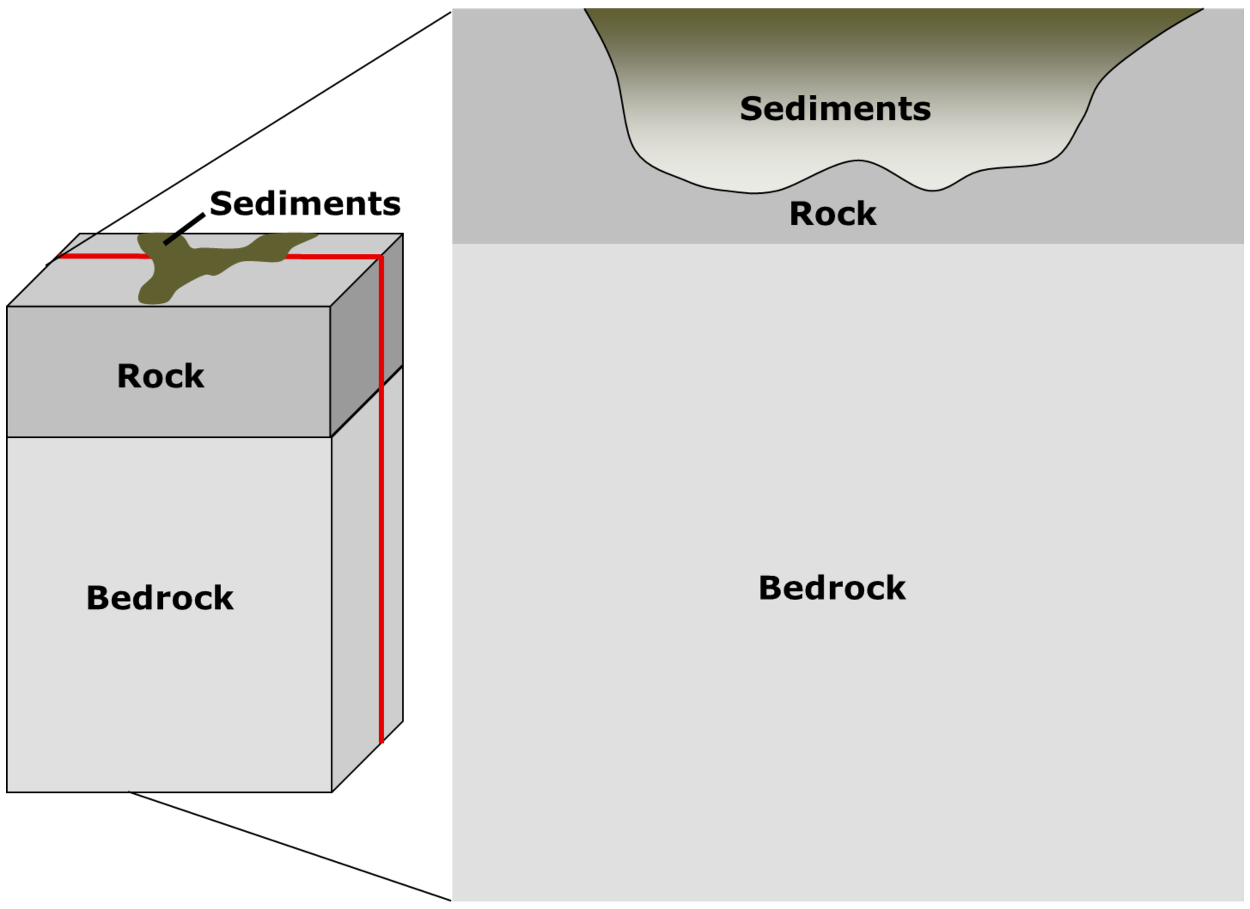

Figure 1 Schematic illustration of the medium

The medium could be covered by a uniform Cartesian or a discontinuous grid. In case of the discontinuous grid, a finer grid covers the upper part of the medium whereas a coarser grid covers the bottom part of the medium. Grid spacing in the finer grid is several times smaller than the grid spacing in the coarser grid. An example of the discontinuous grid is illustrated in Figure 2.

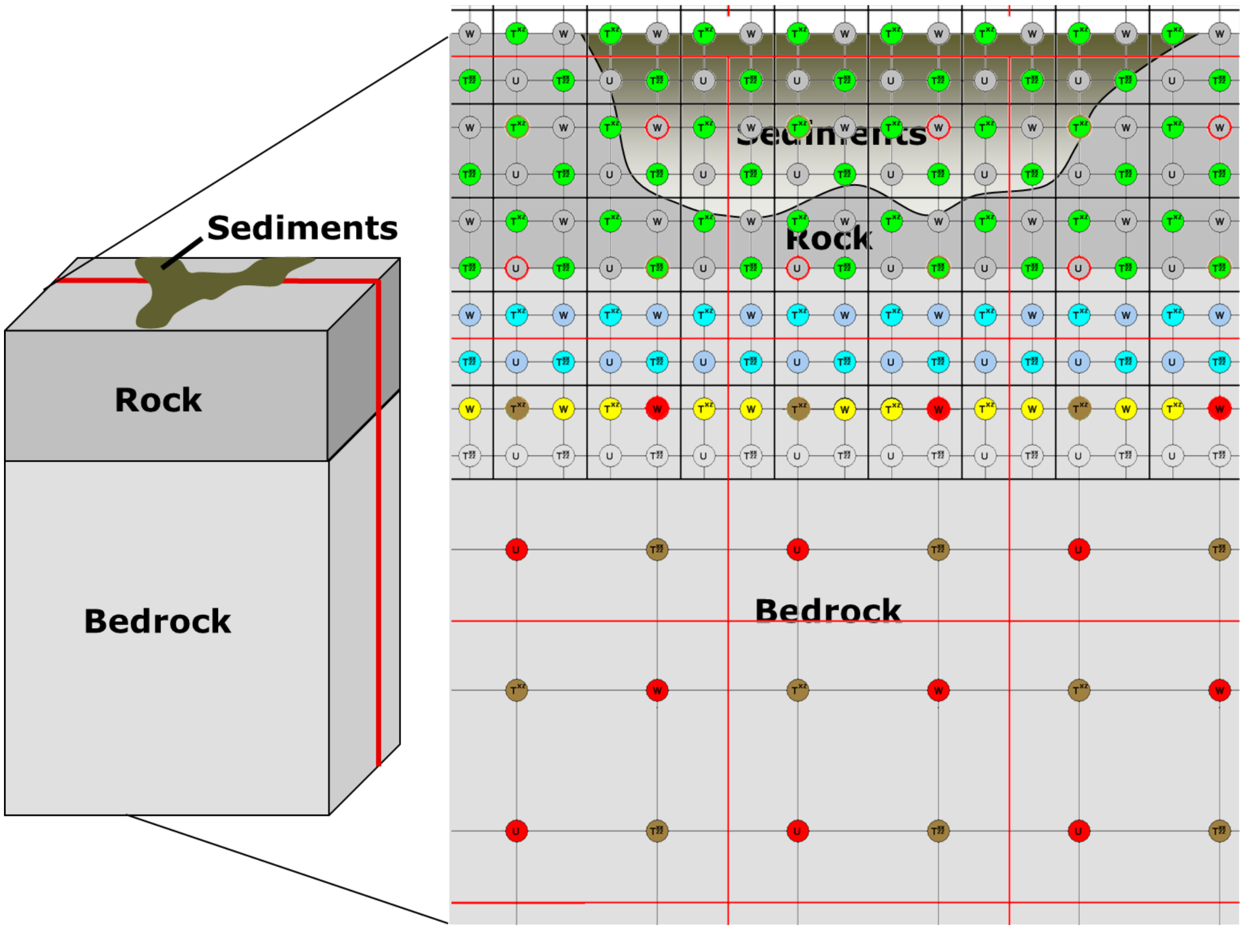

Figure 2 Illustration of the medium covered by the discontinuous grid (vertical cross-section)

Surfaces of material interfaces are specified by a rectilinear mesh of points at the horizontal (x,y) plane. Positions (depths) of the interfaces are defined at each point of the plane.

The spatial distributions of the wave speeds, density and quality factors should be described in the FORTRAN module `mod_func.f90` using functions. There are two sets of FORTRAN functions: The set of functions for the 3D heterogeneous part (functions depend on the x, y and z coordinates), and the set of functions for the 1D heterogeneous part (functions depend only on the z coordinate). The example is given in Subsection 4.2.

FDMod3D can be executed in a serial or parallel mode. In the parallel mode the computational model is partitioned only in the x-direction (easting). This means that the parallelization is less efficient if the model is prolonged in the North-South direction. The maximum number of processes should be smaller than the number of grid spacings in the x-direction in the coarser grid.

4.1 Compilation

The FORTRAN compiler is necessary for the successful compilation. The minimum level is FORTRAN 95. The `fpp` FORTRAN pre-processor is necessary because the source code uses a few macros for conditional compilation. The conditional compilation makes it possible to choose either serial or parallel version of the executable program. No non-standard library is required.

Because the source code consists of 45 files, the compilation is performed using the `make` utility. The `make` utility automatically builds an executable (target) program from the source code by reading a file called Makefile. Makefile specifies building of the executable program. Therefore, it is necessary to edit the provided Makefile before the first compilation. Examples how to edit the Makefile are given in the next subsections.

The building of executable program is invoked by statement `make` or `make –f Makefile`. After successful compilation and linking the executable program `FDMod3D` should be created.

4.1.1 Serial program

The only part which should be edited in the Makefile, is the part defining the FORTRAN compiler and flags for compilation and linking. In the Makefile there are these three lines:

```
FC = ifort
FFLAGS = -fpp –Ofast
LDFLAGS = -Ofast
```

This example is valid for the Intel FORTRAN Compiler `ifort`. In case of other compiler, the compiler name should be assigned to variable FC, compilation flags to variable FFLAGS and linking flags to LDFLAGS. The better performance could be achieved using highly optimized compilation (flags like –O3 or –Ofast), but the executable program should be always checked against a low-level optimized version.

4.1.2 Parallel (MPI) program

The source code uses the Message Passing Interface libraries for parallelisation. A suitable implementation of MPI should be installed. The code was tested with OpenMPI and MPICH but all implementations with standard MPI 1.2 should work.

The only part, which should be edited in the Makefile, is the part defining the FORTRAN compiler and flags for compilation and linking. In the Makefile there are these three lines:

```
FC = mpif90
FFLAGS = -fpp –Ofast –DUSE_MPI
LDFLAGS = -Ofast
```

This example is valid for an MPI implementation compiled with the Intel FORTRAN Compiler `ifort`. In case of other compiler, the MPI compiler name should be assigned to variable FC, compilation flags to variable FFLAGS and linking flags to LDFLAGS. The better performance could be achieved using highly optimized compilation (flags like –O3 or –Ofast), but the executable program should be always checked against low-level optimized version. The flag –DUSE_MPI defines macro which switches on the parallel version of the code.

If the MPI implementation requires use of statement USE MPI instead of statement INCLUDE 'mpif.h', then it is necessary to define also macro –DMPI2. The Makefile then reads:

```
FC = mpif90
FFLAGS = -fpp –Ofast –DUSE_MPI –DMPI2
LDFLAGS = -Ofast
```

Please, contact your system administrator for advice and help if necessary.

4.2 FORTRAN module mod_func.f90

The file contains two sets of FORTRAN functions.

The first set is applicable to medium above the bottom interface, i.e. to the 3D heterogeneous medium. The set contains functions

- FUNCVS (X,Y,Z) – returns value of the S-wave speed at the position X,Y,Z
- FUNCVP (X,Y,Z) – returns value of the P-wave speed at the position X,Y,Z
- FUNCRHO (X,Y,Z) – returns value of density at the position X,Y,Z
- FUNCQS (X,Y,Z) – returns value of the quality factor for the S waves at the position X,Y,Z
- FUNCQP (X,Y,Z) – returns value of the quality factor for the P waves at the position X,Y,Z

where X,Y,Z are the easting, northing and upward coordinates in meters, respectively.

The second set is applicable to the rest of the medium, i.e. to the 1D vertically heterogeneous medium. The set contains functions

- FUNCVS_1D (Z) – returns value of the S-wave speed at the position Z
- FUNCVP_1D (Z) – returns value of the P-wave speed at the position Z
- FUNCRHO_1D (Z) – returns value of density at the position Z
- FUNCQS_1D (Z) – returns value of the quality factor for the S waves at the position Z
- FUNCQS_1D (Z) – returns value of the quality factor for the P waves at the position Z

where Z is the upward coordinate in meters.

There is also a function called GETDEPTH, accessible inside the module, which can be used in the above listed functions. Function GETDEPTH (X,Y,NI) returns depth of the NI-th material interface at position (X,Y).

An example of `mod_func.f90` for the model of the 2D structure representing a simplified NS profile of the Mygdonian basin going through the TST seismic station (model Can4, see Subsection 19.2.3 in MKG2014) is given here:

```
MODULE MOD_FUNC

! module contains functions for calculation of
! P and S-wave velocities, density and quality factors
! in 3D sediments and in 1D bedrock

! Note: Z IS LESS OR EQUAL TO 0

use mod_model,      only: ni
use mod_interfaces, only: getdepth

IMPLICIT NONE

CONTAINS
```

```
!=================== 3D INHOMOGENEOUS STRUCTURE =====================

!------------------------------------------------------- S-wave velocity
  FUNCTION FUNCVS(X,Y,Z)

    USE NRTYPE, ONLY: WP

    REAL(WP), INTENT(IN) :: X,Y,Z
    REAL(WP)             :: FUNCVS
    REAL(WP)             :: H1, H2, H3, Z1, Z2, Z3

    H1 = - 17.3
    H2 = - 72.5
    H3 = -115.6

    IF ( ( Y < 2500. ) .AND. ( Y > -2500. ) ) THEN
      Z1 =      H1 * MIN ( 1., (2500.-Y) / 1500. )
      Z2 = Z1 + H2 * MIN ( 1., (2500.-Y) / 1500. )
      Z3 = Z2 + H3 * MIN ( 1., (2500.-Y) / 1500. )
    ELSE
      Z1 = 0.
      Z2 = 0.
      Z3 = 0.
    END IF

    IF ( Z > Z1 ) THEN
      FUNCVS = 200.
    ELSE IF ( Z > Z2 ) THEN
      FUNCVS = 350.
    ELSE IF ( Z > Z3 ) THEN
      FUNCVS = 650.
    ELSE
      FUNCVS = 2600.
    END IF

  END FUNCTION FUNCVS

!------------------------------------------------------- P-wave velocity
  FUNCTION FUNCVP(X,Y,Z)

    USE NRTYPE, ONLY: WP

    REAL(WP), INTENT(IN) :: X,Y,Z
    REAL(WP)             :: FUNCVP
    REAL(WP)             :: H1, H2, H3, Z1, Z2, Z3

    H1 = - 17.3
    H2 = - 72.5
    H3 = -115.6

    IF ( ( Y < 2500. ) .AND. ( Y > -2500. ) ) THEN
      Z1 =      H1 * MIN ( 1., (2500.-Y) / 1500. )
      Z2 = Z1 + H2 * MIN ( 1., (2500.-Y) / 1500. )
      Z3 = Z2 + H3 * MIN ( 1., (2500.-Y) / 1500. )
    ELSE
      Z1 = 0.
      Z2 = 0.
      Z3 = 0.
    END IF

    IF ( Z > Z1 ) THEN
```

```
        FUNCVP = 1500.
      ELSE IF ( Z > Z2 ) THEN
        FUNCVP = 1800.
      ELSE IF ( Z > Z3 ) THEN
        FUNCVP = 2500.
      ELSE
        FUNCVP = 4500.
      END IF

    END FUNCTION FUNCVP

  !---------------------------------------------------------------- Density
    FUNCTION FUNCRHO(X,Y,Z)

      USE NRTYPE, ONLY: WP

      REAL(WP), INTENT(IN) :: X,Y,Z
      REAL(WP)             :: FUNCRHO
      REAL(WP)             :: H1, H2, H3, Z1, Z2, Z3

      H1 = - 17.3
      H2 = - 72.5
      H3 = -115.6

      IF ( ( Y < 2500. ) .AND. ( Y > -2500. ) ) THEN
        Z1 =      H1 * MIN ( 1., (2500.-Y) / 1500. )
        Z2 = Z1 + H2 * MIN ( 1., (2500.-Y) / 1500. )
        Z3 = Z2 + H3 * MIN ( 1., (2500.-Y) / 1500. )
      ELSE
        Z1 = 0.
        Z2 = 0.
        Z3 = 0.
      END IF

      IF ( Z > Z1 ) THEN
        FUNCRHO = 2100.

      ELSE IF ( Z > Z2 ) THEN
        FUNCRHO = 2100.
      ELSE IF ( Z > Z3 ) THEN
        FUNCRHO = 2200.
      ELSE
        FUNCRHO = 2600.
      END IF

    END FUNCTION FUNCRHO

  !---------------------------------------------------- S-wave quality factor
    FUNCTION FUNCQS(X,Y,Z)

      USE NRTYPE, ONLY: WP

      REAL(WP), INTENT(IN) :: X,Y,Z
      REAL(WP)             :: FUNCQS
      REAL(WP)             :: Z1, Z2, Z3

      FUNCQS = 9999.

    END FUNCTION FUNCQS
```

```
!------------------------------------------------ P-wave quality factor
  FUNCTION FUNCQP(X,Y,Z)

    USE NRTYPE, ONLY: WP

    REAL(WP), INTENT(IN) :: X,Y,Z
    REAL(WP)             :: FUNCQP

    FUNCQP = 9999.

  END FUNCTION FUNCQP

!================== 1D INHOMOGENEOUS BEDROCK =======================
!------------------------------------------------------ S-wave velocity
  FUNCTION FUNCVS_1D(Z)

    USE NRTYPE, ONLY: WP

    REAL(WP), INTENT(IN)   :: Z
    REAL(WP)               :: FUNCVS_1D

    FUNCVS_1D = 2600.

  END FUNCTION FUNCVS_1D

!------------------------------------------------------ P-wave velocity
  FUNCTION FUNCVP_1D(Z)

    USE NRTYPE, ONLY: WP

    REAL(WP), INTENT(IN) :: Z
    REAL(WP)             :: FUNCVP_1D

    FUNCVP_1D = 4500.

  END FUNCTION FUNCVP_1D

!-------------------------------------------------------------- Density
  FUNCTION FUNCRHO_1D(Z)

    USE NRTYPE, ONLY: WP

    REAL(WP), INTENT(IN) :: Z
    REAL(WP)             :: FUNCRHO_1D

    FUNCRHO_1D = 2600.

  END FUNCTION FUNCRHO_1D

!------------------------------------------------ S-wave quality factor
  FUNCTION FUNCQS_1D(Z)

    USE NRTYPE, ONLY: WP

    REAL(WP), INTENT(IN) :: Z
    REAL(WP)             :: FUNCQS_1D

    FUNCQS_1D = 9999.

  END FUNCTION FUNCQS_1D
```

```
!------------------------------------------------ P-wave quality factor
  FUNCTION FUNCQP_1D(Z)

    USE NRTYPE, ONLY: WP

    REAL(WP), INTENT(IN) :: Z
    REAL(WP)             :: FUNCQP_1D

    FUNCQP_1D = 9999.

  END FUNCTION

END MODULE
```

Geometry of the structure is shown in Figure 3, the material parameters are in Figure 4.

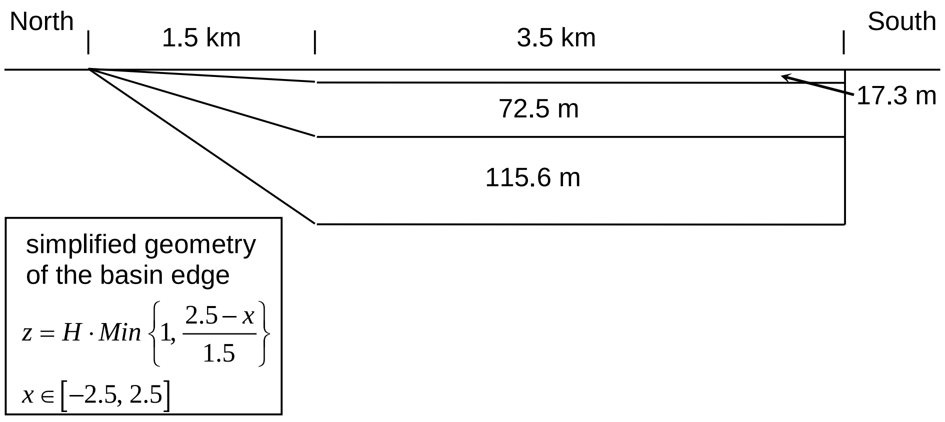

Figure 3 Geometry of model Can4

	S wave speed	P wave speed	Density	P wave quality factor	S wave quality factor
	(m/s)	(m/s)	(kg/m^3)		
Top layer	200	1500	2100	9999	9999
Medium layer	350	1800	2200	9999	9999
Bottom layer	650	2500	2200	9999	9999
Bedrock	2600	4500	2600	9999	9999

Figure 4 Material parameters of model Can4

4.3 Input data files

FDMod3D requires these input data files:

- an auxiliary file containing only the name of the current computation (the same file as for FDSim3D)
- input data file with a set of controlling parameters for the computation
- input data files containing surfaces of material interfaces

Note
In the following description of the input data *default* means that the variables need not to be specified.

4.3.1 HF_FDSim3D

Auxiliary input data file. The file type is ASCII and consists of the following variable:

Name of Variable	*Type*	*Description*
JOBNAME	A17	The name of the current computation. This name is taken as a basis for constructing (by the program) names of other input and output files.

4.3.2 *JOBNAME*.MD

Input data file specifying several controlling variables. The file type is ASCII and consists of the following variables:

```
NAMELIST  /CONTROL/   H, PTS, KEY_Q, NK,  KEY_FILTER, NPX, NPY, NI,            &
                      FLTR_RANGE, REFINE
```

Name of Variable	*Type*	*Description*
H	real	The spatial grid spacing in the coarser grid in meters.
PTS	integer	The number of points in each direction of a cube over which the integrals are to be numerically calculated; the number of the integration points is PTS ^{3} (PTS has to be even). Note: The cube has dimensions of the grid cell (in the coarser or finer grid). It is centred at a grid position of a relevant field quantity. See Chapter 9 in MKG2014 for more. *default:* PTS = 8

Name of Variable	*Type*	*Description*
KEY_Q	logical	.TRUE.: The viscoelastic model is constructed. .FALSE.: The perfectly elastic model is constructed. *default:* KEY_Q = .TRUE.
NK	integer	The number of the yz-planes computed in advance. It should be less than the total numbers of the grid cells in the x-direction (easting) in the coarser grid divided by the number of processes used for computation; a larger value means faster computation but also larger memory requirements. *default*: NK = 1
KEY_FILTER	logical	.TRUE.: Surfaces of the material interfaces are filtered using moving average filter .FALSE.: No filtration of the surfaces. *default:* KEY_FILTER = .FALSE.
FLTR_RANGE	real	Length of the side of a square over which the surfaces of the material interface are filtered. *default:* FLTR_RANGE = 3*H
NI	integer	The number of the material-interface surfaces. *default:* NI = 1
NPX, NPY	Integer	The numbers of the mesh points defining the surfaces of the material interfaces. The first number is in the x direction, the second number is in the y direction. See Subsection 5.3.3. Note: term mesh is used in relation to description of the material-interface surfaces, term grid means the finite-difference grid.
REFINE	integer	The ratio between the grid spacings in the coarser and finer grids. *default:* REFINE = 1

NAMELIST /BOUNDARIES/ XBMIN, XBMAX, YBMIN, YBMAX, ZBMIN, ZHF

Name of Variable	*Type*	*Description*
XBMIN, XBMAX	real	x-coordinates (easting) of the left and right boundaries of the computational model in meters.

Name of Variable	*Type*	*Description*
YBMIN, YBMAX	real	y-coordinates (northing) of the front and rear boundaries of the computational model in meters.
ZBMIN	real	z-coordinate (upward) of the bottom boundary of the computational model in meters. (The free surface is located at the z = 0 m.)
ZHF	real	z-coordinate (upward) of the bottom boundary of the finer grid in meters. If ZHF = 0 then the created grid will be uniform Cartesian.

NAMELIST /ATTEN/ FRJMIN, FRJMAX, FRANGE, FREF

Name of Variable	*Type*	*Description*
FRJMIN	real	The lower limit of the frequency range in which the discrete grid model should correspond to the desired $Q(\omega)$ law. The value is in Hz. *default:* FRJMIN is determined from FRANGE
FRJMAX	real	The upper limit of the frequency range in which the discrete grid model should correspond to the desired $Q(\omega)$ law. It should be larger (less than one order) than the maximum frequency f_{AC} up to which the computation should be sufficiently accurate. The value is in Hz.
FRANGE	real	This variable determines the frequency range for the attenuation in which the discrete grid model should correspond to the desired $Q(\omega)$ law. FRANGE = 3, e.g., means frequency range $\left\langle \text{FRJMAX} * 10^{-3}, \text{FRJMAX} \right\rangle$. The value is ignored if FRJMIN > 0
FREF	real	The reference frequency at which the values of wave speeds are prescribed for the grid model. The value is in Hz.

At the end of file *JOBNAME*.MD are names of NI input data files with specification of the material-interface surfaces. The names are read by

```
DO I = 1, NI
  READ (10,*) FSED(I)
```

```
    END DO
```

where `FSED` is a character string with not more than 20 alphanumeric characters.
Note:
The order of the input data files must correspond to the order of the material-interface surfaces starting from the uppermost surface and ending with the bottom interface. The interfaces may partially coincide but they must not intersect.

4.3.3 Files with the surfaces of the material interfaces

Each file contains NPX × NPY rows. Each row corresponds to one point at the horizontal (x,y) plane and contains three columns. The first and second columns contain the x- and y- coordinates, respectively. The third column contains depth of the material interface. The depth is always less or equal to 0 and all values are in meters. The points should compose a rectilinear mesh of NPX × NPY points. The mesh should cover an area larger than the target computational model (at least by one coarser-grid spacing in each direction).

The order of the NPX × NPY rows must satisfy the following condition: for each value of the y-coordinate, the value of the x-coordinate must increase.

At least one such file has to be prepared because the NI-th file delimitates the 3D and 1D heterogeneous parts of the model.

4.4 Output data files

FDMod3D generates several output data files:

- a log file containing the input parameters (as read by the program) and error messages
- data file with the second set of controlling parameters for the computation
- data file containing elastic parameters and densities describing types of material cells
- data file containing anelastic parameters describing types of material cells
- one or two input data files containing spatial distribution of material cell types

4.4.1 MODEL.LOG

Log file containing the input parameters read from the input data files and information about progress. In case of certain error(s) in the computation the file also contains the error message. The file type is ASCII.

4.4.2 *JOBNAME*.INM

The output data file makes the input data file for the finite-difference program FDSim3D. See Subsection 3.2.1

4.4.3 *MO_FILE_NAME*

The output data file makes the input data file for the finite-difference program FDSim3D. See Subsection 3.2.2

4.4.4 *Q_FILE_NAME*

The output data file makes the input data file for the finite-difference program FDSim3D. See Subsection 3.2.3

4.4.5 *JMH_FILE_NAME*

The output data file makes the input data file for the finite-difference program FDSim3D. See Subsection 3.2.4

4.4.6 *JMHF_FILE_NAME*

The output data file makes the input data file for the finite-difference program FDSim3D. See Subsection 3.2.5

5 Model partition code FDModPar3D

The FDModPar3D is designed to partition the FD-grid model (created by FDMod3D) into several parts. The partitioning of the model is necessary only in case of parallel execution of the FD program FDSim3D. The number of parts is equal to a desired number of processes.

5.1 Compilation

The FORTRAN compiler is necessary for the successful compilation. The minimum level is FORTRAN 95. No non-standard library is required. To code is serial and command

```
ifort -O -o FDModPar3D FDModPar3D.f90
```

or similar should work. The example is given for the Intel FORTRAN Compiler.

5.2 Input data files

FDModPar3D requires the same input data files as program FDSim3D:

- Manually prepared input data files
  - ➢ an auxiliary file containing only the name of the current computation
  - ➢ input data file with the first set of controlling parameters for the computation
- Input data files created by the model preparation code FDMod3D
  - ➢ input data file containing elastic parameters and densities describing types of material cells
  - ➢ input data file containing anelastic parameters describing types of material cells
  - ➢ one or two input data files containing spatial distribution of material cell types

5.2.1 HF_FDSim3D

Auxiliary input data file. See Subsection 3.1.1.

5.2.2 *JOBNAME*.IN

Input data file specifying several controlling variables. See Subsection 3.1.2.

5.2.3 *MO_FILE_NAME*

Input data file specifying the elastic parameters and densities describing types of material cells. See Subsection 3.2.2.

5.2.4 *Q_FILE_NAME*

Input data file specifying the anelastic parameters describing types of material cells. See Subsection 3.2.3.

5.2.5 *JMH_FILE_NAME*

Input data file specifying spatial distribution of the material cell types in the **coarser** spatial grid whose grid spacing is H. See Subsection 3.2.4.

5.2.6 *JMHF_FILE_NAME*

Input data file specifying spatial distribution of material cell types in the **finer** spatial grid whose grid spacing is H / REFINE. See Subsection 3.2.5.

5.3 Output data files

Program FDModPar3D generates several output data files. A name of each output data file appends a three-digit number from 000 to NPROC-1, where NPROC = PROCX × PROCY × PROCZ

5.3.1 *MO_FILE_NAMEnnn*

Data file specifying the elastic parameters and densities describing types of material cells. It contains the nnn-th part of file *MO_FILE_NAME*. See Subsection 3.2.2

5.3.2 *Q_FILE_NAMEnnn*

Data file specifying the anelastic parameters describing types of material cells. It contains the nnn-th part of the file *Q_FILE_NAME*. See Subsection 3.2.3

5.3.3 ***JMH_FILE_NAMEnnn***

Data file specifying spatial distribution of the material cell types in the **coarser** spatial grid whose grid spacing is H. It contains the nnn-th part of file *JMH_FILE_NAME*. See Subsection 3.2.4

5.3.4 ***JMHF_FILE_NAMEnnn***

Data file specifying spatial distribution of material cell types in the **finer** spatial grid whose grid spacing is H / REFINE. It contains the nnn-th part of file *JMHF_FILE_NAME*. See Subsection 3.2.5

6 Code SourceTF

SourceTF generates a time signal which can be used as a source-time function.

6.1 Compilation

The FORTRAN compiler is necessary for the successful compilation. The minimum level is FORTRAN 95. No non-standard library is required.

Because the source code consists of 3 files, the compilation is performed using the `make` utility. The `make` utility automatically builds an executable (target) program from the source code by reading a file called Makefile. Makefile specifies building of the executable program. Therefore, it is necessary to edit the provided Makefile before the first compilation. The only part which should be edited in the Makefile, is the part defining the FORTRAN compiler and flags for compilation and linking. In the Makefile there are these three lines:

```
FC = ifort
FFLAGS = –Ofast
LDFLAGS = -Ofast
```

This example is valid for the Intel FORTRAN Compiler `ifort`. In case of other compiler, the compiler name should be assigned to variable FC, compilation flags to variable FFLAGS and linking flags to LDFLAGS. The better performance could be achieved using highly optimized compilation (flags like –O3 or –Ofast), but the executable program should be always checked against a low-level optimized version.

The building of executable program is invoked by statement `make` or `make –f Makefile`. After successful compilation and linking the executable program `SourceTF` should be created.

6.2 Input file

6.2.1 SOURCETF.IN

The file contains several controlling variables grouped in two namelists. The file type is ASCII.

```
NAMELIST  /INPUT/        NSIG,  DT
```

Name of Variable	*Type*	*Description*
NSIG	integer	Key determining type of the generated signal: = 1: Küpper = 2: Ricker = 3: Gabor = 4: Berlage
DT	real	The time step Δt in seconds.

The next namelist is one of the following namelists – depending on the chosen type of the signal.

Küpper signal (NSIG = 1)

The signal is defined by

$$s(t) = \sin\left(2\pi \frac{t}{T}\right) - \frac{1}{2}\sin\left(4\pi \frac{t}{T}\right),$$

where T is (approximately) the dominant period. The signal is defined in interval $\langle 0, T \rangle$.

NAMELIST /SIGNAL_1/ TP

Name of Variable	*Type*	*Description*
TP	real	The dominant period T.

Ricker signal (NSIG = 2)

The signal is defined by

$$s(t) = \frac{\sqrt{\pi}}{2}\left(a - \frac{1}{2}\right)e^{-a}; \qquad a = \left(\pi \frac{t - t_S}{t_P}\right)^2,$$

where t_P is the dominant period and $t_S = 1.1 t_P$. The signal is defined in interval $\langle 0, 2t_S \rangle$.

NAMELIST /SIGNAL_2/ TP, TS

Name of Variable	*Type*	*Description*
TP	real	The dominant period t_P in seconds.
TS	real	The time shift t_S in seconds. If $\mathsf{TS} = 0$, then TS is determined using $t_S = 1.1 t_P$.

Gabor signal (NSIG = 3)

The signal is defined by

$$s(t) = e^{-\left(\frac{2\pi f_P (t-t_S)}{\gamma}\right)^2} \cos\left(2\pi f_P (t-t_S) + \Psi\right),$$

where f_P is (for certain values of γ and Ψ) the dominant frequency, γ controls the width of the signal envelope and $t_S = 0.45\frac{\gamma}{f_P}$. The signal is defined in interval $\langle 0, 2t_S \rangle$.

NAMELIST /SIGNAL_3/ GAMA, FP, PSI, TS

Name of Variable	*Type*	*Description*
GAMA	real	Parameter γ controlling the width of the signal envelope.
FP	real	The dominant frequency f_P in *Hz*.
PSI	real	Phase Ψ in radians.
TS	real	The time shift t_S in seconds. If $\mathsf{TS} = 0$, then TS is determined using $t_S = 0.45\frac{\gamma}{f_P}$.

Berlage signal (NSIG = 4)

The signal is defined by

$$s(t) = (t-t_S)^{\zeta} e^{-\frac{2\pi f_P (t-t_S)}{\gamma}} \sin\left(2\pi f_P (t-t_S)\right),$$

where f_P is the dominant frequency and γ controls the width of the signal envelope. The signal is defined in interval $\langle 0, t_S \rangle$.

NAMELIST /SIGNAL_4/ GAMA, FP, ZETA, TS

Name of Variable	*Type*	*Description*
GAMA	real	Parameter γ controlling the width of the signal envelope.
FP	real	The dominant frequency f_P in *Hz*.
ZETA	real	Parameter ζ.
TS	real	The time shift t_S in seconds.

6.3 Output files

Program SourceTF generates three output files:

- a file containing the signal; it can be read by FDSim3D,
- a file containing the signal and its envelope,
- a file containing the power, amplitude and phase Fourier spectra of the signal.

6.3.1 SIGNAL.DAT

The file type is ASCII and contains the generated signal in the two-column form:

Time	*Signal*

6.3.2 ENVELOPE.DAT

The file type is ASCII and contains the envelope of generated signal in the two-column form:

Time	*Envelope of the signal*

6.3.3 SPECTRUM.DAT

The file type is ASCII and contains the power, amplitude and phase Fourier spectrum of the generated signal in the four-column form:

Frequency	*Power Fourier spectrum of the signal*	*Amplitude Fourier spectrum of the signal*	*Phase Fourier spectrum of the signal*

7 Execution

The programs run in non-interactive mode, that is, the input is read in only from the input data files and output is written into the output data files. All input and output data files should be in the same directory from which the programs are executed. The programs are suitable for running in background or in queuing system.

7.1 Serial execution

The sequence of the necessary steps to run the program in a serial mode could be written as follows:

1. Prepare input data files for the model preparation code FDMod3D
    a. an auxiliary file containing only the name of the current computation HF_FDSim3D
    b. input data file with a set of the controlling parameters JOBNAME.MD
    c. files with the surfaces of the material interfaces
2. Prepare FORTRAN module mod_func.f90
3. Compile and build FDMod3D according to Subsection 4.1.1
4. Run model preparation code FDMod3D
5. Prepare input data files for the finite-difference code FDSim3D
    a. input data file with the first set of the controlling parameters JOBNAME.IN
    b. input data file containing the names and positions of the receivers
    c. input data file containing parameters specifying the type of a wavefield excitation
    d. input data file(s) containing the source time function(s)
6. Compile and build FDSim3D according to Section 2.1
7. Run FDSim3D

7.2 Parallel execution

The sequence of the necessary steps to run the program in a parallel mode could be written as follows:

1. Prepare input data files for the model preparation code FDMod3D
    a. an auxiliary file containing only the name of the current computation HF_FDSim3D
    b. input data file with a set of the controlling parameters JOBNAME.MD
    c. files with the surfaces of the material interfaces
2. Prepare FORTRAN module mod_func.f90

3. Compile and build FDMod3D according to Subsection 4.1.2
4. Run model preparation code FDMod3D in the parallel mode with a desired number of processes
5. Prepare input data files for the finite-difference code FDSim3D
    a. input data file with the first set of the controlling parameters JOBNAME.IN
    b. input data file containing the names and positions of the receivers
    c. input data file containing parameters specifying the type of a wavefield excitation
    d. input data file(s) containing the source time function(s)
6. Compile and run code FDModPar3D for model partitioning in the serial mode.
8. Compile and build FDSim3D according to Section 2.2
7. Run FDSim3D in the parallel mode with a desired number of processes. The number of processes used for FDSim3d and FDMod3D could be different.