Marchenko
Basic-PlaneWave-MME-3D-MDD

Jan Thorbecke, Joeri Brackenhoff, Lele Zhang, Giovanni Meles, Johno van IJsseldijk

August 8, 2024

Contents

1 Getting Started

1.1 Installation

There are different ways to get the source code for the Marchenko implementations. Our advise is to get a tagged snapshot from the GitHub repository. These tagged snapshots are tested and stable releases. The latest tagged version that can be cloned from the repository with
`git clone -b '2.1.1' --single-branch git://github.com/JanThorbecke/OpenSource.git`
and contains the full package. To get a version with the latest development and bug fixes just get a clone of the most current version
`git clone git://github.com/JanThorbecke/OpenSource.git`
The package extracts into the directory `OpenSource`, with the following sub-directories:

- FFTlib: basic library for FFT's includes a wrapper for MKL, not based on FFTW.
- MDD: Multiple Dimensional Deconvolution to solve different $Ax = b$ problems by $x = \frac{bA^*}{AA^*}$.
- corrvir: seismic interferometry (correlation) for passive signals, tested on many data sets.
- doc: documentation related to the code.
- extrap: recursive wavefield depth extrapolation, includes a recursive depth migration program.
- extrap3d: 3D version of the above.
- fdacrtmc: RTM based on fdelmodc.
- fdelmodc: finite difference modeling (visco)-acoustic, and (visco)-elastic.
- fdelmodc3D: 3D version for acoustic media.
- fdemmodc: EM finite difference code.
- marchenko: basic, plane-wave and MME implementations.
- marchenko3D: 3D version of the basic algorithm.
- raytime: eikonal solver.
- raytime3D: 3D eikonal solver.
- utils: basic (pre-) processing and additional programs for 3D-Marchenko.
- zfp: ZFP data compression library from Peter Lindstrom .

Besides the Marchenko algorithms the OpenSource package contains lots of other software. In this manual we will only describe the different Marchenko implementations that can be found in the directories `marchenko, marchenko3D, utils`.
The `INSTALL` file in the ROOT directory contains guidelines how to compile the code package. The `README.md` briefly explains the different code packages and how to reproduce the figures in our published papers. Section A of this manual contains a brief (one-sentence) explanation of the meaning of the Marchenko source code files in the source tree of this package.
The code is used by many different people and new options are build into the code on a regular basis.

1.2 Compilation and Linking

1. To compile and link the code you first have to set the ROOT variable in the Make_include file which can be found in the directory where you have found the `README.md` and `INSTALL` instructions.

2. Check the compiler and CFLAGS options in the file Make_include and adapt to the system you are using. The default options are set for a the GNU C-compiler on a Linux system. A Fortran or g++ compiler is only needed to compile the MDD code. The compilation of the source code has been compiled and tested with several versions of GNU, AMD, PGI, Cray and Intel compilers.

3. If the compiler options are set in the Make_include file you can type

   ```
   > make
   ```

   and the Makefile will execute the commands to compile and link the source code in all the sub-directories.

The compiled FFT and ZFP libraries will be placed in the `lib/` directory, the executables in the `bin/` directory and the include files of the FFT and ZFP libraries in the `include/` directory.
To use the executables don't forget to include the pathname in your PATH:

```
bash:
export PATH='path_to_this_directory'/bin:$PATH:
csh:
setenv PATH 'path_to_this_directory'/bin:$PATH:
```

On Linux systems using the bash shell you can put the `export PATH='path_to_this_directory'/bin:$PATH:` setting in `$HOME/.bashrc`, to set it every time you login.
Other useful make commands are:

- `make clean`: removes all object files, but leaves libraries and executables
- `make realclean`: removes also object files, libraries and executables.

Important note: The examples and demo scripts make use programs of Seismic Unix (SU). Please make sure that SU is compiled without XDR: the XDR flag (`-DSUXDR`) in $CWPROOT/Makefile.config must **NOT** be set in compiling SU. The SU output files of fdelmodc are all base on local IEEE data. When the XDR flag is set in SU you have to convert the output files of fdelmodc (and all the programs in the utils directory: basop, fconv, extendmodel, makemod, makewave) with `suoldtonew`, before using SU programs.

1.3 Reproduce the results in our papers

1.3.1 Finite Difference for Seismic Interferometry

If the compilation has finished without errors and produced an executable called bin/fdelmodc you can run one of the demo programs by running

```
> ./fdelmodc_plane.scr
```

in the directory `fdelmodc/demo/`. This demo directory contains many scripts which demonstrate the different possibilities of the modeling program.

- To reproduce the Figures shown in the Geophysics manuscript **"Finite-difference modeling experiments for seismic interferometry"** (Thorbecke and Draganov, 2011) the scripts in FiguresPaper directory can be used. Please read the README in the FiguresPaper directory for more instructions and guidelines.

To clean-up all the produced output files in the `fdelmodc/demo/` and `fdelmodc/FiguresPaper/` directory you can run the `clean` script in those directories.
An extensive manual of fdelmodc can be found in `doc/fdelmodcManual.pdf`.

1.3.2 Marchenko: basic, plane-waves and MME

If the compilation has finished without errors and produced an executable in `bin/` called `marchenko` you can run one of the demo programs by running a set of scripts that are explained in a README in one of the directories `marchenko/demo/oneD` or `demo/twoD`.

- To reproduce the Figures shown in the Geophysics paper **"Implementation of the Marchenko method"** the scripts in `marchenko/demo/oneD` directory can be used. The README in this directory gives more instructions and guidelines.
- To reproduce the Figures shown in the Scientific Reports paper **"Virtual acoustics in inhomogeneous media with single-sided access"** the scripts in marchenko/demo/ScientificReports directory can be used. The README in this directory gives more instructions and guidelines.
- To reproduce the Figures shown in the Geophysics paper **"Implementation of the Marchenko Multiple Elimination algorithm"** the scripts in `marchenko/demo/mme` directory can be used. The README_PRIMARIES in this directory gives more instructions and guidelines.
- To reproduce the Figures shown in the Geophysics paper **"Implementation of the 3D Marchenko method"** the scripts in `marchenko3D/demo/marchenko3D/oneD` directory can be used. The README in this directory gives more instructions and guidelines.
- To reproduce the Figures shown in the IEEE (under review) paper **"Design, implementation and application of the Marchenko Plane-wave algorithm"** the scripts in `marchenko/demo/planewave` directory can be used. The README in this directory gives more instructions and guidelines.

A brief manual about the MME program 'marchenko_primaries' can be found in doc/MMEmanual.pdf

1.3.3 MDD: target replacement

- To reproduce the Figures shown in the paper **"New Paper Johno"** the scripts in `MDD/demo/` directory can be used. The README in this directory gives more instructions and guidelines.

1.4 Citation references to code and algorithms

0. DOI reference of this software release
   https://zenodo.org/badge/latestdoi/23060862

1. If the Finite Difference code has helped you in your research please refer to the following paper in your publications:

   Finite-difference modeling experiments for seismic interferometry,
   Jan Thorbecke and Deyan Draganov, 2011, Geophysics, Vol. 76, no. 6 (November-December); p H1–H18, doi: 10.1190/GEO2010-0039.1 Download: pdf

2. If the Marchenko code has helped you in your research please refer to this paper in your publications:

   Implementation of the Marchenko method,
   Jan Thorbecke, Evert Slob, Joeri Brackenhoff, Joost van der Neut, and Kees Wapenaar, 2017, Geophysics, Vol. 82, no. 6 (November-December); p. WB29–WB45, doi: 10.1190/GEO2017-0108.1 Download: pdf

3. If you used the code to construct homogeneous Green's functions, please refer to this paper in your related publications:

   Virtual sources and receivers in the real Earth: Considerations for practical applications,
   Brackenhoff, J., Thorbecke, J., and Wapenaar, K., 2019, Journal of Geophysical Research - Solid Earth, Vol. 124, 11,802-11,821. pdf-file

   Virtual acoustics in inhomogeneous media with single-sided access,
   Wapenaar, K., Brackenhoff, J., Thorbecke, J., van der Neut, J., Slob, E., and Verschuur, E., 2018, Scientific Reports, Vol. 8, 2497. Download: pdf

4. When you are using the marchenko_primaries algorithm developed by Lele Zhang please refer to the following papers:

   Free-surface and internal multiple elimination in one step without adaptive subtraction,
   Lele Zhang and Evert Slob 2019, Geophysics, Vol. 84, no. 1 (January-February); p. A7-A11, doi: 10.1190/GEO2018-0548.1 Download: pdf

   Free-surface and internal multiple elimination in one step without adaptive subtraction,
   Jan Thorbecke, Lele Zhang, Kees Wapenaar and Evert Slob 2021, Geophysics, Vol. 86, no. 2 (March-April); p. xxxx, doi: xxxx Download: pdf

5. When you are using the plane wave versions of marchenko or marchenko_primaries algorithm developed by Giovanni Meles please refer to the following paper:

   Virtual plane-wave imaging via Marchenko redatuming:
   Meles, G.A., K. Wapenaar, and J. Thorbecke, 2018, Geophysical Journal International, Vol. **214** (1), p. 508–519.

6. If you use the fdacrtmc code of Max Holicki please refer to the following paper:

   Acoustic directional snapshot wavefield decomposition,
   Holicki, M., Drijkoningen, G., and Wapenaar, K., 2019, Acoustic directional snapshot wavefield decomposition: Geophysical Prospecting, Vol. 67, 32-51 Download: pdf

7. If you use the vmar code of Johno van IJsseldijk please refer to the following paper:

   Extracting small time-lapse traveltime changes in a reservoir using primaries and internal multiples after Marchenko-based target zone isolation:
   Van IJsseldijk, J., van der Neut, J., Thorbecke, J., and Wapenaar, K., 2023, Geophysics, Vol. 88 (2), R135-R143. Download: pdf

   -7- A reference to the extrapolation and migration programs is

   Design of one-way wavefield extrapolation operators, using smooth functions in WLSQ optimisation.
   Jan Thorbecke, Kees Wapenaar, Gerd Swinnen, 2004, Geophysics Vol 69, p. 1037-1045 Download: pdf

2 Introduction to Marchenko

In this section we describe in detail the implementational aspects of the two dimensional iterative acoustic-Marchenko method based on focusing functions. Although the algorithm is straightforward to implement, the treatment of amplitudes, and the initialisation steps of the first iterations require special attention. The input of the method is a reflection response without free-surface multiples and deconvolved by its source wavelet. The output of an SRME scheme can (in principle) provide this reflection response. A (smooth) background model is needed to calculate an initial focusing function to start the algorithm. The Numerical Examples section demonstrates the use of the algorithm and provides a user's first steps with the Marchenko technique.

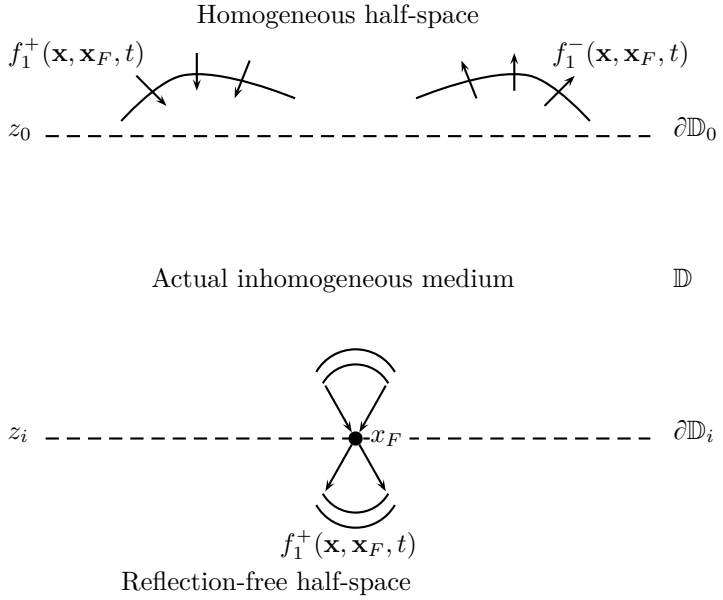


Figure 1: Downgoing and upgoing components of the focusing function f_1 for the 2D wave equation in a truncated medium.

The Marchenko method is briefly introduced here aiming at an explanation of the method that helps to understand the algorithm. The references mentioned in the introduction provide additional details on the derivation of this method. In an imagined medium truncated below level z_i we introduce a focusing function f_1. The truncated medium is identical to the actual medium above depth level z_i and reflection-free below this depth level. As illustrated in Figure 1 the actual and truncated media are both reflection free above the surface boundary $\partial\mathbb{D}_0$. We also introduce up- and downgoing parts of the f_1 focusing function (Slob et al., 2014)

$$f_1(\mathbf{x},\mathbf{x}_F,t) = f_1^+(\mathbf{x},\mathbf{x}_F,t) + f_1^-(\mathbf{x},\mathbf{x}_F,t),$$

where $\mathbf{x}_F = (x_F, z_i)$ is a focal position on the boundary $\partial\mathbb{D}_i$, $\mathbf{x}$ an observation point in the medium, and t is time (see Figure 1). In our notation the first argument represents the receiver location and the second argument stands for the focal point. The superscript $^+$ in f_1^+ denotes a downgoing field at observation point $\mathbf{x}$, and the superscript $^-$ in f_1^- an upgoing field, also at $\mathbf{x}$. Below boundary $\partial\mathbb{D}_i$ only f_1^+ continues as a diverging downgoing field into the reflection-free half-space.
The $f_1^\pm$ focusing functions are defined to relate the up- and downgoing Green's functions in the actual medium with the reflection response at the surface (Wapenaar et al., 2014b):

$$G^+(\mathbf{x}_F,\mathbf{x}_R,t) = -\int_{\partial\mathbb{D}_0}\int_{t'=-\infty}^{t} R(\mathbf{x}_R,\mathbf{x},t-t')f_1^-(\mathbf{x},\mathbf{x}_F,-t')\mathrm{d}t'\mathrm{d}\mathbf{x} + f_1^+(\mathbf{x}_R,\mathbf{x}_F,-t), \tag{1}$$

$$G^-(\mathbf{x}_F,\mathbf{x}_R,t) = \int_{\partial\mathbb{D}_0}\int_{t'=-\infty}^{t} R(\mathbf{x}_R,\mathbf{x},t-t')f_1^+(\mathbf{x},\mathbf{x}_F,t')\mathrm{d}t'\mathrm{d}\mathbf{x} - f_1^-(\mathbf{x}_R,\mathbf{x}_F,t). \tag{2}$$

$R(\mathbf{x}_R, \mathbf{x}, t)$ is the reflection response after surface multiple elimination, deghosting and deconvolution of the wavelet. The first argument in R represents the receiver location, the second argument the source location, and the last argument is time. R is a scaled (factor -2) pressure wavefield at $\mathbf{x}_R$ of a vertical force source F_z at $\mathbf{x}$ or, via reciprocity, the particle velocity field V_z at $\mathbf{x}$ of a point source of volume injection rate at $\mathbf{x}_R$ (Wapenaar et al., 2012). This reflection response is related to a Green's function by

$$\frac{\partial R(\mathbf{x}_R, \mathbf{x}, t)}{\partial t} = \frac{2}{\rho(\mathbf{x})} \frac{\partial G^s(\mathbf{x}_R, \mathbf{x}, t)}{\partial z} \tag{3}$$

with G^s the Green's function of the scattered field only (it does not contain the direct field). The upper integration boundary $t' = t$ of the time integral in equations (1) and (2) accounts for the causality of the reflection response $R(\mathbf{x}_R, \mathbf{x}, t - t')$. Summing equations 1 and 2 and using source-receiver reciprocity for the Green's function, gives (Wapenaar et al., 2017):

$$G(\mathbf{x}_R, \mathbf{x}_F, t) = \int_{\partial \mathbb{D}_0} \int_{t'=-\infty}^{t} R(\mathbf{x}_R, \mathbf{x}, t - t') f_2(\mathbf{x}_F, \mathbf{x}, t') \mathrm{d}t' \mathrm{d}\mathbf{x} + f_2(\mathbf{x}_F, \mathbf{x}_R, -t), \tag{4}$$

The Green's function $G(\mathbf{x}_R, \mathbf{x}_F, t)$ represents the response to a virtual point source of volume injection rate at $\mathbf{x}_F$ and pressure receivers at the surface $\mathbf{x}_R$. The focusing function f_2 is defined as

$$f_2(\mathbf{x}_F, \mathbf{x}, t) = f_1^+(\mathbf{x}, \mathbf{x}_F, t) - f_1^-(\mathbf{x}, \mathbf{x}_F, -t). \tag{5}$$

Wapenaar et al. (2014b) introduced $f_2(\mathbf{x}_F, \mathbf{x}, t)$ as a focusing function, which has its focal plane at $\partial \mathbb{D}_0$. Here we merely use f_2 as a compact notation for the combination of the one-way focusing functions f_1^+ and f_1^-, as defined in equation (5). Note that the time-reversed upgoing function f_1^- can be interpreted as a downgoing function, similar as f_1^+. Hence, from here onward we interpret $f_2(\mathbf{x}_F, \mathbf{x}, t)$ as a downgoing focusing function which is emitted into the medium from $\mathbf{x}$ and which focuses at receiver location $\mathbf{x}_F$. The argument change in equation (5) between $\mathbf{x}$ and $\mathbf{x}_F$ in the left-hand side f_2 and the right-hand side $f_1^{\pm}$ follows from the same logic in the order of the arguments as defined in Wapenaar et al. (2014b). In the Numerical Examples section we demonstrate that f_2 can be back-propagated into the medium and focuses at $\mathbf{x}_F$. In Wapenaar et al. (2013) a (reciprocal) relation between $f_2(\mathbf{x}_F, \mathbf{x}, t)$ and a downgoing wavefield $p^+(\mathbf{x}, \mathbf{x}_F, t)$ is given. Together with p^+ there is also a p^- defined, the upgoing reflection response at $\mathbf{x}$ from the focal point at $\mathbf{x}_F$. The sum of p^+ and p^- gives also the Green's function of equation (4). The $p^{\pm}$ functions are just a different notation of the Marchenko method and can be used to compute the Green's functions in a convenient way. These $p^{\pm}$ functions are therefore used in the implementation to compute the Green's function. From an educational point of view the Marchenko method is more easily understood by using the focusing functions only and we will continue along that way.
The above equations, on which the following implementation is based on, use pressure-normalised fields. Other papers derive similar equations based on flux-normalised fields (Wapenaar et al., 2014a; Singh et al., 2015; Van der Neut et al., 2015b). The relationship between pressure- and flux-normalised representations is explained in Wapenaar et al. (2014a).
The Marchenko algorithm estimates focusing functions $f_1^+(\mathbf{x}, \mathbf{x}_F, t)$ and $f_1^-(\mathbf{x}, \mathbf{x}_F, t)$. However, equations (1) and (2) are by themselves insufficient to determine f_1; there are four unknowns, but only two equations. We can eliminate two unknowns by exploiting the fact that focusing functions and Green's functions have different causality properties in the time-space domain. Based on the principle of causality we know that no energy arrives before the first arrival $t_d(\mathbf{x}_R, \mathbf{x}_F)$, hence the Green's function $G(\mathbf{x}_R, \mathbf{x}_F, t < t_d(\mathbf{x}_R, \mathbf{x}_F))$ is zero. This also holds for the up- and downgoing Green's functions and leads to

$$0 = -\int_{\partial \mathbb{D}_0} \int_{t'=-\infty}^{t} R(\mathbf{x}_R, \mathbf{x}, t - t') f_1^-(\mathbf{x}, \mathbf{x}_F, -t') \mathrm{d}t' \mathrm{d}\mathbf{x} + f_1^+(\mathbf{x}_R, \mathbf{x}_F, -t), \tag{6}$$

$$0 = \int_{\partial \mathbb{D}_0} \int_{t'=-\infty}^{t} R(\mathbf{x}_R, \mathbf{x}, t - t') f_1^+(\mathbf{x}, \mathbf{x}_F, t') \mathrm{d}t' \mathrm{d}\mathbf{x} - f_1^-(\mathbf{x}_R, \mathbf{x}_F, t), \tag{7}$$

where $t < t_d(\mathbf{x}_R, \mathbf{x}_F)$ in both equations above.
The combination of equations (6) and (7) is known as the Marchenko equation. These equations form the basis of the iterative scheme, which estimates focusing functions f_1^+ and f_1^-. In Wapenaar et al. (2014a) the relation

$$f_1^+(\mathbf{x}, \mathbf{x}_F, t) = T^{inv}(\mathbf{x}_F, \mathbf{x}, t), \tag{8}$$

is used to derive an initial estimate for f_1^+ that can start the inversion scheme. In equation (8) $T^{inv}(\mathbf{x}_F, \mathbf{x}, t)$ is the inverse of the transmission response of the truncated medium, which is equal to the actual medium between $\partial\mathbb{D}_0$ and $\partial\mathbb{D}_i$, for a source at $\mathbf{x}$ (at $\partial\mathbb{D}_0$) and a receiver at $\partial\mathbb{D}_i$. It is assumed that this inverse transmission response T^{inv} can be composed as a direct wave followed by scattering coda (Van der Neut et al., 2015b).

$$f_1^+(\mathbf{x}, \mathbf{x}_F, t) = T_d^{inv}(\mathbf{x}_F, \mathbf{x}, t) + M^+(\mathbf{x}, \mathbf{x}_F, t), \tag{9}$$

where M^+ is the unknown scattering coda and T_d^{inv} the direct arrival of the inverse transmission response. In equation (9) the inverse of the direct arrival of the transmission response is needed. For simplicity we take the time-reversal of the direct arrival of the Green's function; $G_d(\mathbf{x}, \mathbf{x}_F, -t)$.

$$f_1^+(\mathbf{x}, \mathbf{x}_F, t) \approx G_d(\mathbf{x}, \mathbf{x}_F, -t) + M^+(\mathbf{x}, \mathbf{x}_F, t). \tag{10}$$

We thereby introduce an overall scaling error and an offset dependent amplitude error, proportional to transmission losses, in the final result. $G_d(\mathbf{x}, \mathbf{x}_F, -t)$ is the time-reversed direct arrival part of the transmission response to subsurface focal point $\mathbf{x}_F$ and can for example be computed from a smooth macro model. As mentioned before, the arrival time of $G_d(\mathbf{x}, \mathbf{x}_F, t)$ is t_d, hence $G_d(\mathbf{x}, \mathbf{x}_F, t)$ is zero before $t < t_d$. The multiple scattering coda $M^+(\mathbf{x}, \mathbf{x}_F, t)$ follows after the first arrival of f_1^+, and is zero for $t \leqslant -t_d$. It can also be shown that it is also zero for $t \geqslant +t_d$ (Slob et al., 2014).
Equations (6) and (7) are only valid for $t < t_d$. We therefore define a time-window function:

$$\theta(\mathbf{x}_R, \mathbf{x}_F, t) = \begin{cases} 1 & t < t_d^\varepsilon \\ \frac{1}{2} & t = t_d^\varepsilon \\ 0 & t > t_d^\varepsilon \end{cases}. \tag{11}$$

where time t_d^ε is the time of the direct arrival from the focal point $\mathbf{x}_F$ to $\mathbf{x}_R$ (t_d), minus a small positive constant ε to exclude the wavelet in the direct arrival G_d. For example a time-window that sets all times $t < -t_d^\varepsilon$ to zero and applied to equation (10) mutes $G_d(-t)$, but leaves all events of M^+ in. In the following we will use the shorthand notation θ_t for $\theta(\mathbf{x}_R, \mathbf{x}_F, t)$. In the included Marchenko program there is an input parameter (called `smooth`, see Appendix A for all input parameters) that defines a temporal tapering in this mute-window to suppress high frequency artifacts.
It is further assumed that it is possible to get an estimate of this direct arrival G_d of the transmission response. Given the reflection response $R(\mathbf{x}_R, \mathbf{x}, t)$ and this direct arrival $G_d(\mathbf{x}, \mathbf{x}_F, t)$ from the focal point, the Marchenko algorithm solves for the scattering coda $M^+(\mathbf{x}, \mathbf{x}_F, t)$ to estimate $f_1^+(\mathbf{x}, \mathbf{x}_F, t)$ and $f_1^-(\mathbf{x}, \mathbf{x}_F, t)$.
The iterative solution of the Marchenko equations can now be developed. The iterative scheme is started with the following initialization of M^+;

$$M_0^+(\mathbf{x}_R, \mathbf{x}_F, t) = 0. \tag{12}$$

The subscript in M_0^+ defines the iteration number. By substituting equation (10), using (12) as initialisation, into equation (7) one arrives at the initialisation of f_1^-:

$$f_{1,0}^-(\mathbf{x}_R, \mathbf{x}_F, t) = \theta_t \int_{\partial\mathbb{D}_0} \int_{t'=-\infty}^{t} R(\mathbf{x}_R, \mathbf{x}, t - t') G_d(\mathbf{x}, \mathbf{x}_F, -t') \mathrm{d}t' \mathrm{d}\mathbf{x}. \tag{13}$$

Equation (13) includes the previously defined time-window function θ_t. Equations (10) and (6) are both expressions for f_1^+. By combining these equations the only part remaining in equation (10) is M^+. The iterative update of M^+ for step $k \geqslant 1$ is given by

$$M_k^+(\mathbf{x}_R, \mathbf{x}_F, -t) = \theta_t \int_{\partial\mathbb{D}_0} \int_{t'=-\infty}^{t} R(\mathbf{x}_R, \mathbf{x}, t - t') f_{1,k-1}^-(\mathbf{x}, \mathbf{x}_F, -t') \mathrm{d}t' \mathrm{d}\mathbf{x}. \tag{14}$$

Following the assumption in equation (10), that it is possible to write $f_{1,k}^+$ as a direct field plus scattering coda, the update at step k of $f_{1,k}^+$ is given by

$$f_{1,k}^+(\mathbf{x}_R, \mathbf{x}_F, t) = G_d(\mathbf{x}_R, \mathbf{x}_F, -t) + M_k^+(\mathbf{x}_R, \mathbf{x}_F, t). \tag{15}$$

Using equation (7) and the expression of f_1^+ in equation (15) the update of f_1^- at step k is given by

$$f_{1,k}^-(\mathbf{x}_R, \mathbf{x}_F, t) = f_{1,0}^-(\mathbf{x}_R, \mathbf{x}_F, t) + \theta_t \int_{\partial\mathbb{D}_0} \int_{t'=-\infty}^{t} R(\mathbf{x}_R, \mathbf{x}, t-t') M_k^+(\mathbf{x}, \mathbf{x}_F, t') \mathrm{d}t' \mathrm{d}\mathbf{x}. \tag{16}$$

This completes the definition of the iterative Marchenko scheme. In the next section the first few iterations are discussed in detail and illustrated with simple numerical examples.

3 Marchenko Algorithm

To compute f_1 focusing functions with the Marchenko method two ingredients are needed:

- Reflection data without free-surface multiples, ghosts and deconvolved for the wavelet: $R(\mathbf{x}_R, \mathbf{x}, t)$ with source $\mathbf{x}$ and receiver $\mathbf{x}_R$ on the same surface $\partial\mathbb{D}_0$, and small enough sampling for $\mathbf{x}_R$ and $\mathbf{x}$ to avoid spatial aliasing.
- An estimate of the direct arrival between the receiver positions at the surface ($\mathbf{x}_R$), and the focal point at $\mathbf{x}_F$: $G_d(\mathbf{x}_R, \mathbf{x}_F, t)$, and derived from it the direct arrival time $t_d(\mathbf{x}_R, \mathbf{x}_F, t)$. Note that t_d can also be computed by another method, for example an eikonal solver.

Given these two components the iterative method can be initialised and the iterations of the Marchenko method can start.

3.1 The first few iterations

The initialisation of the method is given by equations (12) and (13). The time-windowed expression for $f^-_{1,0}(\mathbf{x}_R, \mathbf{x}_F, t)$ in equation (13) is renamed to:

$$-N_0(\mathbf{x}_R, \mathbf{x}_F, -t) = \theta_t \int_{\partial\mathbb{D}_0} \int_{t'} R(\mathbf{x}_R, \mathbf{x}, t - t') G_d(\mathbf{x}, \mathbf{x}_F, -t') \mathrm{d}t' \mathrm{d}\mathbf{x}. \tag{17}$$

At each iteration, the spatial integration and temporal convolution with R plays an important role as it is used to define new focusing updates give by terms N_i (see also appendix B of Wapenaar et al. (2014b)). These N_i terms are used to update the estimates of the focusing functions f^+_1 and f^-_1. Although the N_i terms are strictly not needed to describe the method, they are introduced here to remain as close as possible to the actual implementation.
For computational efficiency, the temporal convolution of R is implemented in the Fourier domain. The spatial integration is carried out by summing the resulting traces of the time convolution over a common receiver gather. The introduced time-window sets events for $t > t^{\varepsilon}_d$ to zero, in accordance with equation (17). Applying the mute-window is therefore a crucial and mandatory step in the Marchenko method; without it the method would be incorrect.
Given these initialisations the first step in the algorithm, based on equations (14)-(16), can be computed. This first step, $k = 1$, involves two integration-convolution's with R to update both f^+_1 and f^-_1:

$$\begin{aligned}
M^+_1(\mathbf{x}_R, \mathbf{x}_F, -t) &= \theta_t \int_{\partial\mathbb{D}_0} \int_{t'} R(\mathbf{x}_R, \mathbf{x}, t - t') f^-_{1,0}(\mathbf{x}, \mathbf{x}_F, -t') \mathrm{d}t' \mathrm{d}\mathbf{x} && (18)\\
&= -\theta_t \int_{\partial\mathbb{D}_0} \int_{t'} R(\mathbf{x}_R, \mathbf{x}, t - t') N_0(\mathbf{x}, \mathbf{x}_F, t') \mathrm{d}t' \mathrm{d}\mathbf{x}\\
&= N_1(\mathbf{x}_R, \mathbf{x}_F, -t),\\
f^+_{1,1}(\mathbf{x}_R, \mathbf{x}_F, t) &= G_d(\mathbf{x}_R, \mathbf{x}_F, -t) + M^+_1(\mathbf{x}_R, \mathbf{x}_F, t)\\
&= G_d(\mathbf{x}_R, \mathbf{x}_F, -t) + N_1(\mathbf{x}_F, \mathbf{x}_R, t), && (19)\\
f^-_{1,1}(\mathbf{x}_R, \mathbf{x}_F, t) &= f^-_{1,0}(\mathbf{x}_R, \mathbf{x}_F, t) + \theta_t \int_{\partial\mathbb{D}_0} \int_{t'} R(\mathbf{x}_R, \mathbf{x}, t - t') M^+_1(\mathbf{x}, \mathbf{x}_F, t') \mathrm{d}t' \mathrm{d}\mathbf{x}\\
&= -N_0(\mathbf{x}_R, \mathbf{x}_F, -t) + \theta_t \int_{\partial\mathbb{D}_0} \int_{t'} R(\mathbf{x}_R, \mathbf{x}, t - t') N_1(\mathbf{x}, \mathbf{x}_F, t') \mathrm{d}t' \mathrm{d}\mathbf{x},\\
&= -N_0(\mathbf{x}_R, \mathbf{x}_F, -t) - N_2(\mathbf{x}_R, \mathbf{x}_F, -t), && (20)\\
f_{2,1}(\mathbf{x}_F, \mathbf{x}_R, t) &= G_d(\mathbf{x}_R, \mathbf{x}_F, -t) + N_0(\mathbf{x}_R, \mathbf{x}_F, t) + N_1(\mathbf{x}_R, \mathbf{x}_F, t) + N_2(\mathbf{x}_R, \mathbf{x}_F, t). && (21)
\end{aligned}$$

The first integration-convolution with R in equation (18) is used to update f^+_1 as shown in equation (19). The second integration-convolution in equation (20) updates f^-_1. The update of f_2, introduced in equation (5), includes the results of all integration-convolutions with R.

The next step for $k = 2$ results in the following updates:

$$\begin{aligned} M_2^+(\mathbf{x}_R, \mathbf{x}_F, -t) &= \theta_t \int_{\partial\mathbb{D}_0} \int_{t'} R(\mathbf{x}_R, \mathbf{x}, t - t') f_{1,1}^-(\mathbf{x}, \mathbf{x}_F, -t') \mathrm{d}t' \mathrm{d}\mathbf{x} \qquad (22) \\ &= -\theta_t \int_{\partial\mathbb{D}_0} \int_{t'} R(\mathbf{x}_R, \mathbf{x}, t - t') \{N_0(\mathbf{x}, \mathbf{x}_F, t) + N_2(\mathbf{x}, \mathbf{x}_F, t)\} \mathrm{d}t' \mathrm{d}\mathbf{x} \\ &= N_1(\mathbf{x}_R, \mathbf{x}_F, -t) + N_3(\mathbf{x}_R, \mathbf{x}_F, -t), \\ f_{1,2}^+(\mathbf{x}_R, \mathbf{x}_F, t) &= G_d(\mathbf{x}_R, \mathbf{x}_F, -t) + M_2^+(\mathbf{x}_R, \mathbf{x}_F, t) \\ &= G_d(\mathbf{x}_R, \mathbf{x}_F, -t) + N_1(\mathbf{x}_R, \mathbf{x}_F, t) + N_3(\mathbf{x}_R, \mathbf{x}_F, t), \qquad (23) \\ f_{1,2}^-(\mathbf{x}_R, \mathbf{x}_F, t) &= f_{1,0}^-(\mathbf{x}_R, \mathbf{x}_F, t) + \theta_t \int_{\partial\mathbb{D}_0} \int_{t'} R(\mathbf{x}_R, \mathbf{x}, t - t') M_2^+(\mathbf{x}, \mathbf{x}_F, t') \mathrm{d}t' \mathrm{d}\mathbf{x} \\ &= -N_0(\mathbf{x}_R, \mathbf{x}_F, -t) + \theta_t \int_{\partial\mathbb{D}_0} \int_{t'} R(\mathbf{x}_R, \mathbf{x}, t - t') \{N_1(\mathbf{x}, \mathbf{x}_F, t) + N_3(\mathbf{x}, \mathbf{x}_F, t)\} \mathrm{d}t' \mathrm{d}\mathbf{x} \\ &= -N_0(\mathbf{x}_R, \mathbf{x}_F, -t) - N_2(\mathbf{x}_R, \mathbf{x}_F, -t) - N_4(\mathbf{x}_R, \mathbf{x}_F, -t), \qquad (24) \\ f_{2,2}(\mathbf{x}_F, \mathbf{x}_R, t) &= G_d(\mathbf{x}_R, \mathbf{x}_F, -t) + N_0(\mathbf{x}_R, \mathbf{x}_F, t) + N_1(\mathbf{x}_R, \mathbf{x}_F, t) + \\ & \quad N_2(\mathbf{x}_R, \mathbf{x}_F, t) + N_3(\mathbf{x}_R, \mathbf{x}_F, t) + N_4(\mathbf{x}_R, \mathbf{x}_F, t). \qquad (25) \end{aligned}$$

From these updates it becomes clear that in updating f_1^+ in equation (23) G_d and the odd terms of N_i are used and in updating f_1^- in equation (24) the even terms of N_i are used. The f_2 function in equation (25) is built up from G_d and both even and odd N_i terms.
In the implementation the N_i terms are computed by

$$\begin{aligned} N_{-1}(\mathbf{x}_R, \mathbf{x}_F, -t) &= G_d(\mathbf{x}, \mathbf{x}_F, -t'), \qquad (26) \\ N_i(\mathbf{x}_R, \mathbf{x}_F, -t) &= -\theta_t \int_{\partial\mathbb{D}_0} \int_{t'} R(\mathbf{x}_R, \mathbf{x}, t - t') N_{i-1}(\mathbf{x}, \mathbf{x}_F, t') \mathrm{d}t' \mathrm{d}\mathbf{x}, \qquad (27) \end{aligned}$$

and used to update the focusing functions f_1^+, f_1^- and f_2. This makes the algorithm both simple and efficient. In summary the relations for M_m^+, N_i and the updates for the focusing functions for m iterations with $m \geqslant 1$ are simply:

$$M_m^+(\mathbf{x}_R, \mathbf{x}_F, t) = \sum_{l=0}^{m-1} N_{2l+1}(\mathbf{x}_R, \mathbf{x}_F, t), \qquad (28)$$

$$f_{1,m}^+(\mathbf{x}_R, \mathbf{x}_F, t) = G_d(\mathbf{x}_R, \mathbf{x}_F, -t) + \sum_{l=0}^{m-1} N_{2l+1}(\mathbf{x}_R, \mathbf{x}_F, t), \qquad (29)$$

$$f_{1,m}^-(\mathbf{x}_R, \mathbf{x}_F, t) = -\sum_{l=0}^{m} N_{2l}(\mathbf{x}_R, \mathbf{x}_F, -t), \qquad (30)$$

$$f_{2,m}(\mathbf{x}_F, \mathbf{x}_R, t) = G_d(\mathbf{x}_R, \mathbf{x}_F, -t) + \sum_{l=0}^{2m} N_l(\mathbf{x}_R, \mathbf{x}_F, t). \qquad (31)$$

In the provided program each computation of a focusing update term N_i is called one iteration. The implementation is shown in Algorithm 1 and a flow chart is shown in Figure 2. The initialisation of f_1^-, f_1^+, f_2, and N_i are done just before the iteration starts. The even and odd iterations for N_i update f_1^- and f_1^+ respectively. The Green's function is computed by inserting the estimate of f_2 given by equation (31) into equation (4)

$$\begin{aligned} G(\mathbf{x}_F, \mathbf{x}_R, t) = f_2(\mathbf{x}_F, \mathbf{x}_R, -t) &+ \int_{\partial\mathbb{D}_0} \int_{t'=-\infty}^{t} R(\mathbf{x}_R, \mathbf{x}, t - t') G_d(\mathbf{x}, \mathbf{x}_F, -t) \mathrm{d}t' \mathrm{d}\mathbf{x} \\ &+ \sum_{l=0}^{2m} \int_{\partial\mathbb{D}_0} \int_{t'=-\infty}^{t} R(\mathbf{x}_R, \mathbf{x}, t - t') N_l(\mathbf{x}, \mathbf{x}_F, t') \mathrm{d}t' \mathrm{d}\mathbf{x}. \qquad (32) \end{aligned}$$

In equation (32) the integration-convolution terms can be recognised as the summation of the unmuted N_i terms. By storing the sum of these unmuted terms of the integration-convolution (in p^-) the Green's functions can be calculated as a summation of previously computed values.

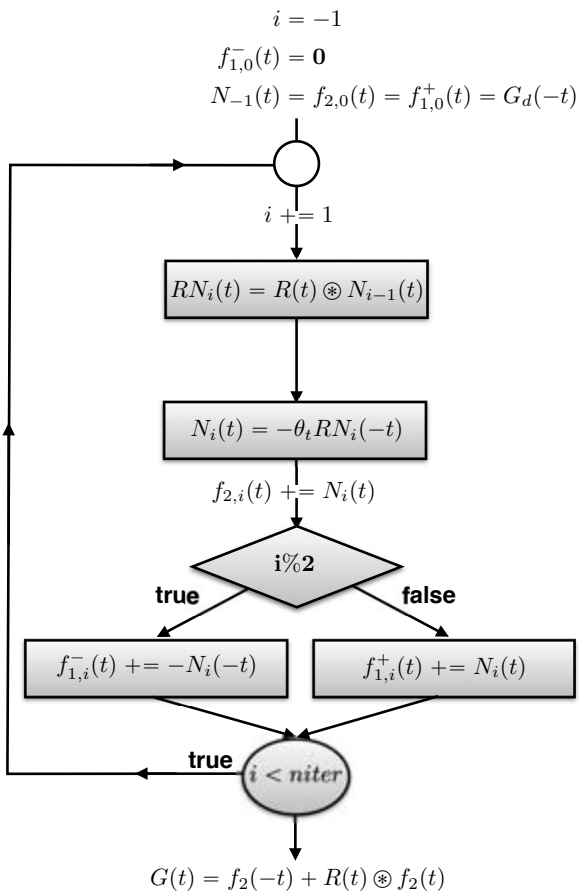


Figure 2: Flow chart of the Marchenko algorithm. In the notation the lateral coordinates are omitted for a more compact notation. The symbol $\circledast$ represents the integration-convolution operator.

The program can compute the results of multiple focal points at the same time (*Nfoc* in Algorithm 1). This is convenient for calculating the Marchenko results (e.g. Green's function) on a depth level, or area of interest in one run. The computational advantage is that the reflection response has to be read in only once to compute the results of multiple focal points. The computations for different focal points are independent of each other. Hence, the code is OpenMP parallelised over the number of focal points (`Nfoc`).

The function `synthesis` in Algorithm 1 computes the integration-convolution, of the focusing update term N_i with R, in the frequency domain (Fourier operator is denoted with $\mathcal{F}$). For the computation of only one focal point, loading the required input data into memory usually takes more time than the computational work. The implementation has additional functionality (not shown in Algorithm 1) to compute the up- and downgoing Green's functions by equations (1) and (2) and write intermediate computed fields (N_i) to disk.

```
Main begin
    Reading SU-style input Data and Allocate arrays
    Initialisation
    Ni(t) = f2p(t) = f1plus(t) = G_d(-t)
    f1min(t) = pmin(t) = 0.0
    for iter ← 0 to niter do
        synthesis(Refl, Ni, iRN)
        Ni(t)=-iRN(-t)
        pmin(t)+=iRN(t)
        applyMute(Ni, muteW)
        f2p(t) += Ni(t)
        else if (iter % 2 == 0) then
            f1min(t) -= Ni(-t)
        else
            f1plus(t) += Ni(t)
        end
    end
    Green(t) = pmin(t) + f2p(-t)
end
synthesis(Refl, Ni, iRN)
begin
    iRN = 0
    ∀l, i: Fop(l,ω,i) = F {Ni(l,i,t)}
    for k ← 0 to nshots do
        #pragma omp parallel for
        for l ← 0 to Nfoc do
            for ω ← ω_min to ω_max do
                sum(ω) = 0
                for i ← 0 to nrecv do
                    sum(ω) += Refl(k,ω,i) * Fop(l,ω,i)
                end
            end
            iRN(l,k,t) = F^-1 {sum(ω)}
        end
    end
end
```

Algorithm 1: Marchenko algorithm as implemented in the provided source code.

3.2 Numerical examples

To use the Marchenko method with numerically modeled data it is very important that the amplitudes of the reflection response are correct. This is certainly also true for field data. Therefore, the importance of amplitude scaling is explained first before discussing the numerical examples in more detail.
In the summation of N_1 and G_d to compute $f^+_{1,1}$ in equation (19), it is important that the amplitude of the measured reflection data is accurate. A wrong amplitude of R will result in a wrong amplitude of $f^+_{1,1}$ and the scheme will not converge. This is illustrated with the following equations. Lets assume that we introduce a wrong scaling factor b in R to update $f^+_{1,i}$. Then the first iterations will compute

$$
\begin{aligned}
-bN_0(\mathbf{x}_R,\mathbf{x}_F,-t) &= \theta_t \int_{\partial\mathbb{D}_0}\int_{t'} bR(\mathbf{x}_R,\mathbf{x},t-t')G_d(\mathbf{x},\mathbf{x}_F,-t')\mathrm{d}t'\mathrm{d}\mathbf{x},\\
-b^2N_1(\mathbf{x}_R,\mathbf{x}_F,-t) &= \theta_t \int_{\partial\mathbb{D}_0}\int_{t'} bR(\mathbf{x}_R,\mathbf{x},t-t')bN_0(\mathbf{x},\mathbf{x}_F,t')\mathrm{d}t'\mathrm{d}\mathbf{x},\\
f^+_{1,1}(\mathbf{x}_R,\mathbf{x}_F,t) &= G_d(\mathbf{x}_R,\mathbf{x}_F,-t) + b^2N_1(\mathbf{x}_R,\mathbf{x}_F,t).
\end{aligned}
$$

The update of $f_{1,1}^{+}$ involves an error of b^2 and in each next update of $f_{1,m}^{+}$ the error in the update N_{2m+1} will grow with b^{2m+2}. A wrong amplitude in G_d is not a problem as the Marchenko equations are linear in the focusing function. An amplitude error can be factored out, and does not change for higher iterations:

$$
\begin{aligned}
-aN_0(\mathbf{x}_R,\mathbf{x}_F,-t) &= \theta_t \int_{\partial\mathbb{D}_0}\int_{t'} R(\mathbf{x}_R,\mathbf{x},t-t')aG_d(\mathbf{x},\mathbf{x}_F,-t')\mathrm{d}t'\mathrm{d}\mathbf{x},\\
-aN_1(\mathbf{x}_R,\mathbf{x}_F,-t) &= \theta_t \int_{\partial\mathbb{D}_0}\int_{t'} R(\mathbf{x}_R,\mathbf{x},t-t')aN_0(\mathbf{x},\mathbf{x}_F,t')\mathrm{d}t'\mathrm{d}\mathbf{x},\\
af_{1,1}^{+}(\mathbf{x}_R,\mathbf{x}_F,t) &= aG_d(\mathbf{x}_R,\mathbf{x}_F,-t) + aN_1(\mathbf{x}_R,\mathbf{x}_F,t).
\end{aligned}
$$

van der Neut et al. (2015c) introduces an adaptive amplitude-correction factor to correct for possible amplitude errors in R. By solving the Marchenko equation in an explicit series the sensitivity of amplitude errors can be adjusted by adaptive subtraction of the focusing update terms. This approach makes it better suited to apply to field data (van der Neut et al., 2015a; Staring et al., 2016).
Brackenhoff (2016); Thomsen (2016) have developed estimation methodologies for a constant scaling factor b of R. These methods compensate for an overall amplitude mismatch in R, which is an important first step to apply the Marchenko method on measured data. Brackenhoff's method, for example makes use of the fact that $G^{-}(\mathbf{x},\mathbf{x}_F,t)$ is identically zero for a point $\mathbf{x}_F$ below the deepest reflector.
The following will provide step by step directions how to compute the reflection response with accurate amplitudes.

- The reflection data must be deconvolved for the wavelet(Mildner et al., 2017). The result of this deconvolution is the reflection response of a zero-phase wavelet with a flat spectrum between the frequencies f_{min} and f_{max}. Since we are computing the reflection response we can avoid deconvolution and directly model the reflection response with a source signature that has a flat frequency spectrum of amplitude 1.0:

$$
s(t) = \int_{f_{min}}^{f_{max}} 1.0\exp(-j2\pi ft)df \tag{33}
$$

  The implemented flat wavelet spectrum has smooth transitions (cosine taper) to the minimum, and from the maximum, frequency to avoid a very long wavelet in the time domain. The provided program `makewave` can generate these waveforms and the provided scripts give the `makewave` parameters used to calculate the source wavelet. Note, in the discrete implementation of the computation of the source wavelet in the frequency domain one must not forget to multiply with the frequency interval Δf, when going from frequency to time with the Fourier transform. The source wavelet used in the examples is shown in Figure 3. A shift of 0.3 seconds (parameter setting `t0=0.3` in `makewave`) is added to the source wavelet to make it causal and suitable to use in the finite-difference program. In the finite difference modeling of the reflection response the recording of the data is postponed with 0.3 seconds (parameter setting `rec_delay=0.3` in `fdelmodc`) to set the peak of the wavelet back at the correct time.

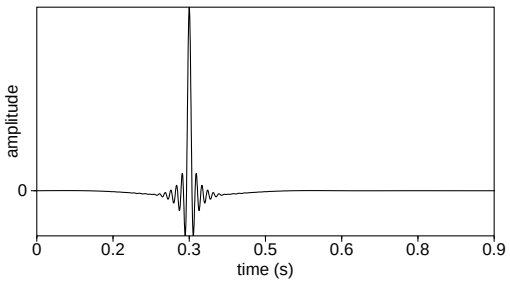
- In the finite-difference program for modeling $R(\mathbf{x}_R,\mathbf{x},t)$ an F_z source of vertical force is chosen (see manual of the finite-difference modeling program `fdelmodc` for an explanation about the options). The receivers are placed at the same surface as the source and measure the pressure field.

- The amplitude scaling factor, in the finite-difference scheme for an F_z source with time signature $s(t)$, is defined in the update of particle velocity V_z as

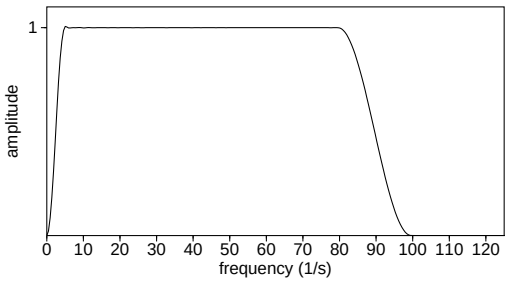
$$
V_z(x,z,t+\Delta t) = V_z(x,z,t) - \frac{\Delta P(x,z,t)}{\rho\Delta z} + \frac{\Delta t}{\rho\Delta x^2}s(t). \tag{34}
$$

  The discrete intervals $\Delta t, \Delta x = \Delta z$ are the steps in the finite-difference program and ρ the local density at the injection grid-point of the source. $\frac{\Delta P}{\Delta z}$ is a fourth order finite-difference implementation of the first derivative to z of pressure field P.

- To compute R, from the Green's functions calculated by the finite-difference program, only a factor -2 is needed (equation (10) in Wapenaar et al., 2012). This factor -2 is included in the `marchenko` program when it reads in the reflection response R.



a) Source wavelet for modeling R.



b) Amplitude spectrum of source wavelet.

Figure 3: Source wavelet with a flat frequency spectrum between f_{min}(=5 Hz) and f_{max}(=80 Hz) used to model the reflection response.

- The time convolution of R is implemented by a forward Fourier transformation from time to frequency domain, multiplication in the frequency domain, and back transformation to the time domain. In the numerical implementation the multiplication with Δt, for the convolution in time and with Δx for the integration over x, must be included as well. Together with the standard scaling factor of $\frac{1}{N}$ for discrete Fourier transformations when going from time to frequency and back to time, with N the number of time samples, the scale factor to compute the time-convolution and space integration in the frequency domain becomes:

$$\frac{\Delta x \Delta t}{N}.$$

3.2.1 Building up the Green's function

The Marchenko algorithm is illustrated with a 2-dimensional horizontally layered model shown in Figure 4. The numerical modeling is carried out with a finite difference modeling program (Thorbecke and Draganov, 2011) that is also included in the software package. The input source signature used to model the reflection response $R(\mathbf{x}_R, \mathbf{x}, t)$ is approximately a sinc-function with a flat spectrum of amplitude 1, as shown in Figure 3.
The full reflection matrix $R(\mathbf{x}_R, \mathbf{x}, t)$, for a fixed-spread geometry, can be constructed from one forward modeled shot (Figure 4c), since the model contains no lateral variations. The constructed fixed-spread geometry ranges from -2250 to 2250 meter with 5 meter distance between source and receiver positions. The 5 meter distance is chosen to avoid spatial aliasing. We use a laterally invariant medium, because the time to compute the reflection response R in a laterally variant medium is too large to be practical for the desired reproducibility of the examples in this paper. However, the Marchenko method does not make any assumption about the medium and can handle lateral variations as well. Moreover, the demo directory of the Marchenko program contains also an example with lateral variations in the model (`marchenko/demo/twoD`).
The transmission response, recorded at the surface for a source at 900 m depth is shown in Figure 4d. It has been modeled with a zero-phase Ricker source wavelet s(t) that has its peak at 25 Hz. It is important that the chosen source wavelet to model G_d be zero-phase, otherwise the time reversals applied in the algorithm would not work properly and the Marchenko scheme would not converge. It is also preferable to choose a source wavelet that decreases rapidly in time. This is to minimise the occurence of overlap

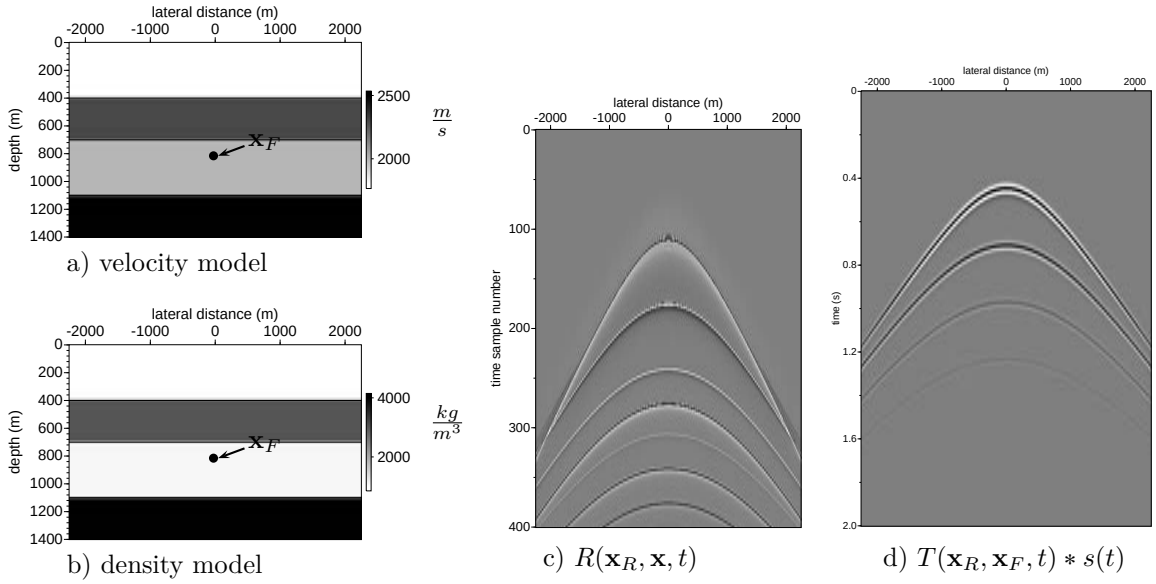


Figure 4: Four layer model with velocity (a) and density (b) contrasts. A shot record, with source position $\mathbf{x}(x = 0, z = 0)$ and receivers at $\mathbf{x}_R(x = x_r, z = 0)$ (c), and the transmission response from a source at $\mathbf{x}_F(x = 0, z = 900)$ (d). Note that the source wavelet used to compute R (c) is given by Figure 3 and T (d) is modeled with a Ricker wavelet with a peak at 25 Hz.

between the direct arrival and the first reflections as is assumed in equation (10). In case of an overlap, the defined window-function (θ_t in equation (11)) cuts through the overlapping events and the first reflection is not retrieved correctly.

The initialisation step used to compute $f_{1,0}^-$ (equation (17)) is illustrated in Figure 5. Each shot record in $R(\mathbf{x}_R, \mathbf{x}, t)$ is convolved with $G_d(\mathbf{x}_R, \mathbf{x}_F, -t)$, where $G_d(\mathbf{x}_R, \mathbf{x}_F, -t)$ shown in Figure 5 only contains the time-reversal of the full transmission response shown in Figure 4d. By making use of shift invariance $R(\mathbf{x}_R, \mathbf{x}, t) = R(\mathbf{x}_R - \mathbf{x}, 0, t)$, the time-convolution result is integrated (summed over all receiver positions $\mathbf{x}_R$) and results in one trace at the $\mathbf{x}$ position in the N_0 panel.

In $-N_0(\mathbf{x}, \mathbf{x}_F, -t)$ the dotted lines indicate the cut-off boundaries of the implemented time window $\theta(\mathbf{x}, \mathbf{x}_F, t)$. To suppress wrap-around events (from positive times wrapping to negative times) the time window $\theta(\mathbf{x}, \mathbf{x}_F, t)$, as introduced in equation (11), is symmetrized. Hence, from here onward $\theta(\mathbf{x}, \mathbf{x}_F, t)$ is zero for $t > t_d^\varepsilon$ and $t < -t_d^\varepsilon$, and unity for times inside $-t_d^\varepsilon < t < t_d^\varepsilon$. For deep focal points one can also extend the time axis by padding zeros at the end of the array and in that way avoid the influence of wrap-around events in the time domain. In the Appendix the treatment of time wrap-around is explained in more detail.

The events before the top dotted line and the events after the bottom dotted line are muted. The two remaining events originate from the two reflectors above the chosen focal point at 900 meter depth. A detailed explanation of the different events in the focusing functions is given by Van der Neut et al. (2015b). Staring et al. (2016) give a similar explanation in case free-surface multiples are included in the Marchenko method. This initialisation of f_1^- is the input of the next step to compute a first estimate of f_1^+, given in equations (18) and (19).

The computation of $f_{1,1}^+$ involves the same time convolution and spatial integration operation and is illustrated in Figure 6. The result of the integration and convolution; $-N_1(\mathbf{x}, \mathbf{x}_F, -t)$ is, according to equation (19), time reversed, multiplied by -1 and added to $G_d(\mathbf{x}, \mathbf{x}_F, -t)$ to get the first estimate $f_{1,1}^+$. Note, that the lower (causal) part of the time window $\theta(\mathbf{x}, \mathbf{x}_F, t)$ mutes also the event at direct arrival time. This event at the direct arrival time t_d will end up in the update of the Green's function and will adjust the amplitude of the direct arrival in the Green's function (Van der Neut et al., 2015b). This update of the direct arrival in the Green's function is explained in more detail below.

Figure 7 shows the results of the first 4 iterations of the Marchenko method. The first column represents the results of each convolution and integration of the focusing update term N_i with R. From these figures (all with the same clipping factor) one can observe that the amplitude of N_i becomes smaller with each

next iteration.
The trace in the fifth column is a comparison between the reference Green's function (solid gray) and the computed Green's function (dotted black). In these traces one can observe (indicated with arrows) that some events are weakened by subsequent iterations: The computed Green's function converges to the reference Green's function.
To get a better understanding of the computation of the Green's function, the first few iterations are discussed in more detail. The initialisation of the method starts with G_d (equation (26)) and G is computed according to equation (32). This gives

$$\begin{aligned} f_{2,0}(\mathbf{x}_F, \mathbf{x}_R, t) &= G_d(\mathbf{x}_R, \mathbf{x}_F, -t) \\ G_0(\mathbf{x}_R, \mathbf{x}_F, t) &= G_d(\mathbf{x}_R, \mathbf{x}_F, -t) + \int_{\partial\mathbb{D}_0} \int_{t'=-\infty}^{t} R(\mathbf{x}_R, \mathbf{x}, t-t') G_d(\mathbf{x}, \mathbf{x}_F, -t) \mathrm{d}t' \mathrm{d}\mathbf{x} \\ &\quad + N_0(\mathbf{x}_R, \mathbf{x}_F, -t) \end{aligned} \tag{35}$$

Note that in equation (35) the result of the first integration-convolution with R is **not** muted with θ_t. The initial estimate of the Green's function is thus built up of three terms:

1. The direct arrival of the transmission response ($G_d(\mathbf{x}_R, \mathbf{x}_F, -t)$).
2. The integration-convolution of R with G_d, this is the (unmuted) top left panel in Figure 7.
3. A θ_t muted and multiplied by -1 version of the integration-convolution of R with G_d: $N_0(\mathbf{x}_R, \mathbf{x}_F, -t) = -f_{1,0}^{-}(\mathbf{x}_R, \mathbf{x}_F, t)$, the second panel in the top row of Figure 7 multiplied by -1.

It is important to note that the result of the combination of the second and third term just subtracts $f_{1,0}^{-}(t)$ (the events within the black-dotted lines) from the unmuted integration-convolution of R with G_d. This is the same as the inverse operation of the time-window θ_t. To get the first estimate of the Green's function, $G_d(\mathbf{x}_R, \mathbf{x}_F, -t)$ is added to this result and gives the top right panel in Figure 7. In the next iteration we have

$$\begin{aligned} f_{2,1}(\mathbf{x}_F, \mathbf{x}_R, t) &= G_d(\mathbf{x}_F, \mathbf{x}_R, -t) + N_0(\mathbf{x}_F, \mathbf{x}_R, t) \\ G_1(\mathbf{x}_F, \mathbf{x}_R, t) &= G_0(\mathbf{x}_R, \mathbf{x}_F, t) + \int_{\partial\mathbb{D}_0} \int_{t'=-\infty}^{t} R(\mathbf{x}_R, \mathbf{x}, t-t') N_0(\mathbf{x}, \mathbf{x}_F, t) \mathrm{d}t' \mathrm{d}\mathbf{x} \\ &\quad + N_1(\mathbf{x}_R, \mathbf{x}_F, -t) \end{aligned} \tag{36}$$

Compared to the previous iteration two new terms are added:

1. The integration-convolution of R with N_0, this is the (unmuted) left panel for $i = 1$ in Figure 7.
2. The θ_t muted, time reversed, multiplied by -1 version of the integration-convolution of R with N_0: $N_1(-t)$.

The combination of these two terms results in the subtraction of the events within the black-dotted lines from the unmuted integration-convolution of R with N_0.
Each next iteration follows this same pattern: the events within the time window θ_t (above t_d^ε) are used to update the f_1 focusing function, the events outside the time window θ_t (below t_d^ε) are used to update the Green's function. Application of the window function θ_t separates the focusing function and the Green's function.
There is one important remark to make; the direct arrival T_d^{inv} in the focusing function f_1^+ is not updated, whereas the direct arrival G_d in the Green's function G is updated. In the first iteration (top row in Figure 7) the direct arrival in the Green's function G_0 is equal to G_d. In the second iteration shown in Figure 7 the amplitude of the direct arrival is corrected by the event in the unmuted $-N_1(-t)$ just below the black-dotted line of the mute window. This event in the unmuted $-N_1(-t)$ has an opposite sign to the direct arrival and decreases the amplitude of the direct arrival (Van der Neut et al., 2015b). In the plotted trace of G_0 the amplitude of the direct arrival (dotted line) is much higher than the reference (gray line). In G_1 the amplitudes of the direct arrival between reference and computed Green's function are already much closer. We do not expect that the scheme started with the approximation $T_d^{inv} \approx G_d$ will arrive at the correct amplitudes; in order to achieve accurate amplitudes the inverse of the transmission transpose had to be used and not G_d. There is an offset dependent scaling factor between the reference and the

computed Green's function. Thorbecke et al. (2013) showed that this estimate of the direct arrival does not have to be precise and can be based on a macro model. The relative amplitudes between the events of the computed Green's function is correct and shown in the trace comparison with the reference output in Figure 7.

The iterative corrections of the amplitude of the Green's function are needed to take into account transmission losses. The result is that the upgoing field that arrives at $t = t_d$ has an amplitude that is equal to the local reflection coefficient of depth level z_f (Slob et al., 2014). In the next section we will see how good this correction is, when the f_2 focusing function is emitted into the medium.

Broggini et al. (2014) use the energy before the direct arrival in the Green's function to define the convergence of the scheme. In the provided Marchenko program there is no stopping criteria built in, to give the user the freedom to choose the number of iterations carried out. The energy in the focusing update term ($\sqrt{\sum_{x,t} N_i^2(x,t)}$) is computed and printed for each iteration and can be monitored for convergence. In each next iteration this energy should become smaller. The convergence rate for 8 iterations is shown in Figure 8.

A comparison with the reference Green's function and the Marchenko computed Green's function after 8 iterations is shown in Figure 9. The difference with the reference Green's function is negligible in the middle part of the picture around $x = 0$. A small amplitude mismatch increases slightly with increasing offset. Closer to the edge of the acquisition ($\pm$2250 meter) the difference with the reference becomes larger, because the full Fresnel zone is not included in the acquisition. The higher wavenumbers, more present at earlier times, are also not captured by the limited acquisition. Note that after $\sim$ 1.5 seconds, the presence of higher wavenumbers becomes smaller, and the amplitude error at the far offsets also decreases. To suppress artefacts from limited acquisition aperture, tapers can be applied to the edges of the initial focusing operator and/or the reflection response. In our experience these tapers have limited effects on suppressing these artefacts. Depending on the specific events at the boundaries of the model the finite aperture effect could slightly be attenuated. In some cases the taper shifts the problem to the non-tapered part adjacent to the tapered region and finite aperture artefacts remain. Another, usually smaller, amplitude mismatch is caused by the use of the time-reversal of the direct arrival in the transmission response G_d (equation (10)) instead of the inverse.

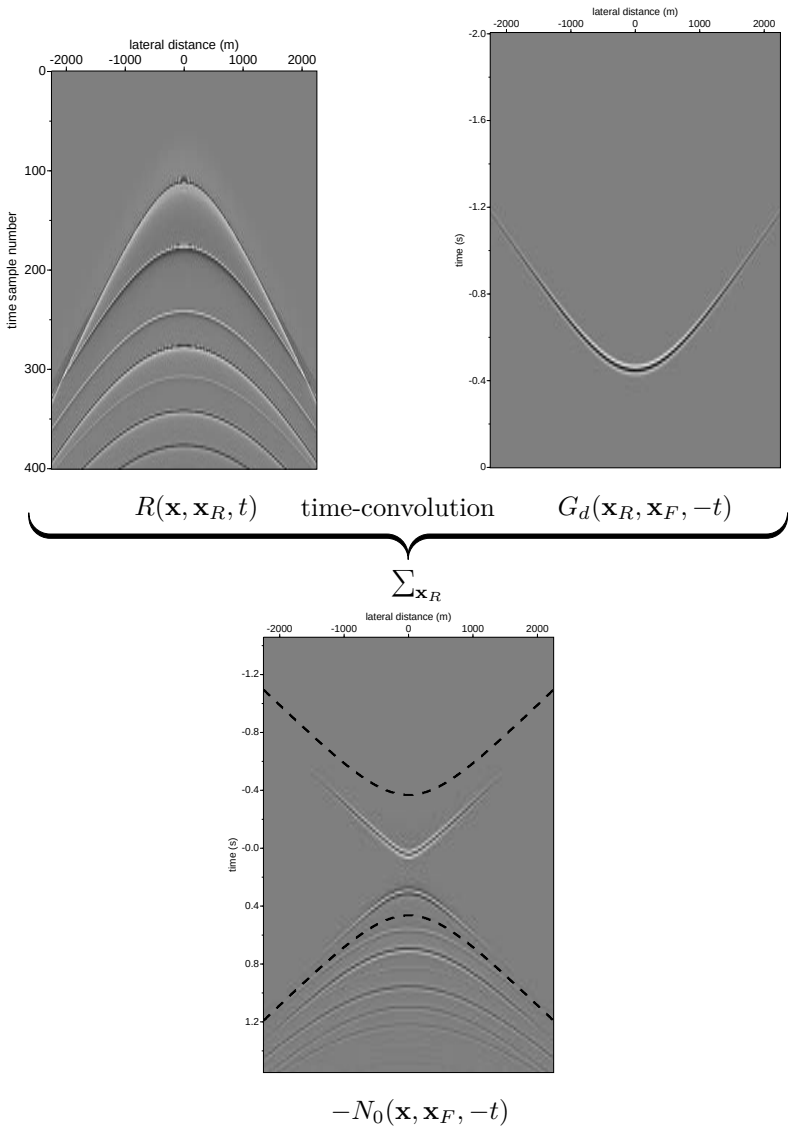


Figure 5: Initialisation step to compute $f_{1,0}^{-}(\mathbf{x}, \mathbf{x}_F, t) = -N_0(\mathbf{x}, \mathbf{x}_F, -t)$. After applying the time window $\theta(\mathbf{x}, \mathbf{x}_F, t) = \theta_t$ only events between the dotted lines remain in N_0. The mute window at $t < 0$ is applied to mute the wrap-around events of the temporal convolution. This extra window at $t < 0$ is only a practical solution and not needed from the theory. Note the difference in the time axes of the three panels; positive for $R(t)$, negative for $G_d(-t)$ and negative and positive for $N_0(-t)$).

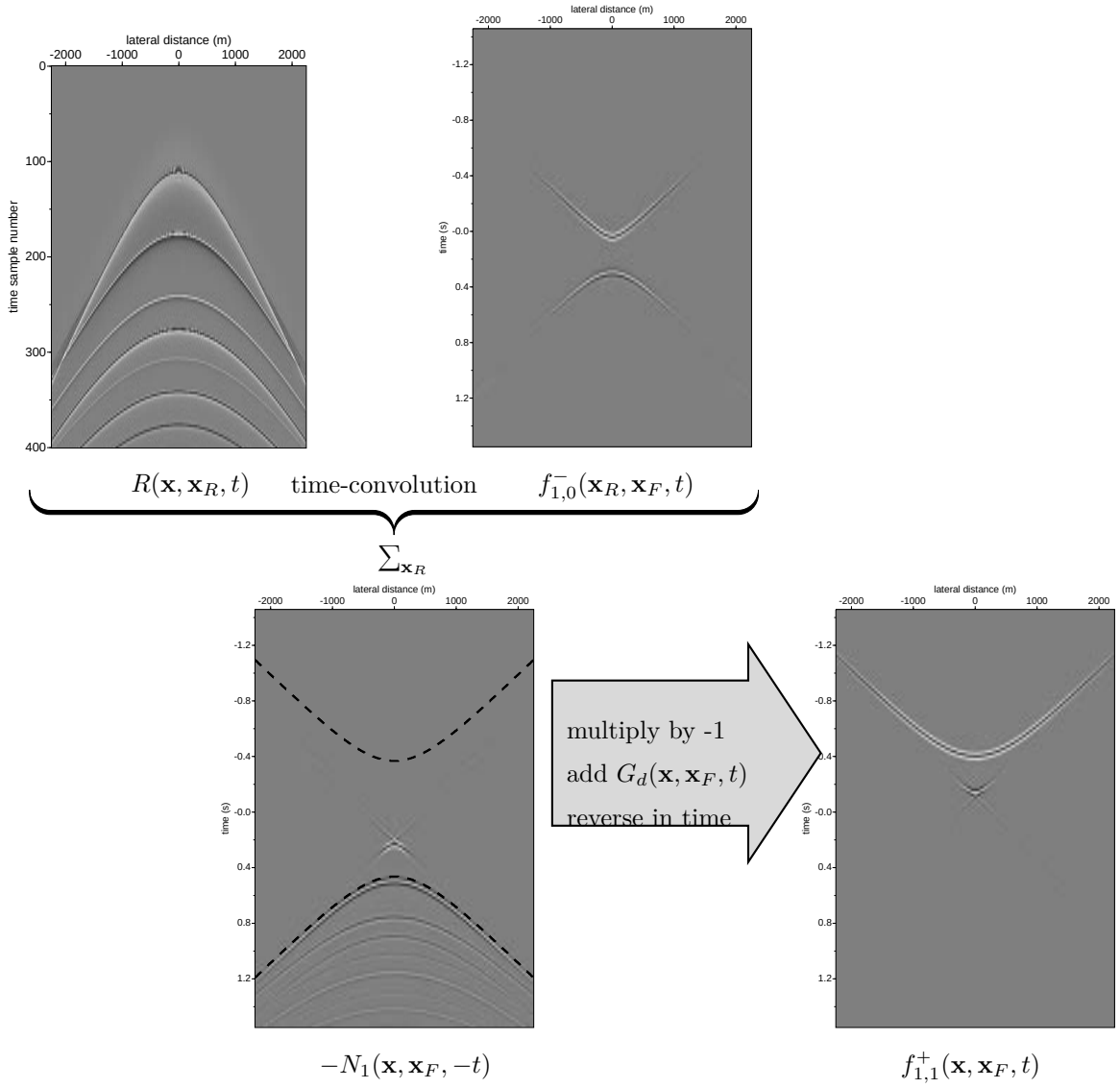


Figure 6: First iteration to compute $f^+_{1,1}(\mathbf{x}, \mathbf{x}_F, t)$ from $f^-_{1,0}(\mathbf{x}_R, \mathbf{x}_F, t)$. In the summation of G_d with N_1 it is important that the amplitudes of R are correct.

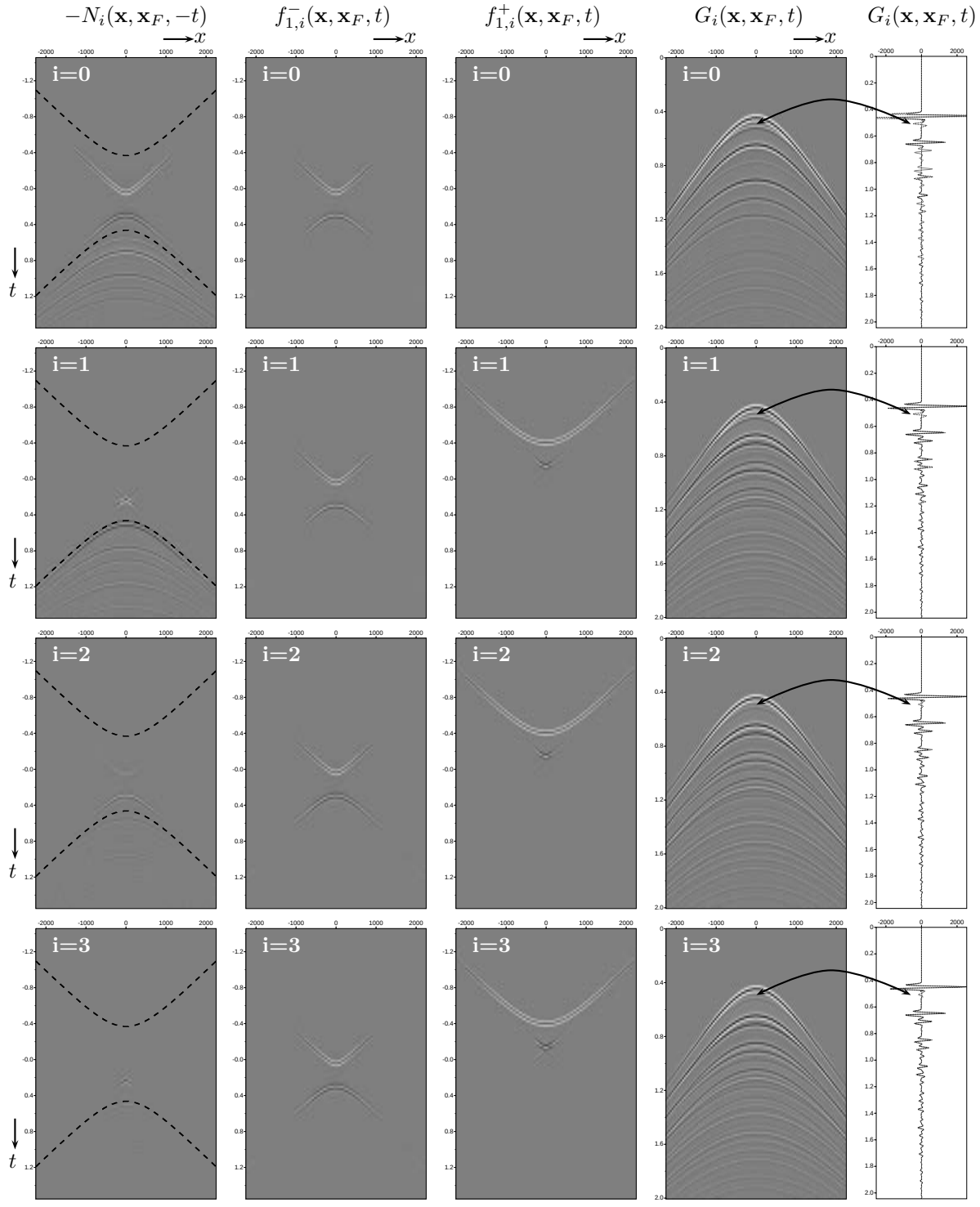


Figure 7: Four successive iterations of the Marchenko method. The arrows point to an event that does not belong to the Green's function and is weakened at each iteration. $f^-_{1,i}$ (the 2nd column) only changes from $i = 1$ to $i = 2$, while $f^+_{1,i}$ (the 3rd column) changes from $i = 0$ to $i = 1$ and from $i = 2$ to $i = 3$. The clip level for N_i and G_i is the same for all panels. Labels of the horizontal and vertical axes are the same for all panels, and are shown for the top and left panels.

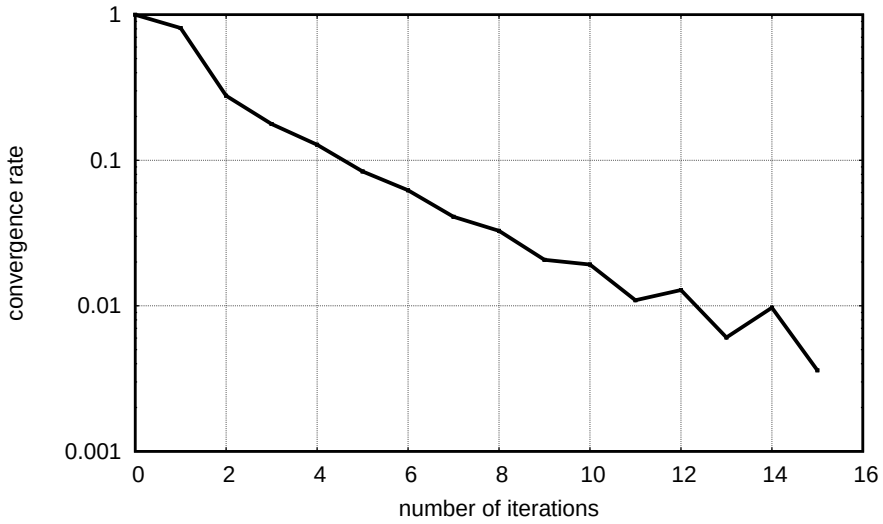


Figure 8: Logarithmic convergence rate of the `marchenko/demo/oneD` example for 16 iterations. The bumps at the end of the curve are caused by limited aperture artefacts. These artefacts are 2 orders of magnitude smaller than the main events.

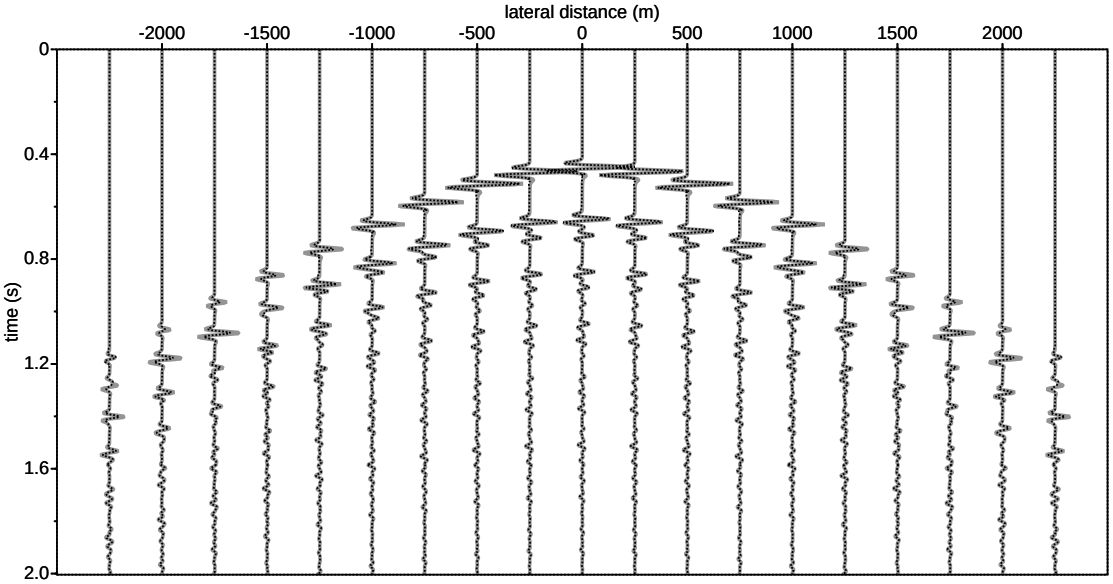


Figure 9: Comparison of the Marchenko computed Green's function after 8 iterations with the reference Green's function: solid-gray trace in the background is the reference and dotted-black trace the Green's function computed with the Marchenko method.

3.2.2 Propagating focusing function

One of the properties of the defined $f_2(\mathbf{x}_F, \mathbf{x}, t)$ focusing function in equation (5) is that it will focus at $t = 0$ at the focal point $\mathbf{x}_F$. This property can be demonstrated by emitting $f_2(\mathbf{x}_F, \mathbf{x}, t)$ from $\partial\mathbb{D}_0$ into the medium and show that it has a focus at position $\mathbf{x}_F$ at snapshot $t = 0$ (Singh et al., 2016; Wapenaar et al., 2017). If the transmission losses in the events in f_2 have correctly been taken into account then all internal multiples will be cancelled at (and only at) $t = 0$. The left column of Figure 10 shows 5 snapshots at times $-0.30, -0.15, -0.03, -0.02$, and 0.0. The snapshot at $t = 0$ indeed shows only a focus at the focal point. In the snapshots at times $t = -0.03, t = -0.02$ it is observed that events related to internal multiples have opposite amplitude and travel towards each other to cancel out at $t = 0$.
The second column of Figure 10 shows snapshots at positive times, after the wavefield has focussed at $t = 0$. After $t = 0$ the focussed and dimmed events separate again and continue their path.

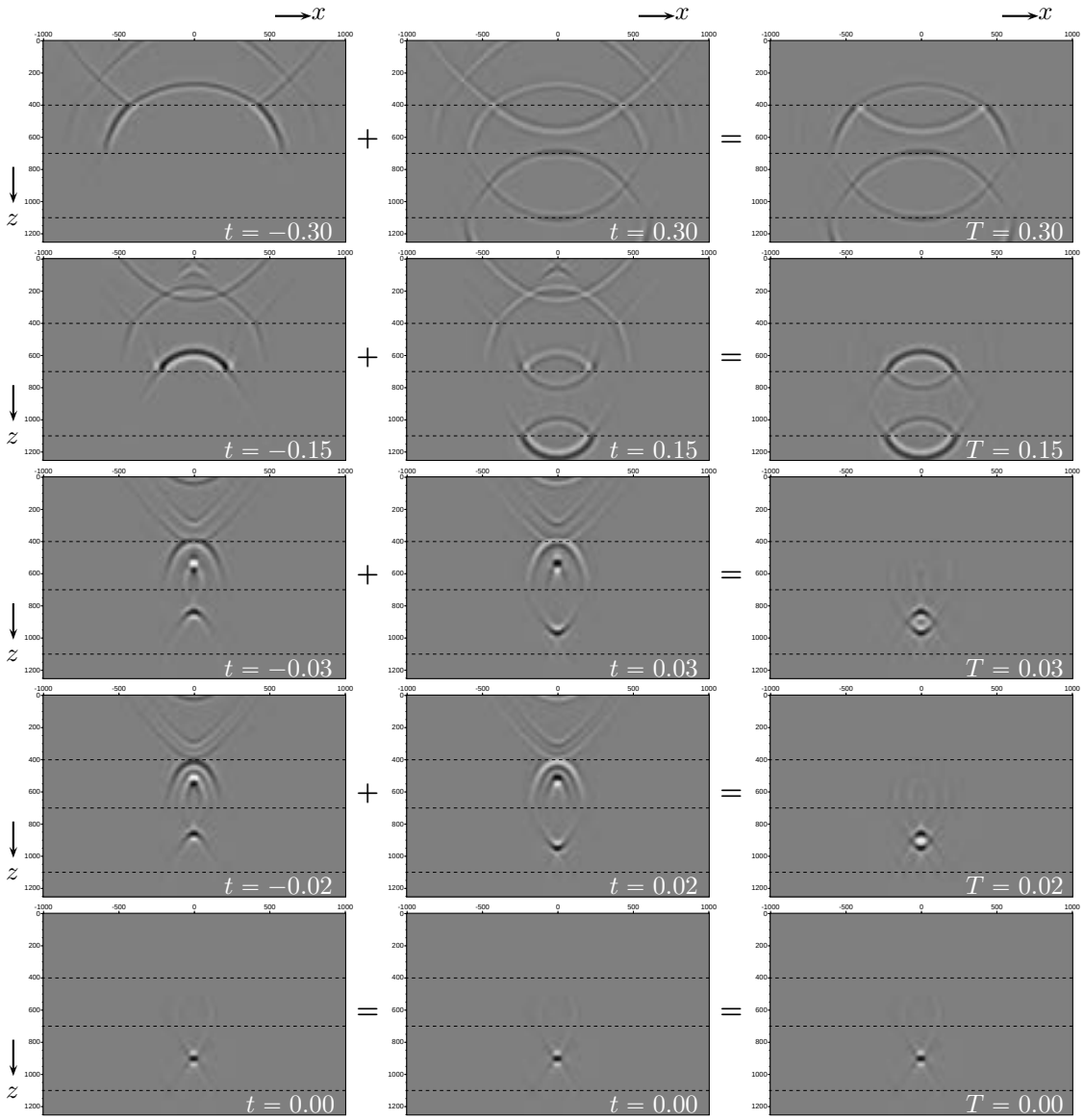


Figure 10: Snapshots of propagation of focusing function f_2 through the actual medium. The left column shows snapshots at a-causal times, the middle column snapshots at causal times. The rightmost column shows the addition of the acausal snapshots at negative times with the corresponding causal snapshots at positive times (time T). Labels of the horizontal and vertical axes are the same for all panels, and are shown for the top and left panels.

Adding the snapshots at negative times to the corresponding snapshots at positive times defines the snapshots of the homogeneous Green's function (Wapenaar et al., 2016a) with a virtual sourse at $\mathbf{x}_F$. The third column in Figure 10 shows these combined snapshots, where the snapshots at positive and negative times are summed, and represent the causal part of the homogeneous Green's function. These snapshots can be interpreted as the response of a virtual source located at the position of the focal point $\mathbf{x}_F$.

3.3 Parameters in program `marchenko`

The self-doc of the program is shown by typing `marchenko` on the command line without any arguments. You will then see the following list of parameters:
If you are not considering special cases, the default values are most of the times sufficient and only a few parameters have to be changed from their default values to get started.
The provided marchenko source-code package contains two main programs:

- `fmute`: picks the first arrival time from a transmission response and mutes along this time
- `marchenko`: solves for the focusing functions in the Marchenko method and computes Green's functions

The `fmute` program tracks the first arrival from a transmission response to a focal point in the subsurface. Its main use is to separate the direct arrival of the transmission response (G_d) from the multiple scattering coda (M^+). In the examples provided the transmission response is also computed by finite difference modeling and the direct arrival needs to be separated from the coda. For example `fmute` is used to compute $G_d(t)$ from $T(t)$ in Figure 4d. The program `fmute` is not needed if a method is used (e.g. eikonal solver) that computes the direct arrival in a direct way. The output G_d of the `fmute` program is the input `file_inv` of the `marchenko` program. The different parameters of `fmute` are shown in the self-documentation of the program:

```
fmute - mute in time domain file_shot along curve of maximum amplitude in file_mute

fmute file_shot= {file_mute=} [optional parameters]

Required parameters:

  file_mute= ................. input file with event that defines the mute line
  file_shot= ................. input data that is muted

Optional parameters:

  file_out= .................. output file
  above=0 .................... mute after(0), before(1) or around(2) the maximum times of file_mute
  ........................... options 4 is the inverse of 0 and -1 the inverse of 1
  shift=0 .................... number of points above(positive) / below(negative) maximum time for mute
  check=0 .................... plots muting window on top of file_mute: output file check.su
  scale=0 .................... scale data by dividing through maximum
  hw=15 ...................... number of time samples to look up and down in next trace for maximum
  smooth=0 ................... number of points to smooth mute with cosine window
  verbose=0 .................. silent option; >0 display info
```

If `file_mute` is not provided, `file_shot` will be used instead to pick the first arrival times.
The `above` option is explained in Figure 11 and separates in different ways the direct arrival time (t_d) from the coda . The `above=0` and `above=4` options have also a truncation point at the lower end of the time-axis, with the time reversal of t_d, to mute wrap-around events introduced by the periodicity of the discrete Fourier transform. Note that the lower end of the time-axis can also represent negative times. The `above=2` option defines a passing window around t_d and is convenient to select the direct arrival of the transmission response in case the first arrival also contains head-waves.
To find the first arrival time in `file_mute` a simple tracking algorithm is implemented. At the trace position equal to the source position the algorithm searches for the maximum value in that trace. It is assumed that this is the first arrival time at the source position. For complex models this might not be true and it is therefore always good to enable `check=1` and verify in the created output file `check.su` if the program has tracked the correct direct arrival time. Starting at the time-sample position of the

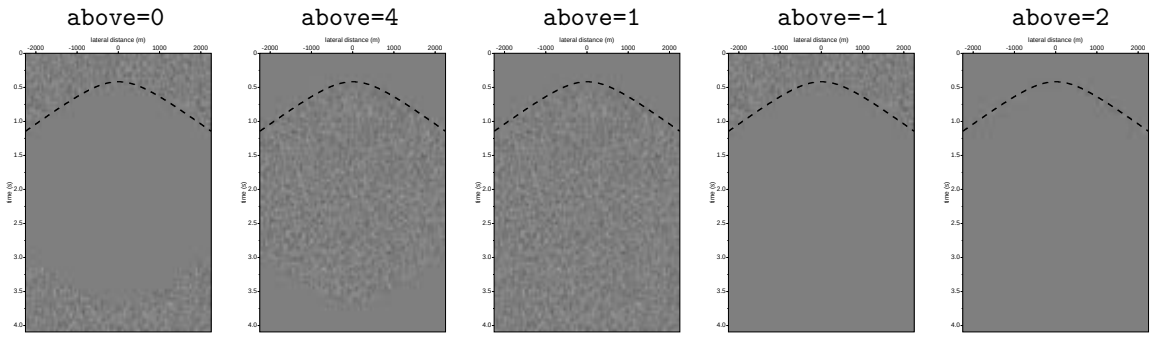


Figure 11: The different options of the `above` parameter in the `fmute` and `marchenko` programs, illustrated with a shot panel consisting of noise.

maximum (j_{max}) in the source trace i the algorithm looks in neighbouring traces ($i \pm 1$) for the maximum. It only searches for this maximum in a restricted time window. For example the maximum in the left trace is searched in the time window $j_{max} - \texttt{hw} < t_{i-1} < j_{max} + \texttt{hw}$, where `hw` is a number of samples given as input parameter. If there are head-waves present the search algorithm can lose track of the direct arrival, so it is good practice to choose a small `hw` (4-8 samples).

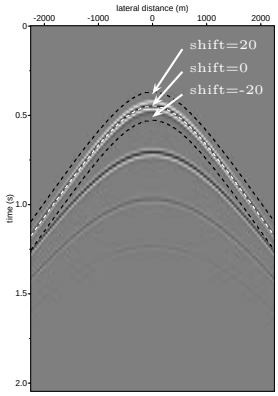


Figure 12: The `shift` parameter in the `fmute` and `marchenko` programs.

The `shift` option represents the ε in t_d^{ε} and is needed to include the width of the wavelet in the mute-window. Figure 12 shows the effect of setting a negative or positive shift to exclude or include the width of the wavelet. With the `above=-1` option a positive shift will mute the direct arrival while a negative will preserve the direct arrival.
The parameter `smooth` defines a transition zone (in samples) going from 1 to 0 in the mute-window. Using a few time-samples (3-5) for the smooth transition zone is enough to give satisfactory results. The direction of the taper, going from 1 to 0, is away from $\pm t_d$.
The `marchenko` program has the following parameters and options:

```
MARCHENKO - Iterative Green's function and focusing functions retrieval

marchenko file_tinv= file_shot= [optional parameters]

Required parameters:

  file_tinv= ............... direct arrival from focal point: G_d
  file_shot= ............... Reflection response: R

Optional parameters:

INTEGRATION
  tap=0 .................... lateral taper focusing(1), shot(2) or both(3)
```

```
  ntap=0 ................... number of taper points at boundaries
  fmin=0 ................... minimum frequency in the Fourier transform
  fmax=70 .................. maximum frequency in the Fourier transform
MARCHENKO ITERATIONS
  niter=10 ................. number of iterations
MUTE-WINDOW
  above=0 .................. mute above(1), around(0) or below(-1) the first travel times of file_tinv
  shift=12 ................. number of points above(positive) / below(negative) travel time for mute
  hw=8 ..................... window in time samples to look for maximum in next trace
  smooth=5 ................. number of points to smooth mute with cosine window
  plane_wave=0 ............. enable plane-wave illumination function
  src_angle=0 .............. angle of plane source array
  src_velo=1500 ............ velocity to use in src_angle definition
REFLECTION RESPONSE CORRECTION
  tsq=0.0 .................. scale factor n for t^n for true amplitude recovery
  Q=0.0 .....,.............. Q correction factor
  f0=0.0 ................... ... for Q correction factor
  scale=2 .................. scale factor of R for summation of Ni with G_d
  pad=0 .................... amount of samples to pad the reflection series
  reci=0 ................... 1; add receivers as shots 2; only use receivers as shot positions
  countmin=0 ............... 0.3*nxrcv; minumum number of reciprocal traces for a contribution
OUTPUT DEFINITION
  file_green= .............. output file with full Green function(s)
  file_gplus= .............. output file with G+
  file_gmin= ............... output file with G-
  file_f1plus= ............. output file with f1+
  file_f1min= .............. output file with f1-
  file_f2= ................. output file with f2 (=p+)
  file_pplus= .............. output file with p+
  file_pmin= ............... output file with p-
  file_iter= ............... output file with -Ni(-t) for each iteration
  rotate=1 ................. 1: t=0 at nt/2 (middle) 0: t=0 at sample 0 for f1+,-
  verbose=0 ................ silent option; >0 displays info
```

The number of iterations required for convergence depends on the reflection strengths and on the number of events in the model; a complex model will need more iterations. Typically the number of iterations is between 8 and 20. An automatic stopping criterion could be based on the energy in the focusing update N_i. This stopping criterion is not implemented to give the user the freedom to choose the number of iterations.

To suppress artefacts from a limited acquisition aperture, tapers can be applied to the edges of the initial focusing operator (`tap=1`) and/or the reflection response (`tap=2`). In our experience these tapers have limited effects on suppressing the finite-acquisition related artefacts and tapering is usually not enabled. The mute-window parameters have the same meaning as in the `fmute` program.

The temporal convolution of events at positive times in the focusing update term causes events in R to be shifted forward in time. Events at negative times will shift events in R backward in time. In the Marchenko method it is important that these backward shifted events are properly handled. For deeper focal points some events can be shifted to negative times. By implementing the temporal convolution in the frequency domain, we make use of the periodic property of the discrete Fourier transformation: negative times wrap-around to the end of the discrete time axis.

The reason to symmetrise the time window θ_t is to suppress unwanted time wrap-around effects. The time-wrap-around effects can also be avoided by padding zeros to the time traces in R; making the time traces `2*nt` long, where the last `nt` samples are zeros. The parameter `pad` will pad zeros to the time traces of R. Adding extra time samples will lead to longer computation times. Therefore, we prefer to use a symmetrised time window to suppress the unwanted effects of time wrap-around.

The `scale` parameter can be useful when the modeled data does not have the correct amplitude, and represents the previously mentioned b scaling factor of the reflection response. The program can optionally, when the file-names `file_*` are defined, output results of computed Green's- and focusing-functions. Defining `file_iter` writes for each iteration the focusing update term $-N_i(-t)$ (=iRN(t) in Algorithm 1) before applying the mute window. By setting the `verbose` option to 2 the energy of the focusing update term is printed out for each iteration and can be used to monitor the convergence of the scheme. The code to reproduce all figures in this paper can be found in the directory `marchenko/demo/oneD`. The `README` file in that directory explains in detail how to run the scripts. A more complicated (laterally varying) model can be found in the directory `marchenko/demo/twoD`. This example usually takes several hours to complete the reflection data modeling on a personal computer, and is thus not discussed.

In addition to the Marchenko programs, the package also contains the previously published finite difference modeling code, that is used to model all data in the examples, in directory `fdelmodc` (Thorbecke and Draganov, 2011). The directory `utils` contains programs to calculate a gridded model (`makemod`), source wavelets (`makewave`) as well as programs for basic processing steps.
In the next subsections all the parameters will be described in more detail and guidelines will be given how to use them.

Verbose The parameter `verbose` prints messages and produces additional files during the running of the program. Table 1 shows the kind of messages and the extra files printed using different values for `verbose`. Those messages and files contain extra information for the user. The output files produced by different setting of the verbose parameter are:

setting	messages printed to stdout
0	no messages only warnings
1	data information, source, receiver, parameter info
2	+ iteration convergence
3	+ mute-window, OpenMP info
4	+ shot gather processing
>4	

Table 1: *The files and messages produced by different values of the* `verbose` *parameter.*

3.4 Examples to run the code

The demo directory contains scripts which demonstrate the different possibilities of the modeling program. In the subsections below most demo script are explained and results are shown.

4 Primaries removal: MME and T-MME algortihms

4.1 Introduction

In this section, we describe the implementation of both Marchenko Multiple Elimination (MME) and Transmission-compensated Marchenko Multiple Elimination (T-MME) schemes in detail. Both schemes eliminate internal multiple reflections without the need for model information or adaptive subtraction. Only a reflection response without source wavelet and free-surface related multiple reflections is required as input. The paper is organised as follows: In the theory section, we briefly review the equations of the MME and T-MME schemes. In the implementation section the processing details are explained step by step and this section provides a user's first step with the MME and T-MME schemes. The mechanism of the algorithm is illustrated with a simple three-reflector 1.5-dimensional horizontally layered model. This simple model is chosen to keep the number of events limited and to allow for an explanation that can be followed more easily. The method is not limited to simple models and can successfully be applied to complicated 3D media as well (see for example Zhang and Slob (2020c)).

Theory

In this section we give a brief overview of the theory of both MME and T-MME schemes. The acquisition surface is located at the surface boundary $\partial\mathbb{D}_0$. The reflection response $R(\mathbf{x}_0, \mathbf{x}_0', t)$ is measured with source and receiver positioned at $\mathbf{x}_0'$ and $\mathbf{x}_0$, which is free from free-surface related multiple reflections and source wavelet. The time is denoted t.

Marchenko multiple elimination

As presented by Zhang et al. (2019), we give the equations of the Marchenko Multiple Elimination (MME) scheme as

$$R_t(\mathbf{x}_0', \mathbf{x}_0'', t = t_2) = R(\mathbf{x}_0', \mathbf{x}_0'', t = t_2) + \sum_{m=1}^{\infty} M_{2m}(\mathbf{x}_0', \mathbf{x}_0'', t = t_2, t_2), \tag{37}$$

with

$$\begin{aligned} M_{2m}(\mathbf{x}_0', \mathbf{x}_0'', t, t_2) = &\int_{t'=0}^{+\infty} \int_{\partial\mathbb{D}_0} R(\mathbf{x}_0''', \mathbf{x}_0', t') H(t - t' - \varepsilon) \mathrm{d}\mathbf{x}_0''' \mathrm{d}t' \times \\ &\int_{t''=0}^{+\infty} \int_{\partial\mathbb{D}_0} R(\mathbf{x}_0, \mathbf{x}_0''', t'') H(t' - t + t_2 - t'' - \varepsilon) \times \\ &M_{2(m-1)}(\mathbf{x}_0, \mathbf{x}_0'', t - t' + t'', t_2) \mathrm{d}\mathbf{x}_0 \mathrm{d}t'', \end{aligned} \tag{38}$$

and initialization

$$M_0(\mathbf{x}_0', \mathbf{x}_0'', t, t_2) = -(H(t + t_2 - \varepsilon) - H(t + \varepsilon)) R(\mathbf{x}_0', \mathbf{x}_0'', -t), \tag{39}$$

where R_t denotes the retrieved dataset without internal multiple reflections at time t_2 and H indicates the Heaviside function, which is used to apply the offset independent truncation window $(\varepsilon, t_2 - \varepsilon)$ in the equations. The constant ε indicates a small positive value which can be taken as the half source time-duration in practice. The initialization of the scheme (with M_0) is the time-reversed shot record at shot position $\mathbf{x}_0''$ for times between $(\varepsilon, t_2 - \varepsilon)$. The left-hand side of equation 37 R_t is the same shot record, but without internal multiples. We follow Zhang and Staring (2018) and make time t_2 constant and independent of the source and receiver positions in the reflection response. Note that the integration is carried out over the receiver coordinate for both integrals, the same as implemented in the source code. The second term in the right-hand side of equation 37 predicts all internal multiple reflections correctly. Equation 39 indicates that the measured reflection response is the only input of the MME scheme given in equation 37. To retrieve a dataset without internal multiple reflections for all times t this process must be repeated for all times t_2.

Equation 38 contains the terms that correct for the internal multiples that are present in $R(\mathbf{x}_0', \mathbf{x}_0'', t)$. To better explain the right-hand side of equation 38 we divide the expression into two parts:

$$M_{2m}(\mathbf{x}_0', \mathbf{x}_0'', t, t_2) = \int_{t'=0}^{+\infty} \int_{\partial \mathbb{D}_0} R(\mathbf{x}_0''', \mathbf{x}_0', t') H(t - t' - \varepsilon) \times M_{2m-1}(\mathbf{x}_0''', \mathbf{x}_0'', t - t', t_2) \mathrm{d}\mathbf{x}_0''' \mathrm{d}t', \tag{40}$$

$$M_{2m-1}(\mathbf{x}_0''', \mathbf{x}_0'', t - t', t_2) = \int_{t''=0}^{+\infty} \int_{\partial \mathbb{D}_0} R(\mathbf{x}_0, \mathbf{x}_0''', t'') H(t' - t + t_2 - t'' - \varepsilon) \times M_{2(m-1)}(\mathbf{x}_0, \mathbf{x}_0'', t - t' + t'', t_2) \mathrm{d}\mathbf{x}_0 \mathrm{d}t''. \tag{41}$$

Equation 40 is a time domain *convolution* of R with M integrated over the spatial coordinate $\mathbf{x}_0'''$, which is the receiver position of the shot at $\mathbf{x}_0'$. Equation 41 is a time domain *correlation* of R with M integrated over the spatial coordinate $\mathbf{x}_0$, which is the receiver position of the shot at $\mathbf{x}_0'''$. The Heaviside function in equation 40 is to exclude negative times for $t - t' < \varepsilon$ and only use the causal time in M_{2m-1}. In equation 41 the function $M_{2(m-1)}$ contains non-zero values for times larger than $t_2 - \varepsilon$. These times should not contribute to the integral and the Heaviside guarantees that $M_{2(m-1)}$ does not have a contribution to the integration result for values of $t'' + t - t' > t_2 - \varepsilon$. Note that equation 40 and 41 perform very similar operations (they differ by a sign change) and are implemented by a single function. To evaluate equation 38 this function is applied two times. The convolution terms are the even numbered operations with this function and the correlation terms are the odd numbered operations.
To better explain the method and for illustration purposes the summation of the even M terms in equation 40 is defined as the field:

$$k_{1,i}^{-}(\mathbf{x}_0', \mathbf{x}_0'', t, t_2) = R(\mathbf{x}_0', \mathbf{x}_0'', t, t_2) - \sum_{m=1}^{i} \int_{t'=0}^{+\infty} \int_{\partial \mathbb{D}_0} R(\mathbf{x}_0''', \mathbf{x}_0', t') H(t - t' - \varepsilon) \times M_{2m-1}(\mathbf{x}_0''', \mathbf{x}_0'', t - t', t_2) \mathrm{d}\mathbf{x}_0''' \mathrm{d}t'. \tag{42}$$

We can evaluate equation 42 also for $t \geqslant t_2 - \varepsilon$ and the equation can be further split in the time domain as follows

$$k_{1,i}^{-}(\mathbf{x}_0', \mathbf{x}_0'', t, t_2) = \begin{cases} v_{1,i}^{-}(\mathbf{x}_0', \mathbf{x}_0'', t, t_2) & t < t_2 - \varepsilon \\ u_{1,i}^{-}(\mathbf{x}_0', \mathbf{x}_0'', t, t_2) & t \geqslant t_2 - \varepsilon \end{cases}, \tag{43}$$

where u_1^{-} and v_1^{-} are similar to the projected Green's function and focusing function in the regular Marchenko scheme as defined in Van der Neut and Wapenaar (2016). The minus superscript in equations 42 and 43 refers to upgoing wavefields at the receiver location $\mathbf{x}_0'$. To solve for M in equations 37-39, k_1^{-} is not needed. The time values in M between ε and $t_2 - \varepsilon$ are used to solve the Marchenko equations and compute a value at t_2 in u_1^{-}. The solution for time t_2 in the MME scheme is collected from u_1^{-} at $t = t_2$. Similarly for the summation of the terms in equation 41, a downgoing function is defined as

$$k_{1,i}^{+}(\mathbf{x}_0', \mathbf{x}_0'', t, t_2) = \int_{t'=0}^{+\infty} \int_{\partial \mathbb{D}_0} R(\mathbf{x}_0', \mathbf{x}_0, -t') v_{1,i}^{-}(\mathbf{x}_0, \mathbf{x}_0'', t - t', t_2) \mathrm{d}t' \mathrm{d}\mathbf{x}_0. \tag{44}$$

$$v_{1,i}^{+}(\mathbf{x}_0''', \mathbf{x}_0'', t, t_2) = \sum_{m=1}^{i} \int_{t''=0}^{+\infty} \int_{\partial \mathbb{D}_0} R(\mathbf{x}_0, \mathbf{x}_0''', t'') H(t' - t + t_2 - t'' - \varepsilon) \times M_{2(m-1)}(\mathbf{x}_0, \mathbf{x}_0'', t + t'', t_2) \mathrm{d}\mathbf{x}_0 \mathrm{d}t'', \tag{45}$$

Equation (44) can be further split in the time domain as follows;

$$k_{1,i}^{+}(\mathbf{x}_0''', \mathbf{x}_0'', t, t_2) = \begin{cases} v_{1,i}^{+}(\mathbf{x}_0''', \mathbf{x}_0'', t, t_2) & t < t_2 - \varepsilon \\ u_{1,i}^{+}(\mathbf{x}_0''', \mathbf{x}_0'', t, t_2) & t \geqslant t_2 - \varepsilon \end{cases}, \tag{46}$$

where v_1^{+} is similar to the projected focusing function in the regular Marchenko scheme as defined in Van der Neut and Wapenaar (2016). In v_1^{+} the multiple annihilator is created and this is demonstrated in the numerical examples section. Equation 44 only holds for $\varepsilon < t < t_2 - \varepsilon$. The plus superscript in equation 44 refers to downgoing wavefields. To solve the MME Marchenko equations v_1^{+} is not needed and only defined for illustration purposes to explain the mechanism of the method.

Time t_2 is the instant two-way travel-time where the solution of the Marchenko equation is computed. The primary reflection is collected from u_1^- for every time instant t_2. This is a computational expensive way, since only one sample is collected in the output. Nevertheless, this process is fully automated and implemented without any human interaction or model information. It is possible to collect more that one sample around the instant time t_2, and take bigger time steps, but the number of samples to use around t_2 must take into consideration the frequency bandwidth of the data. This statement is supported by examples in the detailed discussion of the implementation and allows the implementation of a faster algorithm.
In this MME scheme the primary is collected from the original reflection data. The Marchenko scheme removes all overlapping internal multiples from earlier reflections, the primary is untouched and keeps the physical reflection amplitude as present in the data.

Transmission compensated Marchenko multiple elimination

Both internal multiple reflections and transmission losses in primary reflections can be accounted for by the Transmission-compensated Marchenko Multiple Elimination (T-MME) scheme (Zhang et al., 2019). The equation is given by

$$R_r(\mathbf{x}_0', \mathbf{x}_0'', t = t_2) = R(\mathbf{x}_0', \mathbf{x}_0'', t = t_2) + \sum_{m=1}^{\infty} \bar{M}_{2m}(\mathbf{x}_0', \mathbf{x}_0'', t = t_2, t_2), \tag{47}$$

with

$$\begin{aligned}\bar{M}_{2m}(\mathbf{x}_0', \mathbf{x}_0'', t, t_2) = &\int_{t'=0}^{+\infty} \int_{\partial\mathbb{D}_0} R(\mathbf{x}_0''', \mathbf{x}_0', t') H(t - t' + \varepsilon) \mathrm{d}\mathbf{x}_0''' \mathrm{d}t' \times \\ &\int_{t''=0}^{+\infty} \int_{\partial\mathbb{D}_0} R(\mathbf{x}_0, \mathbf{x}_0''', t'') H(t' - t + t_2 - t'' + \varepsilon) \\ &\bar{M}_{2(m-1)}(\mathbf{x}_0, \mathbf{x}_0'', t - t' + t'', t_2) \mathrm{d}\mathbf{x}_0 \mathrm{d}t''\end{aligned} \tag{48}$$

and

$$\bar{M}_0(\mathbf{x}_0', \mathbf{x}_0'', t, t_2) = -(H(t + t_2 + \varepsilon) - H(t + \varepsilon)) R(\mathbf{x}_0', \mathbf{x}_0'', -t), \tag{49}$$

where R_r denotes the retrieved dataset without internal multiple reflections and transmission losses in primary reflections. The truncation window in equation 48 is different from the window in equation 38, which guarantees that the second term in the right-hand side of equation 47 predicts both internal multiple reflections and transmission losses in primary reflections. In equation 48 the Heaviside guarantees that $\bar{M}_{2(m-1)}$ does not have a contribution for values of $t'' + t - t' > t_2 + \varepsilon$. In contrast to equation 41, time t_2 is now part of the integration and included in the sum of $\bar{M}_{2m}$. Still, as given in equation 49, the measured reflection response is the only input to solve the T-MME scheme given in equation 47.
The primary reflection is, different than in the MME scheme, collected from $\bar{v}_1^-$, which achieves the transmission compensation. There is no need to define a $\bar{k}_1^-$, since $\bar{v}_1^-$ is already part of $\bar{M}_{2m}$. The scheme is applied for every time instant t_2 and has the same advantages and disadvantages as the MME scheme. In the T-MME scheme the amplitude of the primary is automatically transmission compensated, because it is the only way to predict and attenuate internal multiples from earlier primary reflections. We come back to this remark in the explanation of Figure 13.
Both MME and T-MME schemes require **only** the measured reflection response R as input. The reflection response R needs to be deconvolved for the source-wavelet and the free-surface related multiple reflections must be removed. The output of a surface-related multiple elimination (SRME Verschuur et al. (1992)) scheme can meet these requirements. Diffracted and refracted waves are beyond the capability of both schemes and a detailed analysis about these limitations can be found in Zhang et al. (2019).

4.2 Marchenko Algorithm

The basic Marchenko algorithm (MME) is explained in Algorithm 2. The arrays in this algorithm are stored in C-order; the last (most right) addressed dimension is contiguous in memory. The discrete dimensions of these arrays are within square brackets $[\dots]$, the arguments of function calls are within regular brackets $(\dots)$. The *only* data input of the algorithm is the measured refection data R. This

Main **begin**
  Read SU-style input parameters
  Initialization, reading of input parameters and allocate arrays
  READ($R[N_{shots}, i\omega, N_{recv}]$)
  $DD[N_{recv}, it] = \mathcal{F}^{-1}\{R^*[j, i\omega, N_{recv}]\}$
  for $ii \leftarrow istart$ **to** $iend$ **do**
    $M_0[N_{recv}, it] = \begin{cases} 0 & 0 < it < n_t - ii + n_\varepsilon \\ -DD[N_{recv}, it] & n_t - ii + n_\varepsilon \leqslant it < n_t \end{cases}$
    $k^-_{1,0}[N_{shots}, it] = DD[N_{recv}, n_t - it]$
    $v^+_{1,i}[N_{shots}, it] = 0$
    for $i \leftarrow 0$ **to** n_i **do**
      synthesis(R, M_i, RM_i)
      $M_{i+1}[N_{shots}, it] = RM_i[N_{shots}, n_t - it]$
      if *(i % 2 == 0)* **then**
        $M_{i+1}[N_{shots}, it] = 0; \quad ii - n_\varepsilon < it < n_t$
        $v^+_{1,i+1}[N_{shots}, it] = v^+_{1,i}[N_{shots}, it] + M_{i+1}[N_{shots}, it]$
      else
        $k^-_{1,i+1}[N_{shots}, it] = k^-_{1,i}[N_{shots}, it] - M_{i+1}[N_{shots}, n_t - it]$
        $M_{i+1}[N_{shots}, it] = 0; \quad 0 < it < n_t - ii + n_\varepsilon$
      end
    end
    $R_t[j, N_{shots}, ii] = k^-_{1,n_i}[N_{shots}, ii]$
  end
end

Algorithm 2: Basic Marchenko algorithm, without transmission loss compensation, as implemented in the provided source code. Integer time sample number it, that runs from $istart$ to $iend$, represents time $t = it * \Delta t$. The number of recorded time samples is n_t, the time duration of the source signature $\varepsilon = n_\varepsilon * \Delta t$ and time sample ii represents instant time $t_2 = ii * \Delta t$. The number of receivers in R is N_{recv} and number of shots N_{shots}. The i loop represents the number of Marchenko iterations n_i. Note that the sample expression $n_t - it$ stands for negative time $-t$.

reflection data must be properly pre-processed as explained in Brackenhoff et al. (2019). The pre-processing must take care of the following:

- Elimination of free-surface multiples.
  Note that there is also a very similar Marchenko algorithm that takes into account free-surface multiples as well. Ravasi (2017) discusses a redatuming algorithm similar to Singh et al. (2015) and requires a smooth model of the medium, while Zhang and Slob (2019) remove all multiples and does not need any model information.
- Sufficient (i.e. alias free) sampling in the spatial receiver and shot direction.
  Note, there are Marchenko-based methods that can fill in missing shot points or receiver locations, under the assumption that the available data are unaliased (Wapenaar and van IJsseldijk, 2020).
- Compensate for dissipation.
- Shot amplitude regularization.
- Deconvolution for source wavelet.

Following Algorithm 2, the pre-processed reflection data is read from disk, transformed (by FFT operator $\mathcal{F}\{\dots\}$) to the frequency (ω) domain and all shots and receivers are stored into memory. This is the first step in the algorithm and the only significant data read. One single shot record (with shot-number j), where we want to suppress the internal multiples from, is selected from this reflection data in the next step. This shot record is transformed back to time, time reversed (R^*), and stored in array DD. The

first loop in the algorithm loops over the selected number of time samples that are processed to attenuate internal multiples. Typically this represents all samples in the shot record, with a possible exclusion of the number of samples to the first reflection event in the selected shot record. For each time sample ii the iterative Marchenko algorithm is executed. The largest difference from the algorithm described in Thorbecke et al. (2017) is that time-truncation along the first arrival-time (from a focal point in the subsurface) is replaced by a constant time-truncation and the computation of a first arrival-time is not needed anymore. The initialization of the algorithm by M_0 is from the same shot record j from which we would like to attenuate the internal multiples (DD). M_0 is a copy of the time reversed shot record, see equation 39, and set to zero from the first sample 0 to sample $n_t - ii + n_\varepsilon$, where n_t is the total number of samples in the shot record. The extra samples of n_ε take into account the time duration of the wavelet to exclude a possible reflection event at time ii. The initialization of $k^-_{1,0}$ is a complete (no time-muting is carried out) copy of the shot record that still contains all internal multiples we would like to remove. With these two initializations the iterations of the Marchenko algorithm can start. In each iteration an updated field is computed by the integration of M_i with R. This integration process is called `synthesis`, produces the output RM_i, and is explained in more detail below. Depending on the iteration number i, being odd or even, different time muting windows are in use to mute events in RM_i and to compute an updated M_{i+1}. For even iterations the times between $ii - n_\varepsilon$ and n_t are set to zero and for odd iterations the times between 0 and $ii + n_\varepsilon$ are set to zero. Only in the odd iterations $k^-_{1,i}$ is updated with the unmuted M_{i+1}. In this update of $k^-_{1,i}$ internal multiples around time ii are attenuated. This is the update represented in equation 37, where the update M_{2m} is in fact one even and one odd iteration in the implemented Marchenko algorithm, and hence the notation with $_{2m-1}$ in equation 42.

In the regular redatuming Marchenko algorithm (Thorbecke et al., 2017) the truncation windows follow the first arrival time of a focal point in the subsurface. In the Marchenko multiple elimination algorithm, the focal point is projected on the surface and the time-truncation is conveniently chosen at a constant time. The flat time window has the big advantage that it requires no additional model- or data-information. Meles et al. (2020) demonstrate that in the application of the multiple elimination algorithm to dipping plane waves a time truncation consistent with the dip-angle must be used.

Depending on the position of strong reflectors typically, 10-50 Marchenko iterations are needed for each time sample ii in the selected shot record. The presence of strong reflectors in the shallow part makes the convergence slow at large time instances, see also Figure 20. The reason is that higher-order multiples are attenuated with events that are created in v^-_1 and that are removed again later when the first-order multiple is finally removed by a converged multiple attenuator. When the first-order multiple is removed, all multiples are removed and hence all earlier higher-order multiple attenuator artifacts will also vanish. Once the iterations are finished the output of sample ii of the updated Marchenko result u^-_1 is stored in sample ii in the multiple free shot record R_t, the final output of the program that represents the selected shot record with attenuated internal multiples. It is a compute intensive task to solve the Marchenko equations for each sample ii in the shot record. Algorithm 3 is a faster (10-20x) implementation of Algorithm 2.

In Algorithm 3, after the Marchenko equations are solved at time sample $ii - 1$, the next time sample ii is initialized with the result of time sample $ii - 1$ (Zhang and Slob, 2020b). The idea is that to remove the internal multiples at the next time sample there is no need to start from scratch and remove the multiples that were already removed in the previous time-sample. For every next time-sample all earlier attenuated multiples need to be attenuated plus one (or a few) more. In the Marchenko update for the new time sample only the multiples have to be removed that were not removed before. This is a small deviation from the previous results and usually 2 iterations are sufficient to accomplish the update for the next time sample. The initial M_0 in the fast algorithm is the difference between the original data (DD) and $k^{-,(ii-1)}_{1,n_i}$; the already estimated internal multiples from time sample $ii - 1$. The initial $k^-_{1,0}$ is the previous result $k^{-,(ii-1)}_{1,n_i}$. With these initializations the update term RM_i contains only a small correction, since it is based on a converged previous result that is very close to the actual solution. To get the complete internal multiple in M_{i+1}, DD is added to RM_i (Zhang and Slob, 2020b).

In this fast algorithm only one pair of even-odd iterations is needed to reach convergence. In principle we could solve the equations only one time and use that result to update all other time samples. On simple models of numerically modeled data this works fine indeed. However, on geologically complicated models of numerically modeled data and on field data we have to do a full update every 10 to 20 recursive updates and the speed-up of the faster algorithm is limited to one order of magnitude. On complex data-sets we would advise to begin with the basic algorithm and then verify if the fast algorithm can be used to

speed-up the computations. The reason for this limited use of the fast algorithm is that for complex data-sets and a large number of iterations, artifacts, for example introduced by a limited aperture, can get amplified. The primary reflections will still converge, but numerical artifacts are not accounted for in the algorithm and can diverge. In the iterative scheme each update adds two iterations to the already computed result based on for example 30 iterations. With 10 iterative updates, 20 iterations are added and can cause artifacts being amplified to signal level.
In the algorithm we solve the Marchenko equations for each sample ii. From the theory we know that the first event after sample ii is a primary reflector (all multiple reflections generated by the reflectors before sample ii are removed by the scheme). Hence, a number of samples after sample ii will still be free of internal multiples. We could make larger steps with ii and use the Marchenko result for a number of samples (at least n_ε samples, since that is the time resolution we are already working with) after sample ii. This can speed-up the code by n_ε (typically 20) times. This is similar to the fast algorithm, but without making any iterations and directly use the previous computed result.

Main **begin**
  Read SU-style input parameters
  Initialization, reading of input parameters and allocate arrays
  READ($R[N_{shots}, i\omega, N_{recv}]$)
  $DD[N_{recv}, it] = \mathcal{F}^{-1}\{R^*[j, i\omega, N_{recv}]\}$
  for $ii \leftarrow istart$ **to** $iend$ **do**
    $k^-_{1,0}[N_{shots}, it] = k^{-,(ii-1)}_{1,n_i}[N_{shots}, it]$
    $M_0[N_{shots}, it] = \begin{cases} 0 & 0 < it < n_t - ii + n_\varepsilon \\ DD[N_{shots}, n_t - it] - k^-_{1,0}[N_{shots}, n_t - it] & n_t - ii + n_\varepsilon \leqslant it < n_t \end{cases}$
    for $i \leftarrow 0$ **to** n_i **do**
      synthesis(R, M_i, RM_i)
      $M_{i+1}[N_{shots}, it] = RM_i[N_{shots}, n_t - it]$
      if *(i % 2 == 0)* **then**
        $M_{i+1}[N_{shots}, it] = 0; \quad ii - n_\varepsilon < it < n_t$
      else
        $M_{i+1}[N_{shots}, it] = M_{i+1}[N_{shots}, it] - DD[N_{recv}, it]$
        $k^-_{1,i+1}[N_{shots}, it] = -M_{i+1}[N_{shots}, n_t - it]$
        $M_{i+1}[N_{shots}, it] = 0; \quad 0 < it < n_t - ii + n_\varepsilon$
      end
    end
    $R_t[j, N_{shots}, ii] = k^-_{1,n_i}[N_{shots}, ii]$
  end
end

Algorithm 3: Faster Marchenko algorithm that uses previous results from time instant $ii - 1$ ($k^{-,(ii-1)}_{1,n_i}$) as input for the current time instant ii.

The synthesis process shown in Algorithm 4 computes the second integrant in the right-hand side of equation 38. The synthesis function is a straightforward matrix-vector multiplication. The reflection data are stored in such a way that the most inner loop, that sums over the receiver positions within a shot, is contiguous in memory. To speed-up the computation a parallel OpenMP region is carried out over the outer N_{shots} loop. An alternative implementation of the synthesis process is to make the frequency loop the outer loop and use a BLAS `dgemv` function to compute the matrix-vector multiplication. This implementation will also be efficient when all shots are computed at the same time and the BLAS matrix-matrix `dgemm` function becomes the kernel of the synthesis process. Note that in the synthesis process the integration is carried out over the number of receivers per shot and each integration result is stored at the shot position. Thus after the synthesis process N_{shots} output traces are computed.
From a computational point of view the transmission compensated algorithm (T-MME) is the same as the MME algorithm, except for the application of the time-truncation window. The sample length of the wavelet (n_ε) is applied in the opposite time direction for the T-MME algorithm. The extra samples

synthesis($R[N_{shots}, i\omega, N_{recv}), M[N_{shots}, it], RM[N_{shots}, it]$)
begin
  $Fop[i\omega, N_{shots}] = \mathcal{F}\{M[N_{shots}, it]\}$
  $RM[N_{shots}, t] = 0$
  #pragma omp parallel for
  for $k \leftarrow 0$ **to** N_{shots} **do**
    for $i\omega \leftarrow \omega_{min}$ **to** ω_{max} **do**
      for $i \leftarrow 0$ **to** N_{recv} **do**
        $sum[i\omega] = sum[i\omega] + R[k, i\omega, i] * Fop[i\omega, i]$
      end
    end
    $RM[k, it] = \mathcal{F}^{-1}\{sum[i\omega]\}$
  end
end

Algorithm 4: Marchenko synthesis kernel with $i\omega = i * \Delta\omega (= \frac{2\pi}{n_t * \Delta t})$.

of n_ε in the MME algorithm take into account the length of the wavelet to **exclude** a possible event at instant time ii in the initialization and update of M_i. Suppose that time ii is the two-way traveltime of a reflector (see Figure 13a). The reflection of the reflector is excluded in M_i in the MME algorithm (Figure 13a), but included in $\bar{M}_i$ in the T-MME algorithm (Figure 13b). In the T-MME algorithm the reflected event at instant time ii ends up in the updated $\bar{v}_1^-$, while in the MME algorithm the reflected event from the original shot record ends up in u_1^-. When the instant time ii is chosen between two reflectors then there is no difference between the updates made in the MME (Figure 13c) or T-MME (Figure 13d) scheme.
To get to the T-MME scheme from Algorithm 2 one has to replace in the defined time windows $+n_\varepsilon$ with $-n_\varepsilon$. Then R_t becomes R_r that contains the transmission compensated primary reflections.

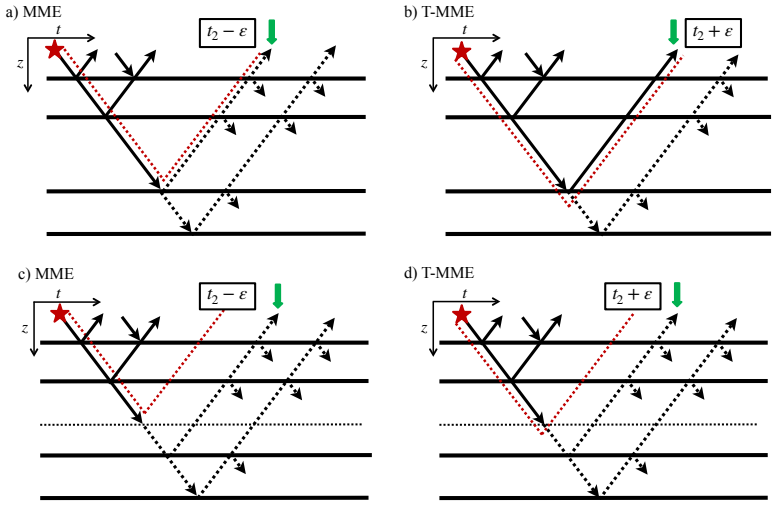


Figure 13: Comparison of the MME and and T-MME schemes. Figures a) and b) show a selected time t_2 equal to the two-way traveltime of the third reflector. The time-truncation window is indicated with a red dotted line. The dotted lines are events that are excluded in M_i, the solid lines are events included in M_i, after application of the time window. Figures c) and d) show a time between two reflectors.

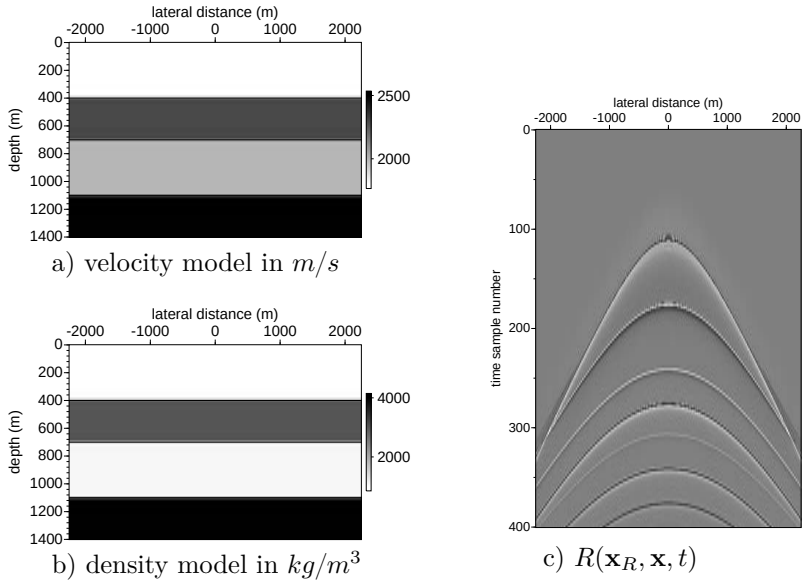


Figure 14: Four layer model with velocity (a) and density (b) parameters. A shot record, with source position $\mathbf{x} = (x = 0, z = 0)$ and receivers at $\mathbf{x}_R = (x = x_r, z = 0)$ (c). The source wavelet in R (c) has a flat frequency spectrum from 5 to 90 Hz.

4.3 Numerical examples

The Marchenko algorithm is illustrated with a 1.5-dimensional horizontally layered model shown in Figure 14. The numerical modeling is carried out with a finite-difference modeling program (Thorbecke and Draganov, 2011) that is also included in the software package. The input source signature, to model the reflection response $R(\mathbf{x}_0', \mathbf{x}_0'', t)$, is approximately a sinc-function with a flat spectrum of amplitude 1 between f_{min} and f_{max} $(5 - 90$ Hz) to represent a deconvolved source wavelet. In the finite-difference program for modeling $R(\mathbf{x}_0', \mathbf{x}_0'', t)$ a source of vertical force is chosen. The receivers are placed at $z = 0$ and measure the pressure field. A fixed spread acquisition is chosen between -2250 to 2250m and the distance between the 901 source/receiver positions is 5m. The receiver traces have a time sampling interval of 4ms and 1024 recorded time samples.

4.3.1 The first iterations

Figure 15 demonstrates the first iteration of equation 41 with $m = 1$ to compute M_1 for time sample-number 276 (t=1.100 s) from M_0. Time-sample 276 corresponds to the zero-offset arrival time of the third reflector. In this first step all shots in the reflection data R are correlated with a time-windowed shot record. In our example we use the middle shot record; $R_0(\mathbf{x}_R, \mathbf{x} = (0, 0), t)$ (shot-number $j = 451$). Before the correlation is carried out the selected shot record is first set to zero beyond time sample $276 - n_\varepsilon$, multiplied by -1 and time reversed, at which moment we have $M_0(t)$ in equation 39. In Figure 15b the shot record is convolved with a Ricker wavelet to reduce the ringing of the flat spectrum of the (deconvolved) wavelet present in R (Figure 15a). The number of n_ε (in this example $n_\varepsilon = 20$) samples excludes the reflection from the third reflector in M_0. In Figure 15 the middle shot record of R (Figure 15a, where we used source receiver reciprocity) is correlated with the time windowed $M_0(-t)$ (Figure 15b) to give the result in Figure 15c. The events in Figure 15b include the first and second reflection and the first internal multiple between the first and second reflector. In the correlation result (Figure 15c) we see the auto-correlation of the three reflection events around $t = 0$ (with events at negative times appearing at the bottom of the panel). Note that the long train of events starting at the positive time-axis in Figure 15c can interfere with events at the end of the time axis. To overcome this time interference we usually pad the time axis with zeros before the transformation to the frequency domain where the correlation is computed.
The correlation result is time-reversed and shown in Figure 15d for the first 400 samples. There are only three events in Figure 15d and these originate from correlation with the three events in Figure 15b with

the first three events $(\mathbf{r}_1, \mathbf{r}_2, \mathbf{m}_1)$ in the shot record. According to the integral in equation 41, to obtain an output trace of M_1 the traces in Figure 15d are summed together. The stationary points of the events in Figure 15d give a contribution in the result of the summation. Besides the stationary points, truncated events (both in time and space) give unwanted contributions that show up as artifacts in the final result. The integration result is set to zero for samples larger than $276 - n_\varepsilon$ and ends up as a trace at position 0 (the middle trace) of M_1 shown in Figure 15e. In Figure 15e the truncation appears to be at sample 160, but that is the truncation in 15b shifted upward in time with the arrival time of the first reflector. The truncation at sample $276 - n_\varepsilon$ is indicated with a dotted line. There are two hyperbolic events visible and a few linear artifacts. The first hyperbolic event originates from correlation of events $\mathbf{r}_2^*.\mathbf{r}_1$ and events $\mathbf{m}_1^*.\mathbf{r}_2$, and the second hyperbolic event from the first internal multiple and first reflector $\mathbf{c}_2 = \mathbf{m}_1^*.\mathbf{r}_1$. The linear events are unwanted artifacts due to truncation and can be suppressed by applying a smooth taper at the truncation boundaries in time and space.

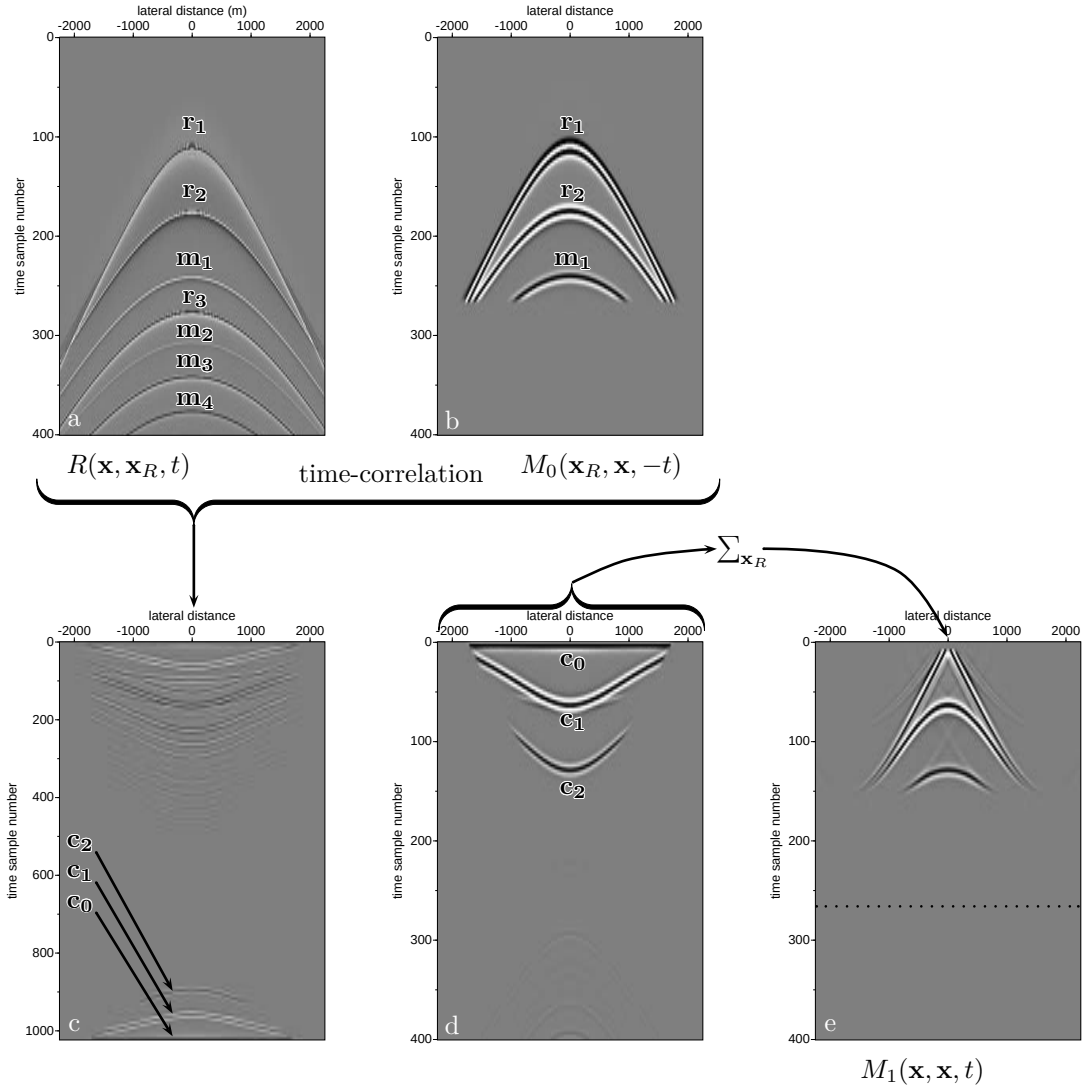


Figure 15: Computational steps to compute M_1 from M_0 at time sample number 276. The middle shot record from R is shown in (a); time truncated after sample $276 - n_\varepsilon$, and convolved with a Ricker wavelet, it gives M_0 in (b). Time-correlation of (a) with (b) gives (c). After time-reversal of (c) and applying the time window again gives (d). The traces in (d) are summed together and only the stationary point of events above sample $276 - n_\varepsilon$ will end-up in the middle trace of M_1 (e). The mute window n_ε samples later than $t = 0$ is needed to mute the autocorrelation of the first-event. The labeled events $\mathbf{r}_i$ indicate the i'th reflector and $\mathbf{m}_l$ the l'th multiple. In (c) the labeled correlated events are $\mathbf{c_0} = \mathbf{r_1^*}.\mathbf{r_1} + \mathbf{r_2^*}.\mathbf{r_2} + \mathbf{m_1^*}.\mathbf{m_1}$, $\mathbf{c_1} = \mathbf{r_2^*}.\mathbf{r_1} + \mathbf{m_1^*}.\mathbf{r_2}$, and $\mathbf{c_2} = \mathbf{m_1^*}.\mathbf{r_1}$.

Figure 16 demonstrates the computation of the second iteration to compute M_2 from M_1 (Figure 15e) according to equation 40 with $m = 1$. The reflection data (Figure 16a) are convolved with M_1 (Figure 16b) that contains three main events; a linear-artifact ($\mathbf{a}_1$) and two correlation results ($\mathbf{c}_1$ and $\mathbf{c}_2$). Convolving M_1 with the middle shot record of the data R gives Figure 16c. The hyperbolic events in M are now back at the same times as reflection events in the shot-record. The linear artifacts in M_1 also convolves with all events in R and introduce many (mostly linear) artifacts. The convolution result is reversed in time (Figure 16d) and after the integration in equation 40 over the lateral coordinate $\mathbf{x}_R$ it becomes the middle trace in M_2 (Figure 16e). Most of the linear artifacts are reduced in amplitude due to the destructive interference in the integration, only the '$\mathbf{a_2} = \mathbf{r_1}.\mathbf{a_1}$' artifact is still present in M_2. The first term in the sum in equation 37 is now computed; M_2. The last events in the time reverse of M_2, presented in Figure 16e, will already attenuate the multiple event $\mathbf{m}_1$ in the shot-record.

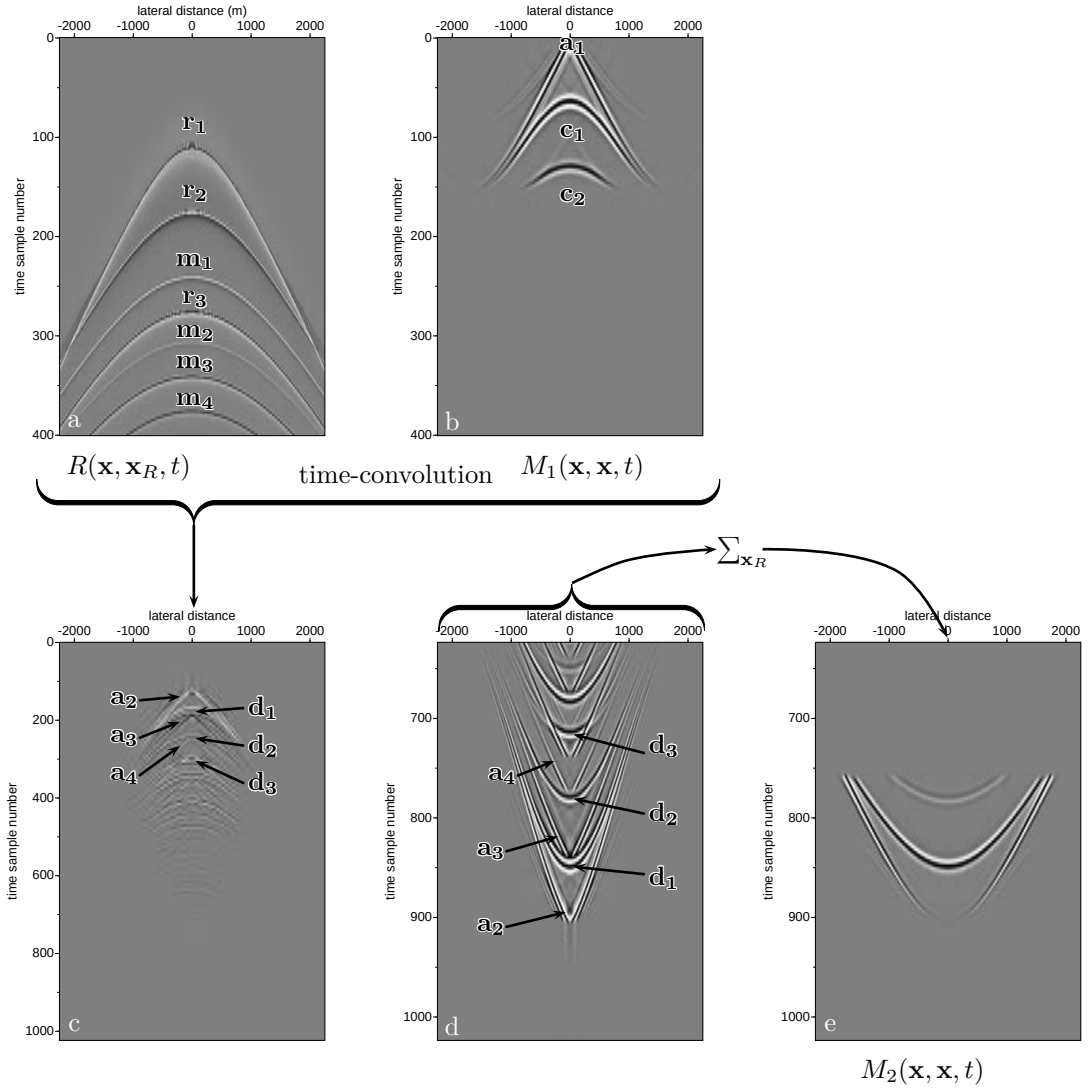
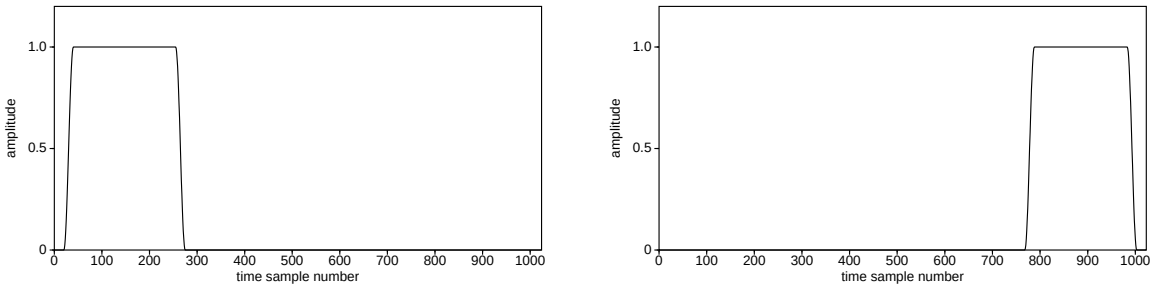


Figure 16: Computational steps to compute $M_2(\mathbf{x}, \mathbf{x}, t)$ from M_1 at time sample number 276. The middle shot record from R is shown in (a) and M_1 truncated after sample 276 (and computed in Figure (15e)) in (b). Time-convolution of (a) with (b) gives (c). After time-reversal of (c) and applying the time window again gives (d). The traces in (d) are summed together and only stationary events later than sample $n_t - 276 + n_\varepsilon$ end-up in the middle trace of M_2 (e). The labeled events $\mathbf{r}_i$ indicate the i'th reflector, $\mathbf{m}_l$ the l'th multiple and $\mathbf{a}_n$ the n'th artifact. The labeled events from the convolution between R(a) and M_1(b) are $\mathbf{a_2} = \mathbf{r_1}.\mathbf{a_1}, \mathbf{a_3} = \mathbf{r_2}.\mathbf{a_1}, \mathbf{a_4} = \mathbf{m_1}.\mathbf{a_1}, \mathbf{d_1} = \mathbf{r_1}.\mathbf{c_1}, \mathbf{d_2} = \mathbf{r_1}.\mathbf{c_2} + \mathbf{r_2}.\mathbf{c_1}$ and $\mathbf{d_3} = \mathbf{r_2}.\mathbf{c_2} + \mathbf{m_1}.\mathbf{c_1}$.

To compute M_1 (in general odd numbered updates to M_i), events are shifted backward in time (corre-

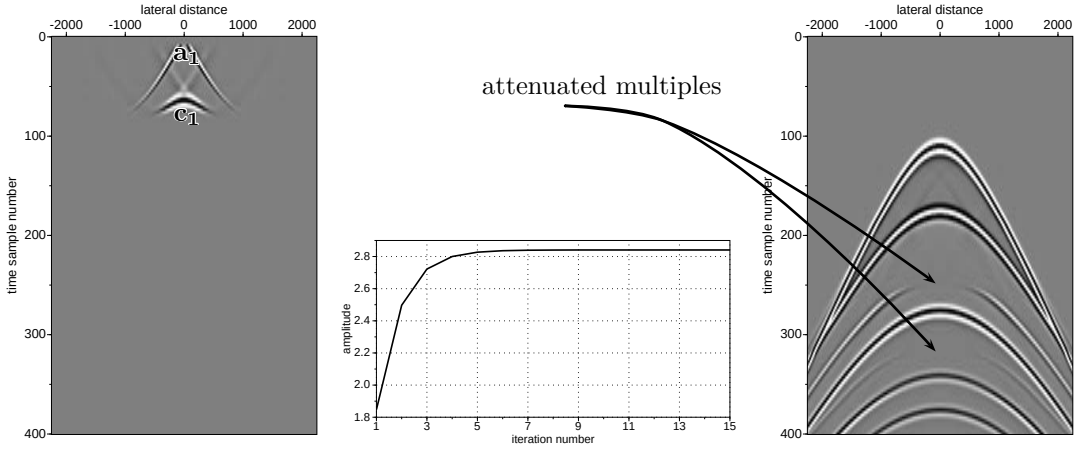
lation) with the times of the events in M_0. To compute M_2 (even numbered updates to M_i) from M_1, events are shifted forward (convolution) in time. The even and odd iterations are treated differently in the scheme. Each even iteration updates M_i and v_1^+, and each odd iteration updates M_i and k_1^-. The scheme reverts the time-axis for each iteration, hence the time windows, that set time samples to zero, switches also. These time windows, for sample 276, are shown in Figure 17. In these time windows a smooth cosine shaped transition zone is used to reduce the time-truncation artifacts.



a) Time-window for even iteration numbers b) Time-window for odd iteration numbers

Figure 17: Time-window functions in the Marchenko scheme with a smooth cosine-shaped transition zone. This transition zone has a default setting of $0.5n_\varepsilon$ samples and is within the n_ε samples.

4.3.2 Multiple removal in action



a) $v_{1,1}^+(t)$ b) Convergence of $\mathbf{c}_1$ c) $k_{1,30}^-(t)$

Figure 18: Creation of the event (labeled $\mathbf{c}_1$) that annihilates all the internal multiples between the first and second reflector, the artifact $\mathbf{a}_1$ is ignored in the analysis. Picture a) shows $v_{1,1}^+(t)$ for the first Marchenko iteration at sample number $ii = 200$. In b) the convergence of the maximum amplitude in $\mathbf{c}_1$ is shown as function of the iteration count. Figure c) shows the annihilated multiples in $k_{1,30}^-(t)$ after 30 iterations.

The results in Figure 18 are partial solutions of the Marchenko equations computed for time sample $ii = 200$. After applying the time window, that sets all samples in M_0 to zero beyond $200 - n_\varepsilon$, there are no internal multiple reflections present anymore in M_0. The times between 0 and sample 200 include $\mathbf{r}_1$ and $\mathbf{r}_2$, but not $\mathbf{m}_1$, see Figure 15b. In the first iteration to compute v_1^+, according to equation 44, one extra event in v_1^+ (Figure 18a event $\mathbf{c}_1$) is created to correct for the amplitude of the second reflector in v_1^-. Note that the time windows in Figure 18a appears to be around sample 80, but this is the time window applied in M_0 shifted by the correlation to negative times and time-reversed. The amplitude of this event $\mathbf{c}_1$ converges to the amplitude that can annihilate the amplitude of the first multiple. Applying the converged v_1^+ on the reflection data through equation 40, causes that all multiples arising from bounces

between the first and second reflector will vanish from the data in equation 37. The scheme finishes

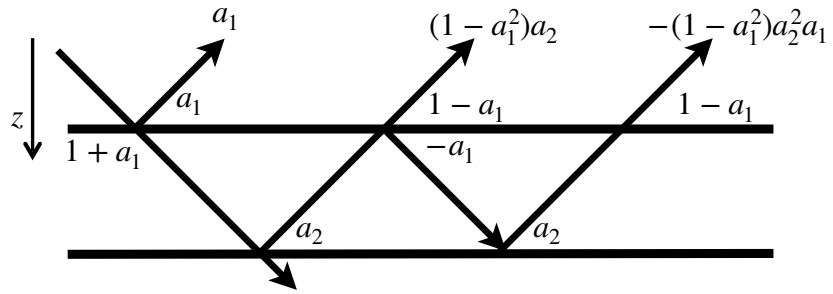


Figure 19: Sketch of the ray-paths and reflection and transmission coefficients in a three layer constant velocity and variable density model. The local reflection coefficients of the first and second reflector are a_1 and a_2 respectively.

For the investigation of the amplitudes of the event in v_1^+ we assume, for the sake of argument, that the reflection coefficient is a constant. The primary reflections in Figure 15b have local reflection coefficient a_1, a_2 for respectively the events labeled $\mathbf{r}_1, \mathbf{r}_2$. We consider the situation in Figure 18, for time sample 200 that only creates one extra event in v_1^+. We demonstrate that after sufficient iterations the event in v_1^+ has converged to an amplitude that can cancel the first-order multiple, and hence all higher-order multiples related to that event. For sample 200, M_0 contains only the primary reflections $\mathbf{r}_1, \mathbf{r}_2$ that have amplitude:

$$r_1^a = a_1, \tag{50}$$
$$r_2^a = (1-a_1^2)a_2. \tag{51}$$

Figure 19 is a sketch of the reflection paths and reflection and transmission coefficients for this two reflector case. According to equation 41, for the first iteration M_0 is correlated with R and integrated over the receiver coordinate. After applying the time window on M_1 only one event remains; event $\mathbf{c}_1 = \mathbf{r}_2^*.\mathbf{r}_1$ in Figure 18a with amplitude:

$$c_{1,1}^a = a_1(1-a_1^2)a_2. \tag{52}$$

The second subscript in $c_{1,1}^a$ indicates the iteration number. This event is convolved with R in the next iteration (according to equation 40) and after time windowing only one event $\mathbf{r}_1.\mathbf{c}_1$ remains at the time of the reflection of the second reflector with amplitude

$$c_{1,2}^a = a_1^2(1-a_1^2)a_2. \tag{53}$$

In each next iteration, alternating between equations 41 and 40, another multiplication with a_1 is added, in general for iteration i we have:

$$c_{1,i}^a = (a_1)^i(1-a_1^2)a_2. \tag{54}$$

Summation of all odd $c_{1,i}^a$ iterations i gives the final amplitude of the multiple annihilation event $\mathbf{c}_1$ in v_1^+. The initialization of v_1^+ is zero and the summation of the odd terms lead to:

$$\begin{aligned}\sum_{i=0}^{n_i} c_{1,1+2i}^a &= \sum_{i=0}^{n_i}(a_1)^{1+2*i}(1-a_1^2)a_2, \\ &= a_1a_2 - a_1^3a_2 + a_1^3a_2 - a_1^5a_2 + a_1^5a_2 - a_1^7a_2 + \dots \\ &\approx a_1a_2.\end{aligned} \tag{55}$$

Application of v_1^+ to the data creates multiple free data in the resulting u_1^- and is shown in Figure 18c. The first-order internal multiple from the data and the multiple annihilation event $\mathbf{c}_1$ in Figure 18a, with amplitude $a_1 a_2$, will meet each other in time just below the first reflector. At that point in time the annihilator cancels the first-order downgoing internal multiple and with that all other related multiples. To be able to cancel the first downgoing internal multiple the annihilator must have the same amplitude as that event. The first-order multiple event $\mathbf{m}_1$ in Figure 15b has amplitude

$$m_1^a = -(1 - a_1^2) a_2^2 a_1. \tag{56}$$

After convergence of the scheme the multiple annihilator event $\mathbf{c}_1$ is convolved with the second reflector $\mathbf{r}_2$ of R in the next iteration and arrives at the same time as $\mathbf{m}_1$ and has the same amplitude as m_1^a:

$$\begin{aligned} c_1^a r_2^a = &= a_1 a_2 . r_2^a, \\ &= (1 - a_1^2) a_2^2 a_1. \end{aligned} \tag{57}$$

This result is added to the data to cancel the internal multiple at $\mathbf{m}_1$ as shown in Figure 18c and equation 37. Furthermore, convolution of $\mathbf{c}_1$ with $\mathbf{m}_1$ will create the annihilator of the second-order multiple, hence $\mathbf{c}_1$ will automatically annihilate all higher-order multiples as well.
To complete the amplitude analysis, the amplitude of the second reflector in $\bar{v}_1^-$ (from the equivalent of equation 43 for the T-MME scheme) can be computed according to equation 48 and constructed from the even amplitude terms in equation 54. The initialization of $\bar{v}_1^-$ is the time reversed shot record (DD in Algorithm 2). Summation of all even $c_{1,i}^a$ iterations i at the time of the second reflector creates the final amplitude for the second reflector in $\bar{v}_1^-$

$$\begin{aligned} a_2 =&(1 - a_1^2) a_2 + \sum_{i=1}^{n_i} a_1^{2*i} (1 - a_1^2) a_2 \\ =&a_2 - a_1^2 a_2 + a_1^2 a_2 - a_1^4 a_2 + a_1^4 a_2 - a_1^6 a_2 + \dots \\ \approx&a_2. \end{aligned} \tag{58}$$

This shows that the transmission compensated local reflectivity can be collected from v_1^- as implemented in the T-MME scheme. The approximation sign is due to a limited number of iterations in the numerical implementation.
Figure 20 is obtained in the same way as Figure 18b, but with high contrast layers. The velocity of the layers is the same as used in Figure 18, but the density contrast between the layers has been increased from a factor 3 (1000-3000) to 10 (500-5000). Compared to Figure 18b the convergence is much slower in this high contrast medium. In equation 55 the higher-order terms will have larger values in high contrast media and require more iterations for convergence.

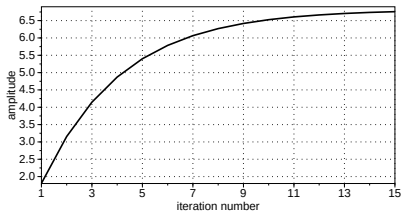


Figure 20: Convergence of the maximum amplitude of the event (labeled $\mathbf{c}_1$ in Figure 18a) that annihilates all the internal multiples between the first and second reflector in a high contrast medium.

4.3.3 Higher iteration counts

The first few iterations for the update terms M_i are shown in Figure 21. The truncation time is chosen at sample 276 and a first-order multiple of the second layer is present in the initialization shot-record M_0 after time-truncation. For higher numbers of iterations the update terms become smaller in amplitude, indicating that the scheme converges. All the updates show the same number of events and only the amplitude of the events change during the iterations.

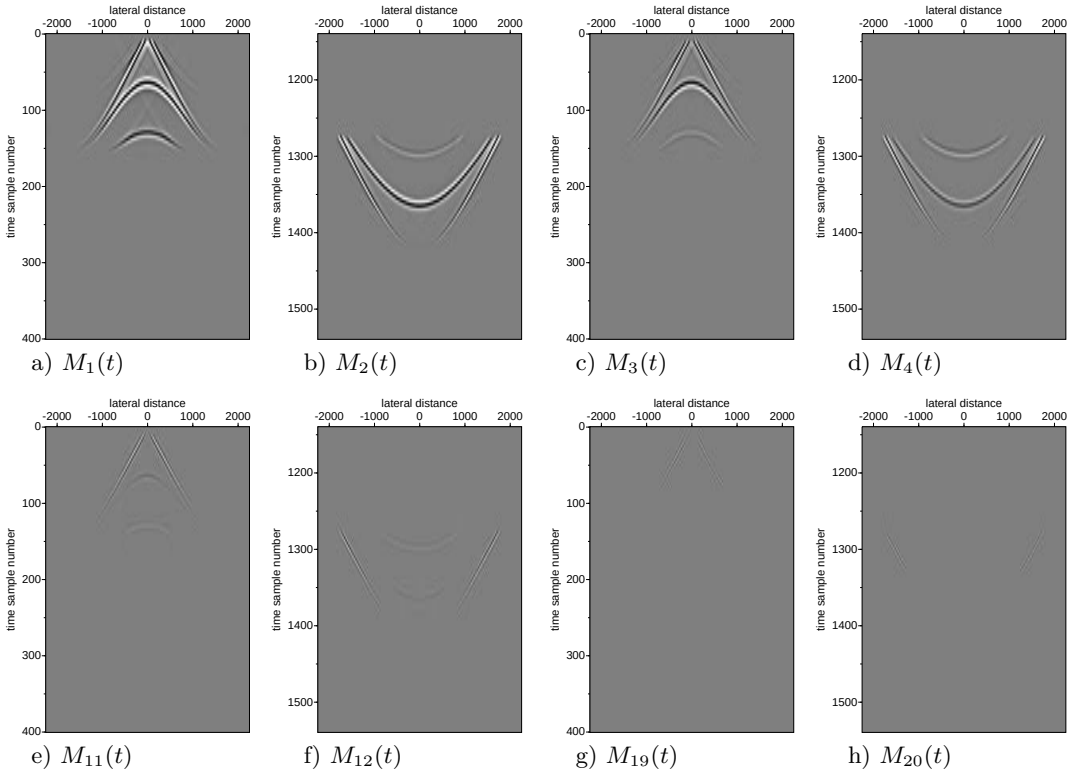


Figure 21: M_i fields for a focal time at sample $ii = 276$; the zero-offset arrival of the third reflector. All figures are plotted with the same clipping factor.

In the odd iterations the function $k^-_{1,i}(t)$, see equation 42, is updated with the odd $M_i(-t)$ terms and four selected iterations are shown in Figure 22. After two iterations all order multiples are predicted, but with incorrect amplitudes. In the following iterations the removal of higher-order multiples is improved because the removal of the first-order multiple improves. After 20 iterations the internal multiple events (indicated with arrows) have further attenuated and are not visible anymore; compare Figure 22b with 22h. The higher-order multiples do not have to be removed by extra events in v^+_1, but are removed automatically by removing the first-order multiple.
In Figure 22f one can observe that the first internal multiple (pointed at by the top arrow) is already attenuated beyond sample $276 - n_\varepsilon + 1$, but is not yet completely attenuated before sample $276 - n_\varepsilon$. The first $276 - n_\varepsilon$ samples belong to v^-_1, where the information on attenuation of the internal multiple is constructed, while samples from $276 - n_\varepsilon + 1$ onward belong to u^-_1, where the multiple is already attenuated. The constant-time cross-section (for all lateral positions) in $k^-_{1,i}$ at sample 276 is stored in the final output R_t at sample 276. In the v^-_1 part (between samples 1 and 276) of k^-_1 the second reflector has its local reflection coefficient as amplitude, while in the u^-_1 part (from sample 276 onward) it has its physical amplitude with two-way transmission effects.

4.3.4 Different time instances

In Figure 23 the Marchenko equations are solved for different time samples ii and it is possible to investigate how k^-_1 changes for larger sample numbers. It is observed that not only at sample ii, but also before and beyond ii the events related to internal multiples are attenuated. Sample point 276 corresponds to the arrival time of the third reflector. The times in Figure 23a-23d are all before sample 276 and we do not observe a change in the number of events. However, going from sample 246 (Figure 23a) to 276 (Figure 23d) one can see that the multiple, arriving in time between the second and third reflector, gets more and more attenuated at larger and larger offsets. This also explains the success of the fast algorithm, to compute the solution in the next time sample there is only a small change needed and a few iterations are sufficient to solve for the multiple attenuation at higher offsets. When the sample

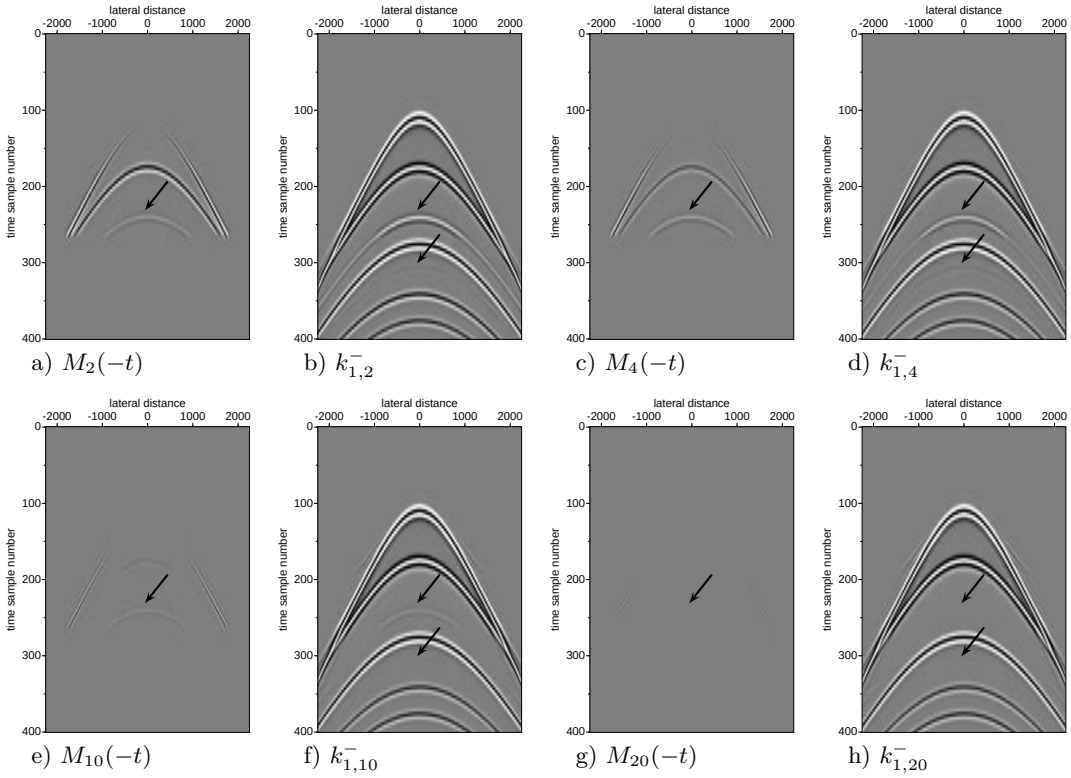


Figure 22: Updates for $k^-_{1,i}$ for a focal time at sample $t_2 = 276$ after i iterations. The arrow indicates the first and second-order internal multiple between the first and second reflector.

time ii passes the arrival time of the third reflector, a non-physical event (pointed at with an arrow in Figure 23e-23h) appears just below the arrival time of the second reflector. This non-physical reflector is the annihilator event in v_1^+ that compensates all internal multiples created between the second and third reflector. The cancelation of the internal multiples is observed at larger time samples (Figure 23e-23h); all internal multiples related to the third reflector are canceled out.
Figure 24a and 24b sketches of the situation where the time instant ii corresponds to a depth above or below the third reflector, respectively. The event that compensates all internal multiples related to the second reflector (green arrow in Figure 24a) coincides in traveltime with the reflection of the second reflector and also compensates for the transmission loss of the reflection from the second reflector. The internal multiples related to the third reflector are compensated by the red-arrow event (Figure 24b) which coincides with the reflection time of the third reflector. This event is also reflected by the second reflector (upward red arrow) and creates the non-physical reflection event observed in Figure 23e-23h and pointed with an arrow.

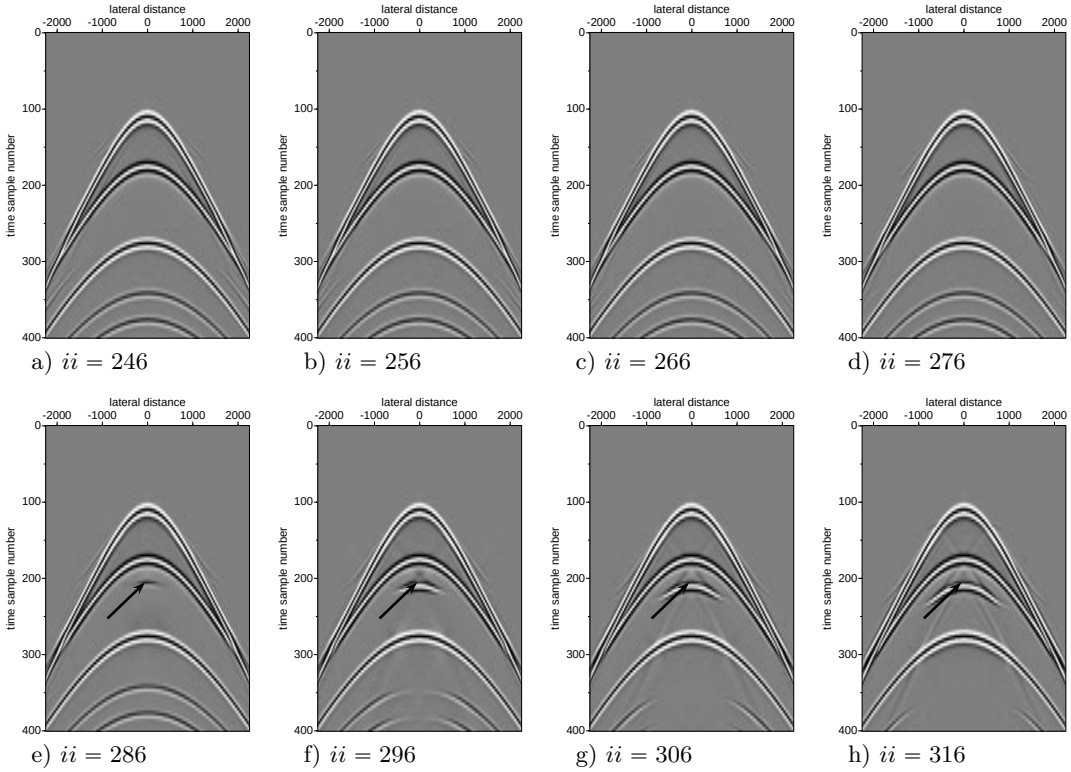


Figure 23: k_1^- after 32 iterations with different time instants $ii = 246$ to $ii = 316$ with steps of 10 samples. From each panel a constant-time cross section is selected at ii and all these cross-sections make up the multiple-free data. The arrows point to an event that compensates all internal multiples created between the second and third reflector.

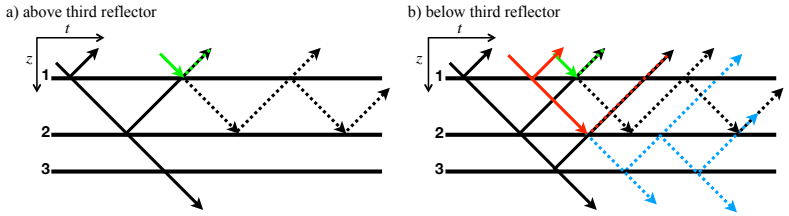


Figure 24: Compensation of internal multiples by events (coloured lines) that are created by the Marchenko method, applied for a point above (a) and below (b) the third reflector. The three reflectors are numbered from top to bottom.

Figures 25a-25d show the same pictures as in Figure 21a-d and Figures 25e-h show the same as Figures 22b,22d,22f,22h, but now with the T-MME scheme (Zhang et al., 2019). The T-MME scheme for time sample 276 includes the reflection of the third reflector, since the time window is now $276 + n_\varepsilon$. This extra reflector introduces new events in $\bar{M}_i$. In the $\bar{k}^-_{1,i}$ terms the non-physical primary, just below the second reflector, is clearly visible. After 20 iterations Figure 25h looks very similar to Figure 23f (time instant 296). The difference is that in the T-MME scheme the truncation starts at $t_2 + \varepsilon$ (sample 276+8) and the value at t_2 (time sample 276) is exactly right in v^- for the local reflection coefficient and stored in the final data output ($R_r[276]$), while in the MME scheme of Figure 23g the truncation starts at $t_2 - \varepsilon$ (sample 296-8) and the value at t_2 (time sample 296) in u^- is the correct value for the physical primary and is stored in the final data output ($R_t[296]$).
In the example for the MME scheme we have shown (in equation 58) that the reflection strength of the second reflector was modified in v_1^- from its physical amplitude to its local reflection coefficient as amplitude. It is exactly this feature that T-MME exploits. When the Marchenko schemes reaches the arrival time of a reflector there is a decision to be made where to put the reflection of that reflector. Setting the truncation time to $t_2 - \varepsilon$ the time-instant t_2 is correctly obtained in u_1^-. Changing the truncation time from $t_2 - \varepsilon$ to $t_2 + \varepsilon$, the time-instant t_2 is correctly obtained in v_1^- instead of in u_1^-. It is the time duration of the source wavelet that allows us to make this choice. By taking $t_2 - \varepsilon$ an error is introduced in v_1^- and hence u_1^- is correct at t_2, whereas by taking $t_2 + \varepsilon$ the error is in u_1^- and v_1^- will be correct at t_2.
The transmission compensated (T-MME) scheme retrieves primary reflections with local reflection coefficients, while in the regular (MME) scheme the primary reflections keep their two-way reflection coefficients that include transmission losses. The local reflection retrieval of the T-MME scheme is exact for a horizontally layered medium, but in laterally varying media it is approximately true (Zhang et al., 2019). The only computational difference between the T-MME and MME schemes is the position of the time-truncation window.

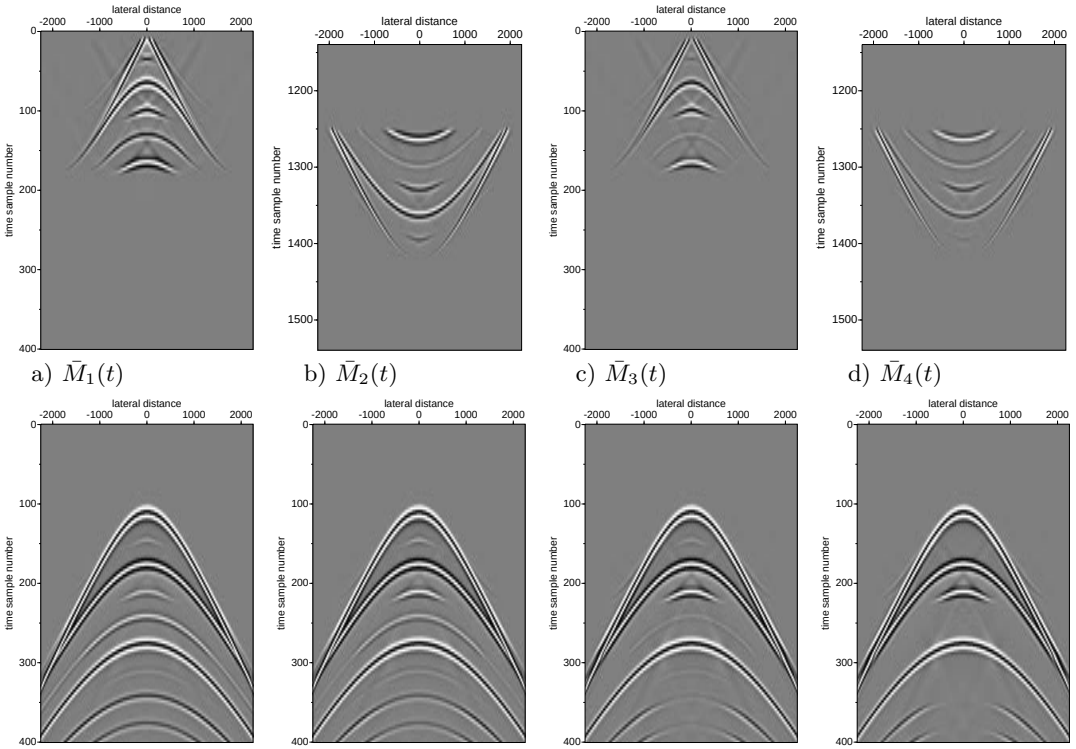


Figure 25: Panels (a)-(d) show the $\bar{M}_i$ fields for a focal time at sample $t_2 = 276$ with the transmission compensated scheme T-MME after $i = 1, 2, 3, 4$ iterations. Panels (e)-(h) show the $\bar{k}_i^-$ fields for a focal time at sample $t_2 = 276$ with the transmission compensated scheme T-MME after $i = 2, 4, 10, 20$ iterations. All figures are plotted with the same clipping factor.

4.4 Parameters in program `marchenko_primaries`

The `marchenko_primaries` program has the following parameters and options:

```
MARCHENKO_primaries - Iterative primary reflections retrieval

marchenko_primaries file_tinv= file_shot= [optional parameters]

Required parameters:

  file_shot= ............... Reflection response: R

Optional parameters:

INTEGRATION
  ishot=nshots/2 ........... shot number(s) to remove internal multiples
  file_tinv= ............... shot-record to remove internal multiples
  file_src= ................ optional source wavelet to convolve selected ishot(s)
COMPUTATION
  tap=0 .................... lateral taper R_ishot(1), file_shot(2), or both(3)
  ntap=0 ................... number of taper points at boundaries
  fmin=0 ................... minimum frequency in the Fourier transform
  fmax=70 .................. maximum frequency in the Fourier transform
  plane_wave=0 ............. model plane wave
  src_angle=0 .............. angle with horizontal of plane source array
  src_velo=1500 ............ velocity to use in src_angle definition
  t0=0.1 ................... time shift in plane-wave source wavelet for migration
MARCHENKO ITERATIONS
  niter=22 ................. number of iterations to initialize and restart
  niterec=2 ................ number of iterations in recursive part of the time-samples
  niterskip=50 ............. restart scheme each niterskip samples with niter iterations
  istart=20 ................ start sample of iterations for primaries
  iend=nt .................. end sample of iterations for primaries
MUTE-WINDOW
  shift=20 ................. number of points to account for wavelet (epsilon in papers)
  smooth=shift/2 ........... number of points to smooth mute with cosine window
REFLECTION RESPONSE CORRECTION
  tsq=0.0 .................. scale factor n for t^n for true amplitude recovery
  Q=0.0 .....,.............. Q correction factor
  f0=0.0 ................... ... for Q correction factor
  scale=2 .................. scale factor of R for summation of Mi with M0
  pad=0 .................... amount of samples to pad the reflection series
OUTPUT DEFINITION
  file_rr= ................. output file with primary only shot record
  file_dd= ................. output file with input of the algorithm
  file_iter= ............... output file with -Mi(-t) for each iteration: writes
              ............... M0.su=M0 : initialisation of algorithm
              ............... RMi: iterative terms
              ............... k1min.su: k1min terms
  file_vplus= .............. output file with v+
  file_vmin= ............... output file with v-
  file_uplus= .............. output file with u+
  file_umin= ............... output file with u-
  file_update= ............. output file with updates only => removed internal multiples
  T=0 ...................... :1 compute transmission-losses compensated primaries
  verbose=0 ................ silent option; >0 displays info


author  : Lele Zhang & Jan Thorbecke : 2020
```

Defining `file_iter` writes for each iteration the focusing update term $-M_i(-t) = RM_i(t)$ in Algorithm 1 before applying the mute window. This same option will also write $k^-_{1,i}$ and $v^+_{1,i}$ after the update. By setting the `verbose=` option to 2 the energy of the focusing update term is printed out for each iteration and can be used to monitor the convergence of the scheme. When `file_update=` is given an output name the program writes the updates (= estimated internal multiples) to disk. The `scale` parameter can be useful when the (modeled) data does not have the correct amplitude.
The parameter `niterskip=` enables the fast algorithm when it is set larger than 1. The first instant time value `istart=` is run with `niter=` iterations. If `niterskip=` is set to a value > 1 the fast algorithm is in effect and the next `niterskip` iterations use only `niterec=2` iterations. After `niterskip` fast iterations

the scheme uses the full `niter` iterations to avoid possible cumulative numerical instabilities caused by amplified artefacts. The scheme continues with `niterec` fast algorithm iterations and the cycle repeats itself. By setting `niterec=0` the scheme does not do any new iterations in the fast cycle and directly uses the result of the previous iteration. The `niterec=0` setting will work well if `niterskip=` is set to ≈`shift` samples and is possible due to limited bandwidth of the data.
The `T=` parameter is a switch to enable the T-MME algorithm. The options `plane_wave, src_angle, src_velo, xorig` use plane-waves as input shot record as explained in Meles et al. (2018, 2020).
The commands to reproduce all figures in this paper can be found in the directory `marchenko/demo/mme`. The README_PRIMARIES in that directory explains in detail how to run the scripts. A more complicated (lateral varying) model can be found in the directory `marchenko/demo/twoD`. This example will take several hours to compute the reflection data and is not discussed here.
Besides the new Marchenko primaries removal program the package also contains the earlier published finite difference modeling code, that is used to model all data in the examples, in directory `fdelmodc` (Thorbecke and Draganov, 2011), and the standard Marchenko programs (Thorbecke et al. (2017)). The directory `utils` contains programs to calculate a gridded model (`makemod`), source wavelets (`makewave`) and programs for basic processing steps.

4.5 Examples to run the code

```
Description of files:

1) model.scr computes the model and the 'basis' shot of R => shot5_rp.su
   - runtime on 4 cores is 4-5 minutes and produces a 3.3 GB data file with 901 shots
2) itertions.scr computes the intermediate results of the multiple attenutation scheme and produces al
nuscript.
   - runtime on 4 cores is
3) epsPrimaries.scr selected output from step 2) are converted to .eps pictures that are used in the F
   To reproduce the postscript files of the manuscript SU postscript plotting programs are required.
-3) epsModel.scr to generate the postscript files for the numerical model

optional scripts not needed to reproduce the figures:
+) primaries.scr computes the internal multiple attenuated (middle) shot record.
   - runtime on 4 cores is ~500 s.
+) primariesPlane.scr: computes the internal moval scheme for plane-waves (see Meles 2020)
+) clean: remove all produced files and start with a clean directory

--------------------------
To reproduce the Figures in the Manuscript:
--------------------------
* Figure 2: Model + Initial wavefield

==> run model.scr to generate the data .su files: this will take 4-5 minutes. The files generate are:
- hom_cp.su, hom_ro.su
- model10_cp.su, model10_ro.su
- shot5_fd_rp.su
- shot5_hom_fd_rp.su
- shot5_rp.su
- wavefw.su

==> run './epsPrimaries.scr Figure2' to generate the postscript files of Figure 2

model_cp_line.eps  => Figure 2a
model_ro_line.eps  => Figure 2b
shotx0_rp.eps  => Figure 2c

It also produces two extra pictures of the wavelet used in the FD modelling:
wavefw_freq.eps
```

```
wavefw.eps

--------------------------
* Figure 3: First Iteration

==> run './iterations.scr Figure34910' to compute the intermediate results of the first iterations of
This will take 15 seconds. The generated files are:

- M0_276000.su
- Mi_2760##.su
- k1min_2760##.su
- v1plus_2760##.su
- iter_2760##.su (not used)
- pred_rr_276.su (not used)
- DDshot_450.su (not used): selected shot record convolved with file_src=wave.su

where ## ranges from 01 to 34

To generate the postscript files for Figure 3:

==> run './epsPrimaries.scr Figure3'

This will produce the following files:

shotx0_rp.eps  => Figure 2c == Figure 3a
M0_276000_flip.eps => Figure 3b
fconvN0fulltime.eps => Figure 3c
fconvN0flip.eps  => Figure 3d
Mi_276001.eps => Figure 3e


--------------------------
* Figure 4 second iteration
To generate the postscript files for Figure 4:
==> run './epsPrimaries.scr Figure4'

This will produce the following files:

fconvN1fulltime.eps => Figure 4c
fconvN1flip.eps => Figure 4d
Mi_276002.eps => Figure 4e

The window time function in Figure 5 is not reproduced.

--------------------------
* Figure 6 v1plus and convergence
==> run './iterations.scr Figure6' to compute the marchenko results with ii=200

To generate the postscript files for Figure 6:
==> run './epsPrimaries.scr Figure6'

This will produce the following files:
v1plus_200001.eps => Figure 6a
v1plus_max.eps => Figure 6b
k1min_200030.eps => Figure 6b

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

```
* Figure 8 To compute the convergence for a strong contrast medium:
cd strongContrast
==> run ./model.scr
==> run ./iterations.scr Figure8

To generate the postscript files for Figure 8:
==> run './epsPrimaries.scr Figure8'

This will produce the following files:
v1plusStrong_max.eps => Figure 8

Don't forget to go back to the main directory with the regular contrast results
cd ../


-------------------------
* Figure 9 iterations M_i

To generate the postscript files for Figure 9:
==> run './epsPrimaries.scr Figure9'

This will produce the following files:
Mi_276002.eps => Figure 9b
Mi_276004.eps => Figure 9d
Mi_276012.eps => Figure 9f
Mi_276020.eps => Figure 9h
Mi_276001.eps => Figure 9a
Mi_276003.eps => Figure 9c
Mi_276011.eps => Figure 9e
Mi_276019.eps => Figure 9g


-------------------------
* Figure 10 iterations M_i and k_1^-

To generate the postscript files for Figure 10:
==> run './epsPrimaries.scr Figure10'

This will produce the following files:
Mi_276002flip.eps => Figure 10a
k1min_276002.eps => Figure 10b
Mi_276004flip.eps => Figure 10c
k1min_276004.eps => Figure 10d
Mi_276010flip.eps => Figure 10e
k1min_276010.eps => Figure 10f
Mi_276020flip.eps => Figure 10g
k1min_276020.eps => Figure 10h


-------------------------
* Figure 11 iterations k_1^- for different ii 246:316:10

To generate the data
==> run ./iterations.scr Figure11
this will take ~2 minutes and generate a lot of files

To generate the postscript files for Figure 11:
```

```
==> run './epsPrimaries.scr Figure11'

This will produce the following files:
k1min_246032.eps => Figure 11a
k1min_256032.eps => Figure 11b
k1min_266032.eps => Figure 11c
k1min_276032.eps => Figure 11d
k1min_286032.eps => Figure 11e
k1min_296032.eps => Figure 11f
k1min_306032.eps => Figure 11g
k1min_316032.eps => Figure 11h


--------------------------
* Figure 13 iterations M_i and k_1^- for ii-276 T-MME scheme

To generate the data
==> run ./iterations.scr Figure13
this will take ~15 seconds
****NOTE this will overwrite the results of the MME-scheme !

To generate the postscript files for Figure 13:
==> run './epsPrimaries.scr Figure13'

This will produce the following files:
Mi_276002T.eps => Figure 13a
Mi_276004T.eps => Figure 13b
Mi_276001T.eps => Figure 13c
Mi_276003T.eps => Figure 13d
k1min_276002T.eps => Figure 13e
k1min_276004T.eps => Figure 13f
k1min_276010T.eps => Figure 13g
k1min_276020T.eps => Figure 13h
```

5 Plane-Wave Marchenko algorithm

This is mainly a copy of our publication in Computer & Geosciences.

5.1 Introduction

Seismic imaging is a technique to image geological structures in the subsurface of the earth from reflected wavefields measured at the surface of the earth. The measured wavefields usually originate from human-activated and controlled sources such as air-guns or vibrating plates. In passive seismic methods the source of the wavefield can originate from earthquakes, ocean waves, or uncoordinated human activities as traffic. The primary reflection of a geological structure, large and strong enough to be detected by a propagating wavefield, is of main interest and is used to compute an image of the subsurface. At each geological structure, wavefields are partly reflected upward and partly transmitted further downward. Between two strong reflecting structures, the wavefield can bounce up and down multiple times and generate so called internal multiples. These multiple reflections are also measured by geophones at the surface and difficult to distinguish from primary reflections. In the imaging step the reflections are migrated from time to depth and construct an image of the subsurface. If multiple reflections are not recognized as such, they will get imaged being primary reflections at wrong depths. These falsely imaged multiples distort the actual image of the subsurface; the distorted image contains much more (ghost) structures that are positioned along with the primary reflections. Therefore, in seismic imaging, it is important to recognize these multiple reflections, and if possible, remove them from the computed image. This removal can be performed at different stages of the processing scheme to construct an image of the subsurface. The internal multiples can be directly removed from the measured reflection data, in the redatuming step or after the imaging step. For removal after the imaging step, a computed prediction of imaged multiples is subtracted from the image to obtain an image without multiples. In this paper, we discuss a method for removing internal multiples during the redatuming step.

Besides internal multiples that are reflected between boundaries within the subsurface, there are also free-surface-related multiples. These multiples are generated by reflections from upcoming waves that bounce back into the subsurface by the surface of the earth. Free surface multiples are not considered in this paper. They are assumed to be removed prior to the removal of the internal multiples.

The Marchenko algorithm can eliminate internal multiples from seismic reflection data (Slob et al., 2014; Behura et al., 2014). In this algorithm the up- and down-going focusing functions, with a focal-point in the subsurface, are key to the method. The goal of the Marchenko method is to retrieve these up- and down-going parts of the focusing functions from the reflection data by solving a coupled set of the so-called Marchenko equations. This set of equations can be solved by iterative methods (Wapenaar et al., 2014a; Thorbecke et al., 2017), or a direct method (van der Neut et al., 2015a; Ravasi, 2017). The Marchenko method has found many different applications, ranging from redatuming for time-lapse monitoring (van IJsseldijk et al., 2023), adaptive subtraction of Marchenko estimated multiples (Staring et al., 2018), homogeneous Green's function retrieval (Brackenhoff et al., 2019), and direct multiple elimination on reflection data (Zhang and Slob, 2020b). In this paper, a particularly efficient application of the Marchenko method for imaging by plane-waves is highlighted, and the implementation details of this method are discussed in more detail.

Meles et al. (2018) show that besides focal-points, focal-planes can also be solved by the Marchenko equations. Meles et al. (2020) build on the Marchenko Multiple Elimination (MME) method proposed by Zhang and Slob (2019) and introduce the plane-wave MME method. The major advantage of the plane-wave-based Marchenko method is that with a minimal effort of one Marchenko run (for a single plane-wave), for each depth level (or time instant for MME), a multiple free image can be built up. Especially in 3D applications, the plane-wave Marchenko method is computationally efficient for building an internal multiple-free image (Brackenhoff et al., 2022). A single plane-wave can be sufficient to build an accurate image if the subsurface interfaces are near-flat. Multiple plane-waves, with different illumination angles, are needed for subsurface interfaces with varying dips, or geologically complex structures, to illuminate the subsurface properly (Almobarak, 2021). The Marchenko algorithm for the focal-plane method is similar to the focal-point algorithm. The initial point-focusing function is in the plane-wave algorithm replaced by a time-reversed direct plane-wave response. The main difference lies in the choice of the time windows to separate the Green's functions from the focusing functions. The minimum conditions to hold for a plane wave are the same as for a point source; a separation in time can be made between the direct and later arrivals. In this paper, we discuss in detail how these time windows are adapted for

the Marchenko plane-wave method, discuss the implementation aspects, and illustrate the method with applications on numerically modeled and field data.
The software accompanied by this paper contains scripts and source code to reproduce all the numerical examples presented in this paper. The code can also be found in its GitHub repository (Thorbecke et al., 2017; Thorbecke and Brackenhoff, 2019), where the most recent updated version and the latest developments are available. To reproduce the figures and perform a few pre- and post-processing steps, Seismic Unix (Cohen and Stockwell, 2016) is required.
Most pictures in this section of the manual can be reproduced by the scripts in `...OpenSource/marchenko/demo/planewave`.

5.2 Theory

The Marchenko method is introduced by two coupled equations that contain four unknown fields (up- and downgoing focusing functions and Green's functions) we would like to retrieve. These fields enable us to suppress internal multiples by using the up- and downgoing focusing functions to redatum seismic reflection data from the acquisition surface $\mathbb{D}_0$ to the focal level(s) in the subsurface. In this redatuming step, the internal multiples of the overburden are suppressed. The seismic reflection data is recorded at acquisition boundary $\mathbb{D}_0$. The reflection response $R(\mathbf{x}_R, \mathbf{x}_S, t)$, a scaled version of the Green's function without the direct arrival (Wapenaar et al., 2012), is measured with sources and receivers positioned at $\mathbf{x}_S$ and $\mathbf{x}_R$ on this boundary. The recording time is denoted by t. This reflection response does not contain free-surface related multiple reflections neither a source wavelet. Hence, pre-processing steps are required to remove the free-surface multiples and the direct arrival, and to deconvolve the wavelet from the measured reflection data.
The up- and downgoing parts of the focusing functions f_1^- and f_1^+ are used to define a relation between the decomposed Green's functions $G^{-,+}$ and $G^{-,-}$ in the actual medium and with the reflection response at the surface Wapenaar et al. (2014b). The focusing functions have a focal point in the subsurface at $\mathbf{x}_A$. This focal point serves as a virtual source for the Green's function. The rightmost + and − superscripts of the decomposed Green's function refer to the direction of propagation (+ for down and − for up) from the virtual source at $\mathbf{x}_A$. The leftmost superscript indicates an up-(−) or downward(+) propagating field at the receiver locations. The relation between the two unknown focusing functions and the two unknown Green's functions is given by the following two equations (Wapenaar et al., 2021);

$$G^{-,+}(\mathbf{x}_R, \mathbf{x}_A, t) + f_1^-(\mathbf{x}_R, \mathbf{x}_A, t) = \int_{\mathbb{D}_0} \int_{t'=0}^{\infty} R(\mathbf{x}_R, \mathbf{x}_S, t') f_1^+(\mathbf{x}_S, \mathbf{x}_A, t - t') \mathrm{d}t' \mathrm{d}\mathbf{x}_S, \tag{59}$$

$$G^{-,-}(\mathbf{x}_R, \mathbf{x}_A, t) + f_1^+(\mathbf{x}_R, \mathbf{x}_A, -t) = \int_{\mathbb{D}_0} \int_{t'=0}^{\infty} R(\mathbf{x}_R, \mathbf{x}_S, t') f_1^-(\mathbf{x}_S, \mathbf{x}_A, t' - t) \mathrm{d}t' \mathrm{d}\mathbf{x}_S. \tag{60}$$

In the compact operator notation of Van der Neut et al. (2015b) equations 59 and 60 are written as

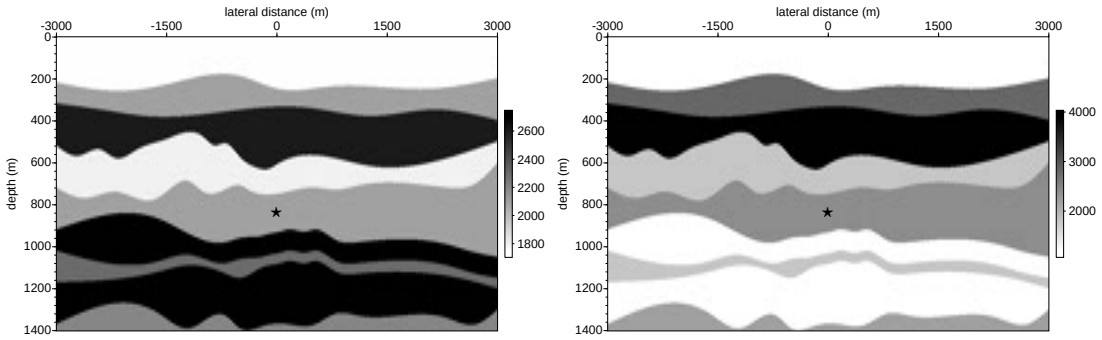
$$G^{-,+} + f_1^- = R f_1^+, \tag{3}$$

$$G^{-,-} + f_1^{+\star} = R f_1^{-\star}, \tag{4}$$

where * denotes the time-reverse. These two equations contain four unknowns: two Green's functions and two focusing functions. The only known in these equations is the measured reflection response R. Wapenaar et al. (2013) use the reasoning that the Green's function and focusing function can, under certain circumstances, be separated in time. Therefore, a time window function (Wapenaar et al., 2014b) $\Theta(t)$ is defined that passes the focusing function and removes the Green's function from the left-hand side of equations 3 and 4. For point-sources that radiate in all directions, one time window is sufficient for both the up- and downgoing traveling waves. Up- and downgoing plane-waves, on the other hand, propagate at opposite dipping angles; hence, two time windows are needed in the plane-wave algorithm. These time windows remove all events that arrive at later times than the direct wave traveling from the virtual source position $\mathbf{x}_A$ to the receiver position $\mathbf{x}_R$ at surface $\mathbb{D}_0$, including the direct wave itself. This results in the following two equations that only have two unknowns (assuming the direct arrival $f_{1,d}^+$ is known)

$$f_1^- = \Theta_b R f_1^+, \tag{5}$$

$$f_1^{+\star} - f_{1,d}^{+\star} = \Theta_a R f_1^{-\star}, \tag{6}$$



a) velocity model in m/s b) density model in kg/m^3

Figure 26: Multi layer model with velocity (a) and density (b) parameters. The location of the virtual point-source is marked with a $\star$.

where $f_1^+ = f_{1,m}^+ + f_{1,d}^+$, with $f_{1,d}^+$ the direct arrival of f_1^+, and $f_{1,m}^+$ events that arrive before the direct arrival time t_d. The separation between $f_{1,d}^+$ and $f_{1,m}^+$ can be successfully applied when there are no overlapping reflection events with the direct response. In other cases, separation can still be applied, but requires additional steps to overcome interference (Zhang et al., 2019). These time windows (Wapenaar et al., 2021) are defined as

$$\Theta_b(t) = \theta(t_b - t), \tag{7}$$

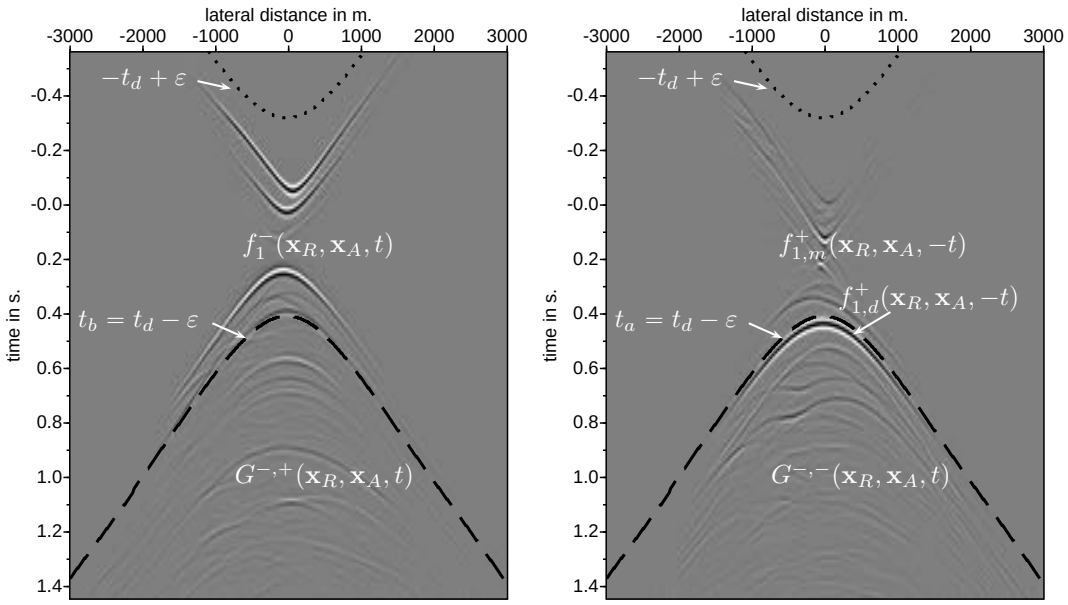
$$\Theta_a(t) = \theta(t_a - t), \tag{8}$$

where $\theta(t)$ denotes a tapered Heaviside step function. Note that two time windows are defined: one at time t_a and one at t_b. In the point-source algorithm, $t_b = t_d - \varepsilon = t_a$ which makes equation 8 equal to equation 7, with the window function $\theta(t_d - \varepsilon - t)$. The ε takes into account the finite length of the band-limited wavelet and ensures that the direct wave is removed from the right-hand side of equation 6 (Broggini et al., 2014). Epsilon is typically chosen as half the dominant wavelength.
To illustrate the application of the time windows, a virtual point-source is defined at a depth of 800 m in the laterally varying model of Figure 26 (after Meles et al., 2018).
Figure 27 shows the focusing functions and Green's functions for the model of Figure 26, and the window functions (indicated with a dashed line) that separate these functions. Figure 27a represents the left-hand side of equation 3 and the time window that separates f_1^- from $G^{-,+}$. In Figure 27b, the time window separates the time-reversal of $f_{1,m}^+$ from $G^{-,-}$ and represents the left-hand side of equation 4. The convolution/correlation in the right-hand sides of equations 3 and 4 is in practice carried out in the frequency domain, and a discrete Fourier transform in time is used to transform the sampled reflection data into the frequency domain. The discrete Fourier transform makes the fields periodic in time, with a periodicity equal to the number of time samples (n_t). Given this periodicity in time, reflection events occurring in time beyond n_t, end-up in negative times. The time windows defined in equations 7 and 8 pass all events earlier in time than $t = t_a$ and will also pass these time wrap-around events. To exclude these wrap-around events a time window is also implemented for negative times. In Figure 27a, we can see that the focusing functions also include events at negative times, and that these events are not present earlier than $-t_a = -t_d + \varepsilon$. Hence, the cutoff point of the time window at negative times is chosen at $-t_a$. The implemented time windows become

$$\Theta_b'(t) = \theta(t_b - t) - \theta(-t_b - t), \tag{9}$$

$$\Theta_a'(t) = \theta(t_a - t) - \theta(-t_a - t), \tag{10}$$

and the time windows at negative times, to suppress time wrap-around, are indicated with the dotted lines in Figure 27. There is no guarantee that this time window suppresses all wrap-around. If these windows are not sufficient to suppress the wrap-around, zeros can be padded to the reflection response.
To solve the unknown focusing functions in the coupled equations 5 and 6 different methods are developed. The iterative method described in Behura et al. (2014) and Wapenaar et al. (2014b) start with $f_{1,d}^+$ as the initial solution of f_1^+ and solve equation 5 for f_1^- and substitute the solution in equation 6 to update f_1^+. This process is repeated until the updates to the focusing functions become very small. This iterative



a) Separation of $G^{-,+}$ and f_1^-

b) Separation of $G^{-,-}$ and $f_1^{+\star}$

Figure 27: Illustration of the time window function to separate the Green's function from the focusing function. The dashed black lines indicate the separation line of the time window and are indicated with white arrows. The dotted line indicates the time window that suppresses time wrap-around.

algorithm to solve the Marchenko equations is shown in Figure 28. The even iterations (starting the iteration count at 0 for the initial solution) in the scheme solve equation 5 and the odd iterations solve equation 6. Thorbecke et al. (2017) explain in more detail the implementation of this iterative algorithm. Depending on the application it is not always needed to solve these equations until convergence. In Staring et al. (2018) the results of the first iteration are used to predict internal multiples and an adaptive subtraction method is used to suppress the predicted multiples. A direct least-squares inversion method to solve the coupled equations is discussed in Van der Neut et al. (2015b) and Ravasi (2017).
Meles et al. (2018) show that plane-wave focusing functions $\tilde{f}_1^+$ and $\tilde{f}_1^-$ and associated plane-wave Green's functions $\tilde{G}^{-,+}$ and $\tilde{G}^{-,-}$ can be obtained by integrating an appropriate set of time-delayed focusing functions f_1^+ and f_1^-, each involving the solution of a Marchenko equation. The tildes represent plane-wave quantities for focusing functions and Green's functions. The plane-wave focusing functions (Wapenaar et al., 2021; Brackenhoff et al., 2022) are defined by the following integration

$$\tilde{f}_1^{\pm}(\mathbf{x}, \mathbf{p}_A, t) = \int_{\mathbb{D}_A} f_1^{\pm}(\mathbf{x}, \mathbf{x}_A, t - \mathbf{p} \cdot \mathbf{x}_{H,A}) d\mathbf{x}_A, \tag{11}$$

with $\mathbf{p} = (p_1, p_2)$ and p_1 and p_2 horizontal ray parameters and $\mathbf{p}_A = (\mathbf{p}, x_{3,A})$ the ray parameter of the plane-wave at surface $\mathbb{D}_A$. The surface $\mathbb{D}_A$ is the depth level at which focusing takes place. The plane-wave Green's functions are defined by a similar integration. Here $\mathbf{x}_{H,A} = (x_{1,A} - x_{1,c}, x_{2,A} - x_{2,c})$, and $(x_{1,c}, x_{2,c})$ is the rotation point of the plane-wave. The rotation point is chosen in the center of the lateral extent of the plane-wave. This rotation point also defines the time origin $t = 0$ of the plane-wave. By making this choice for $t = 0$, the time-axis in the computed plane-wave Green and focusing functions is the same as in the point-source Marchenko solutions with a focal point at the rotation point of the plane-wave.
Note that the plane-wave integration in equation 11 for a time-reversed wave-field $P(\mathbf{x}, \mathbf{x}_A, -t)$ gives

$$\tilde{P}(\mathbf{x}, \mathbf{p}'_A, -t) = \int_{\mathbb{D}_A} P(\mathbf{x}, \mathbf{x}_A, -(t - \mathbf{p} \cdot \mathbf{x}_{H,A})) d\mathbf{x}_A, \tag{12}$$

with $\mathbf{p}'_A = (-\mathbf{p}, x_{3,A})$, a plane-wave dipping with the *opposite* angle as a plane-wave with $\mathbf{p}$. On a surface

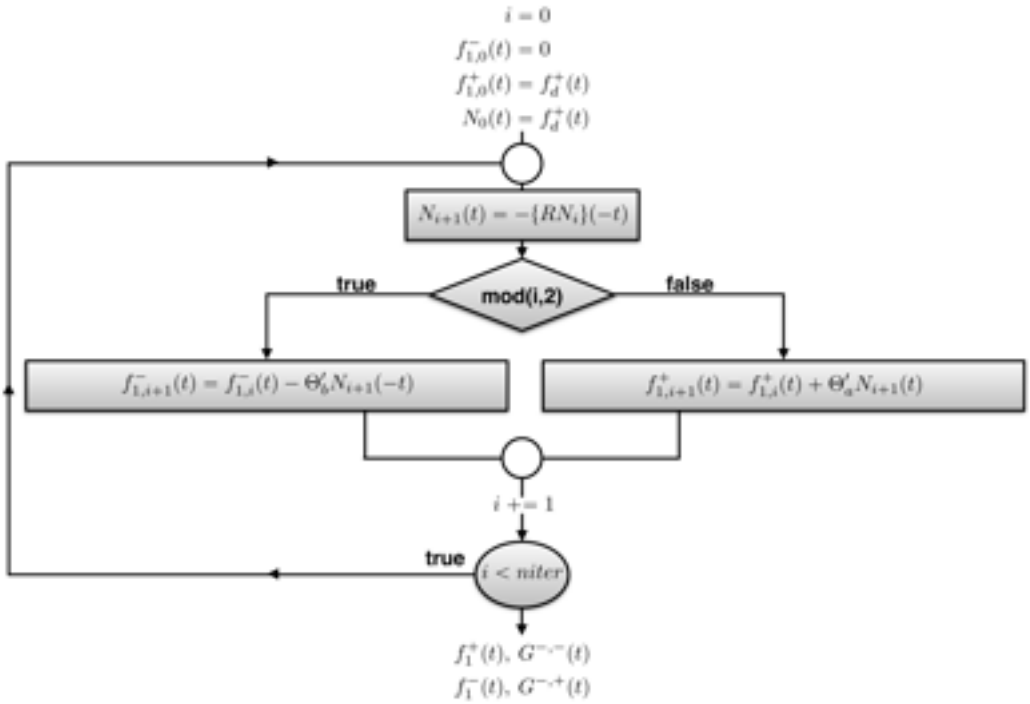


Figure 28: Flow chart of the Marchenko algorithm. The down- (f_1^+) and up-going(f_1^-) focusing functions are alternately updated. The scheme is finished after a pre-defined number of iterations and is usually chosen between 10-20 iterations.

$\mathbb{D}_A$ with a homogeneous velocity along the surface, $\mathbf{p} \cdot \mathbf{x}_{H,A}$ are time shifts that are linearly proportional to the distance from the rotation point.
Applying the same integration as in equation 11 over all fields results in the plane-wave representations of equations 3 and 4 (Meles et al., 2018)

$$\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}_A, t) + \tilde{f}_1^{-}(\mathbf{x}_R, \mathbf{p}_A, t) = \{R\tilde{f}_1^{+}\}(\mathbf{x}_R, \mathbf{p}_A, t), \tag{13}$$

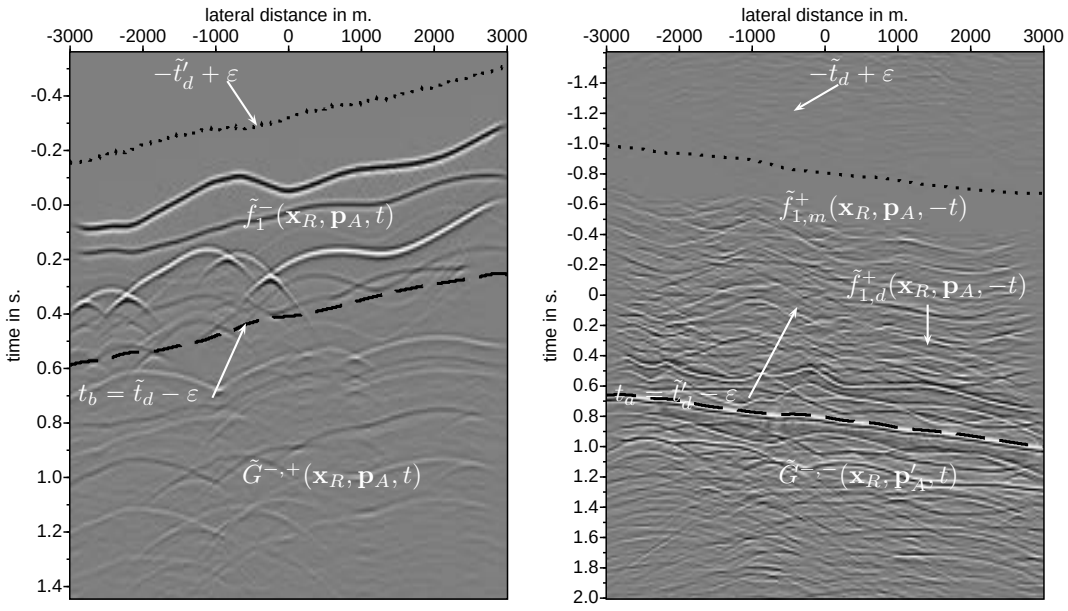
$$\tilde{G}^{-,-}(\mathbf{x}_R, \mathbf{p}_A', t) + \tilde{f}_1^{+\star}(\mathbf{x}_R, \mathbf{p}_A, t) = \{R\tilde{f}_1^{-\star}\}(\mathbf{x}_R, \mathbf{p}_A, t). \tag{14}$$

Applying a time window that separates the Green function from the focusing function gives again two equations with two unknowns (assuming $\tilde{f}_{1,d}^{+\star}(\mathbf{x}_R, \mathbf{p}_A, t)$ is known);

$$\tilde{f}_1^{-}(\mathbf{x}_R, \mathbf{p}_A, t) = \tilde{\Theta}_b\{R\tilde{f}_1^{+}\}(\mathbf{x}_R, \mathbf{p}_A, t), \tag{15}$$

$$\tilde{f}_1^{+\star}(\mathbf{x}_R, \mathbf{p}_A, t) - \tilde{f}_{1,d}^{+\star}(\mathbf{x}_R, \mathbf{p}_A, t) = \tilde{\Theta}_a\{R\tilde{f}_1^{-\star}\}(\mathbf{x}_R, \mathbf{p}_A, t) \tag{16}$$

with $\tilde{f}_1^{+\star} = \tilde{f}_{1,m}^{+\star} + \tilde{f}_{1,d}^{+\star}$, where $\tilde{f}_{1,d}^{+\star}$ is the direct arrival of the plane-wave with a propagation angle defined by $(\mathbf{p}, x_{3,A})$, and $\tilde{f}_{1,m}^{+\star}$ contains the events that arrive before the direct arrival time $\tilde{t}_d$, where $\tilde{t}_d$ is the first arrival time of a plane-wave with a propagation angle defined by $(\mathbf{p}, x_{3,A})$. Note that the Green's functions in equations 13 and 14 contain plane-waves with opposite dipping angles defined by $\mathbf{p}_A$ and $\mathbf{p}_A'$. To separate $\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}_A, t)$ from $\tilde{f}_1^{-}(\mathbf{x}_R, \mathbf{p}_A, t)$ the direct arrival $\tilde{t}_d$, that is the first arrival time of $\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}_A, t)$, is required. The notation for $\tilde{t}_d$ or $\tilde{t}_d'$ is a choice, our choice is the same as made in Wapenaar et al. (2021). This choice uses the $'$ on $\tilde{t}_d$ the same as the $'$ on the propagation angle $\mathbf{p}_A$ in the upgoing Green's functions $\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}_A, t)$ and $\tilde{G}^{-,-}(\mathbf{x}_R, \mathbf{p}_A', t)$. Note that the defined arrival time $\tilde{t}_d'$ has the same arrival time of an upward propagating modeled plane-wave at $(x_{3,A})$ with propagation angle $\mathbf{p}_A'$.
The time window $\tilde{\Theta}_b(t)$ removes $\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}_A, t)$ from equation 13. The first non-zero contribution of $\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}_A, t)$ is at time $t_b = \tilde{t}_d - \varepsilon$. The time window $\tilde{\Theta}_a(t)$ removes $\tilde{G}^{-,-}(\mathbf{x}_R, \mathbf{p}_A', t)$ and $\tilde{f}_{1,d}^{+\star}(\mathbf{x}_R, \mathbf{p}_A, t)$ from equation 14, hence all events at times later than $t_a = \tilde{t}_d' - \varepsilon$ are set to zero. Similar to the point-source



a) Separation of $\tilde{G}^{-,+}(\mathbf{p}_A)$ and $\tilde{f}_1^-(\mathbf{p}_A)$.

b) Separation of $\tilde{G}^{-,-}(\mathbf{p}'_A)$ and $\tilde{f}_1^{+\star}(\mathbf{p}_A)$.

Figure 29: Illustration of the time window function to separate the plane-wave Green's function from the focusing function. The dashed black lines indicate the separation line of the time window and are indicated with white arrows. The dotted line indicates the time window that suppresses time wrap-around.

scheme these time windows are implemented with an additional window at negative times to suppress time wrap-around and are given by

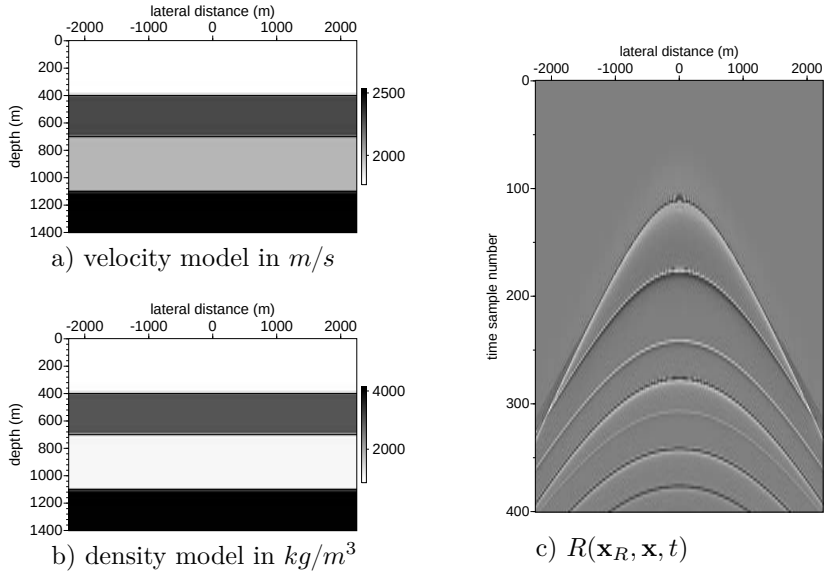
$$\tilde{\Theta}'_b(t) = \theta(\tilde{t}_d - \varepsilon - t) - \theta(-\tilde{t}'_d + \varepsilon - t), \tag{17}$$

$$\tilde{\Theta}'_a(t) = \theta(\tilde{t}'_d - \varepsilon - t) - \theta(-\tilde{t}_d + \varepsilon - t). \tag{18}$$

Similar to Figure 27, Figure 29 shows the plane-wave focusing functions and Green's functions, as in the left-hand side of equations 13 and 14, and the window functions separating them. In this example the depth of the plane-waves is chosen at 800 m in the model shown in Figure 26. In the following section these two time window functions are discussed in more detail and the differences with the point-source implementation of the Marchenko algorithm are explained.

5.3 Basic algorithm

The plane-wave method is illustrated with two numerical examples, a laterally invariant and laterally variant medium. In two dimensions the downward propagating plane-wave focusing function $\tilde{f}_1^+(\mathbf{x}, \mathbf{p}_A, t)$ in equation 11, defines a plane-wave at the focal plane $\mathbb{D}_A$ with a dip angle α and $p_1 = \frac{\sin(\alpha)}{c}$. In the first numerical example we assume a medium with a laterally invariant velocity c for each depth and shown in Figure 30a and b. Figure 31 shows $\tilde{f}_1^+(\mathbf{x}_R, \mathbf{p}_A, t)$, for a focal plane at a depth of 900 m at two different angles: Figure 31a with an angle of 0 degrees and Figure 31c with an angle of 3 degrees. In Figure 31b and 31d the focus function is shown for receivers at the focal level (900 m depth). Similar to the focal-point Marchenko method, the medium for the plane-wave focusing functions is chosen homogeneous below the focal level. The focus function $\tilde{f}_1^+(\mathbf{x}, \mathbf{p}_A, t)$ has a focus in time at $t = \mathbf{p} \cdot \mathbf{x}_{H,A}$. For a dipping plane-wave this time focus occurs at different positions in the model. Figure 31a and c shows the computed $\tilde{f}_1^+(\mathbf{x}_R, \mathbf{p}_A, t)$ at the surface (0 m depth) and includes an extra event around -0.15 seconds that compensates for the multiples generated between the first and second reflector at 400 and 700 meter respectively. The compensation of multiples for point-sources is explained in de detail in Zhang and Slob (2019). The focus functions at focal level (Figure 31b and d) show only one event, the downgoing event present at the surface is compensated by the reflected event from the reflector at 700 meter depth.

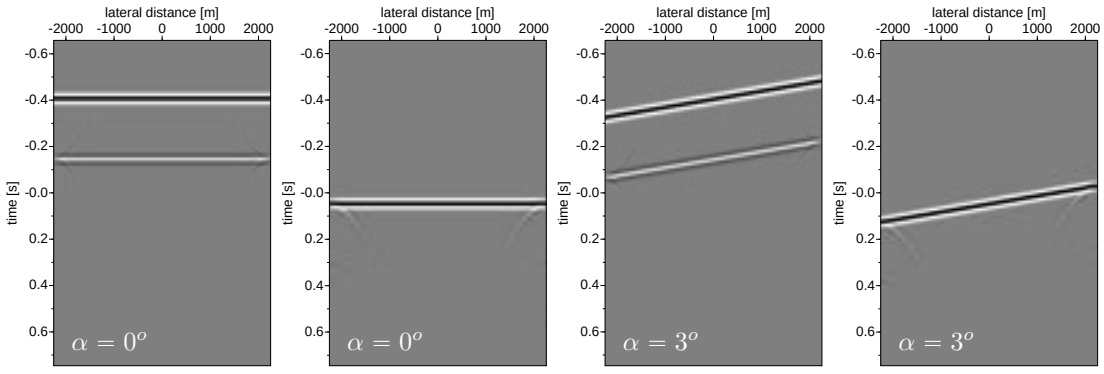


a) velocity model in m/s

b) density model in kg/m^3

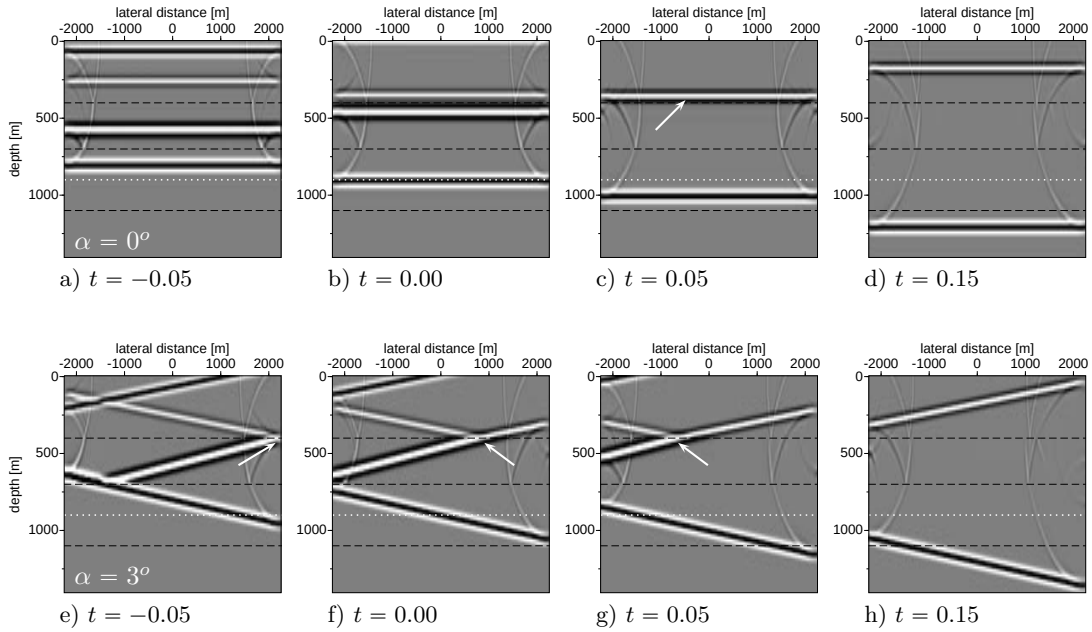
c) $R(\mathbf{x}_R, \mathbf{x}, t)$

Figure 30: Two dimensional four layer model with velocity (a) and density (b) parameters. A common-source record, with source position $\mathbf{x} = (x_1 = 0, x_3 = 0)$ and receivers at $\mathbf{x}_R = (x_1 = x_R, x_3 = 0)$ (c). The source wavelet in R has a flat frequency spectrum from 5 to 90 Hz.



a) depth $x_3 = 0$ m b) depth $x_3 = 900$ m c) depth $x_3 = 0$ m d) depth $x_3 = 900$ m

Figure 31: Time recordings of the plane-wave focusing function $\tilde{f}_1^+(\mathbf{x}_R, \mathbf{p}_A, t)$ with a focal depth of 900 meter measured with receivers at $x_3 = 0$ and $x_3 = 900$ meter in the truncated medium for two different plane-wave propagation angles (0 and 3 degrees). At the end-points of the plane-wave, diffraction curves are present due to the limited lateral extent of the constructed plane-wave.

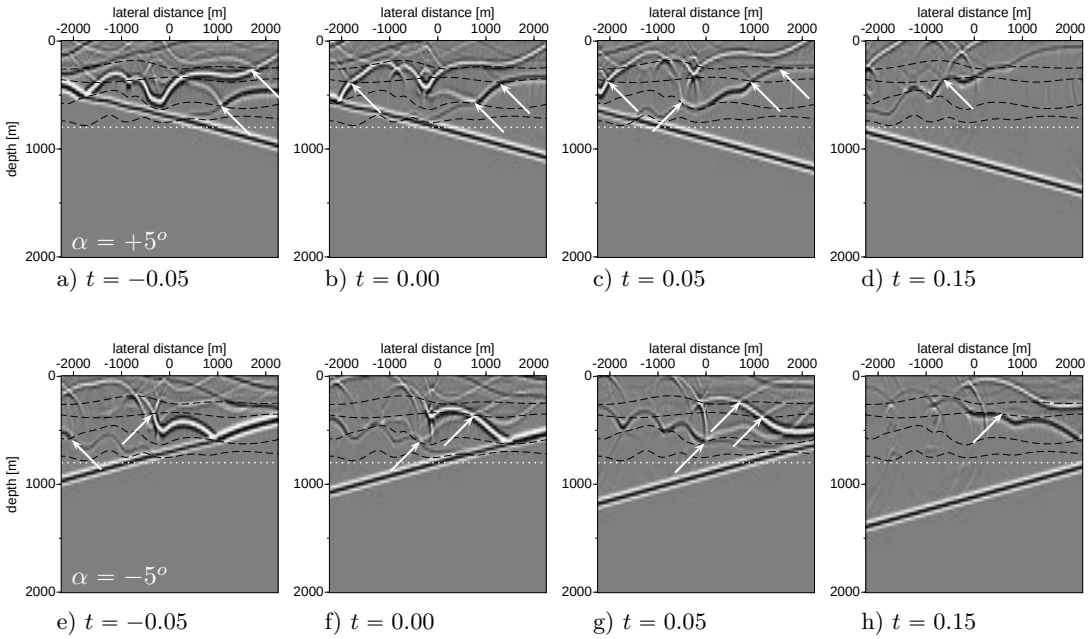


a) $t = -0.05$ b) $t = 0.00$ c) $t = 0.05$ d) $t = 0.15$

e) $t = -0.05$ f) $t = 0.00$ g) $t = 0.05$ h) $t = 0.15$

Figure 32: Time snapshots for a propagating $\tilde{f}_1^+(\mathbf{x}_R, \mathbf{p}_A, t) + \tilde{f}_1^-(\mathbf{x}_R, \mathbf{p}_A, t)$ at two different plane-wave propagation angles. Note the diffraction effects at the edges of the plane-wave. The white dotted-line indicates the focal depth of the plane-wave.

To illustrate this compensation effect, snapshots of the focusing function $\tilde{f}_1^+(\mathbf{x}, \mathbf{p}_A, t)$ (Figure 31a and 31c) propagating into the truncated medium (that is homogeneous below the last reflector at 700 m depth) are shown in Figure 32 for the same angles of 0 and 3 degrees. Figure 32a to 32d shows four different snapshots of the superposition of the down-going horizontal plane-wave $\tilde{f}_1^+(\mathbf{x}_R, \mathbf{p}_A, t)$ and upgoing plane-wave $\tilde{f}_1^-(\mathbf{x}_R, \mathbf{p}_A, t)$. The snapshots of a plane-wave with an angle of 3 degrees are shown in Figure 32e to 32h. At 0.05 seconds before t=0 (Figure 32a and 32e), there are two upward traveling reflected waves from the interfaces at 400 and 700 meter depth and two downgoing events from $\tilde{f}_1^+(\mathbf{x}_R, \mathbf{p}_A, t)$. The snapshots in Figure 32f (at $t = 0.00$ s), 32c and g (at $t = +0.05$ s) show that the second downward traveling event coincides at the first interface (at 400 m depth) with the upgoing reflection of the second interface (at 700 m depth) and these events compensate each other. This is indicated with an arrow in the pictures. The fourth snapshot show that after this compensation all the internal multiples between the reflectors at 400 and 700 m depth have vanished and only one downgoing direct wave and an upgoing reflected wavefield (from the reflector at 700 m depth) are remaining. The compensation of the first upward traveling multiple indicates that $\tilde{f}_1^+(\mathbf{x}_R, \mathbf{p}_A, t)$ is a solution of the Marchenko equations. The illustration in Figure 32 demonstrates that the internal multiple compensation principle also holds for the plane-wave Marchenko method.
In Figure 33 the experiment is repeated in the laterally variant medium of Figure 26. The focal-plane is chosen at 800 m depth, just below the fourth reflector. The same observation is made; the downgoing events in $\tilde{f}_1^+(\mathbf{x}_R, \mathbf{p}_A, t)$ compensate the upgoing events at interfaces and suppress the generation of internal multiples.



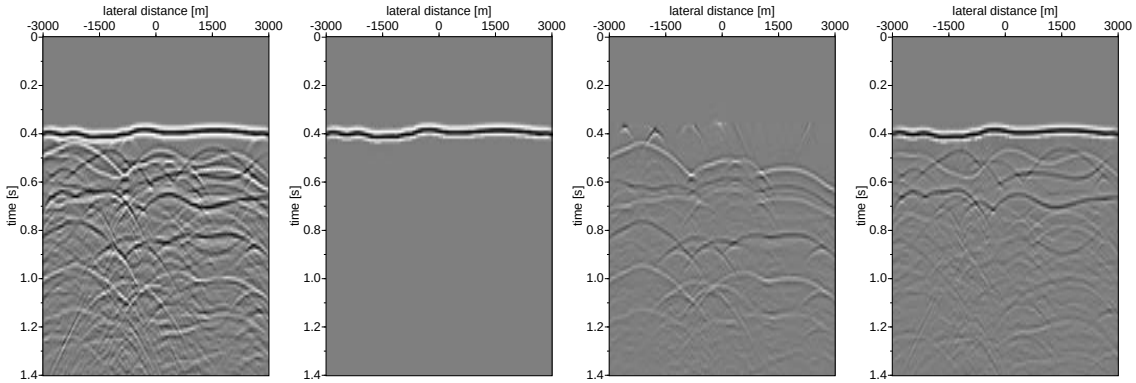
a) $t = -0.05$ b) $t = 0.00$ c) $t = 0.05$ d) $t = 0.15$

e) $t = -0.05$ f) $t = 0.00$ g) $t = 0.05$ h) $t = 0.15$

Figure 33: Time snapshots for a propagating $\tilde{f}_1^+(\mathbf{x}_R, \mathbf{p}_A, t) + \tilde{f}_1^-(\mathbf{x}_R, \mathbf{p}_A, t)$ in model of Figure 26 for two different plane-wave propagation angles. Note that there are remaining diffraction effects originating from edges on the interfaces. The white dotted-line indicates the focal depth of the plane-wave. The arrows indicate positions at a reflector where an up-going reflected field, that generates internal multiples, is compensated by a down-going event from the focusing function.

5.3.1 Horizontal plane-waves

To start the iterative Marchenko algorithm for plane-waves $\tilde{f}_{1,d}^+(\mathbf{x}_R, \mathbf{p}_A, t)$ is required; the first arrival of a plane-wave response from a focal-plane in the subsurface. This forward modeling step can be computed in a macro model estimated from the reflection data. The computed initial response is then muted below the first arrival times to get the time-reversal of the initial focusing field $\tilde{f}_{1,d}^+(\mathbf{x}_R, \mathbf{p}_A, t)$. A first example is made for a horizontal (zero-degree) plane-wave defined at 800 meter depth in the laterally varying model of Figure 26 (same model as in Meles et al., 2018).
Figure 34a shows the forward modeled plane-wave response of a horizontal plane-wave at 800 m depth and receivers at the surface. The first arrivals of that plane-wave, shown in Figure 34b, is the time-reversal of the input field $\tilde{f}_{1,d}^+(\mathbf{x}_R, \mathbf{p}_A, t)$ of the Marchenko scheme. The computed down- and up-going Green's functions, after 16 iterations of equations 15 and 16 and substituting the results in equations 13 and 14, are shown in 34c and 34d respectively and give the expected response. The horizontal plane-wave Marchenko mute-line (as defined by $\tilde{\Theta}'_{a,b}$ in equations 17 and 18), to separate the Green's function from the focusing functions, is symmetric around $t = 0$ because $\tilde{t}_d = \tilde{t}'_d$. This is the same time symmetry as in use for the Marchenko point-source algorithm.

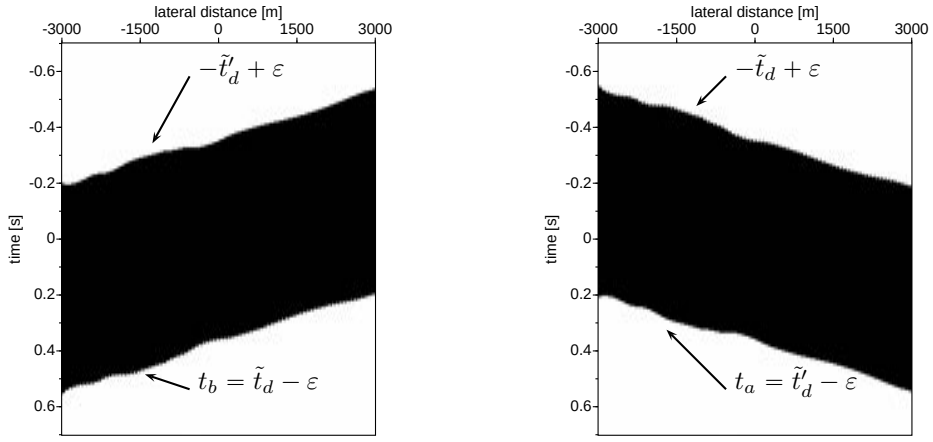


a) $\tilde{G}(\mathbf{x}_R, \mathbf{p}_A, t)$ b) $\tilde{f}^{+\star}_{1,d}(\mathbf{x}_R, \mathbf{p}_A, t)$ c) $\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}_A, t)$ d) $\tilde{G}^{-,-}(\mathbf{x}_R, \mathbf{p}'_A, t)$

Figure 34: Results of the plane-wave Marchenko scheme for a horizontal plane-wave $\mathbf{p}_A = (\mathbf{0}, x_{3,A})$. Adding the up- and downgoing Green's functions of c and d, that are computed with the Marchenko algorithm, gives the same wavefield as the directly forward modeled result in a. All figures are plotted with the same clipping factor.

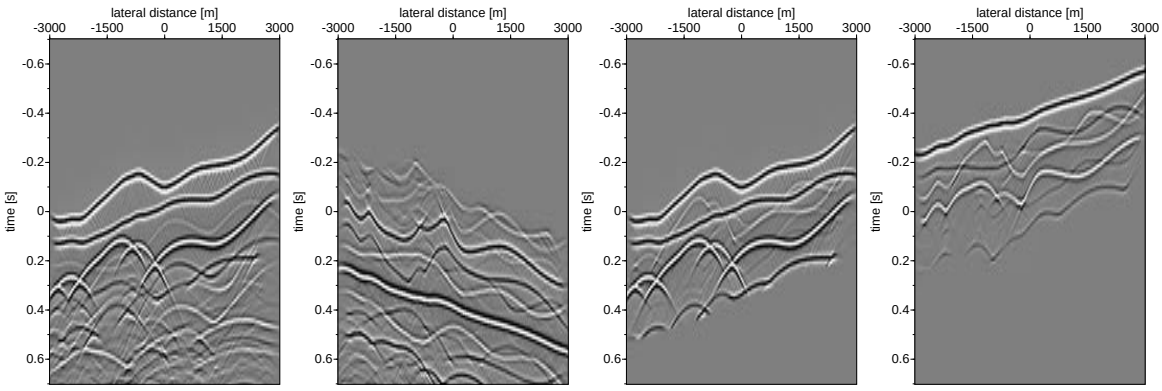
5.3.2 Dipping plane-waves

The Marchenko algorithm for dipping plane-waves follows the same procedure as for horizontal plane-waves. As indicated by equations 17 and 18 the Marchenko time windows have to be designed differently for dipping plane-waves. For horizontal plane-waves the implemented time window is symmetric around $t = 0$ and $\tilde{\Theta}'_a = \tilde{\Theta}'_b$. This does not hold anymore for dipping plane-waves. Figure 35 is an example of time windows that are designed for a dipping plane-wave of +5 degrees. Two time windows are designed: one for the even iterations (Figure 35a), in use by equation 15, and one for the odd iterations, that has a reverse angle (Figure 35b) and in use by equation 16. To design these windows for a positive angle, the first arrival time for a plane-wave at the same depth level with a reverse angle is needed as well.
In the following we explain in more detail what is expressed in the plane-wave Marchenko equations 15 and 16. The plane-wave Marchenko scheme starts with forward modeling the response to a (dipping) plane-wave at depth. This modeled wave-field response is given in Figure 37a, where the depth of the plane-wave is chosen at 800 m. The direct arrival is selected from this modeled field and shown in Figure 37b. The time reverse of this field will be $\tilde{f}^{+}_{1,d}(\mathbf{x}_R, \mathbf{p}_A, t)$. This field $\tilde{f}^{+}_{1,d}(\mathbf{x}_R, \mathbf{p}_A, t)$ is, together with the reflection data R, input of the plane-wave Marchenko scheme.



a) even iterations b) odd iterations

Figure 35: The time windows for dipping plane-waves for even (a) and odd (b) iterations for an angle of +5 degrees. The wavefields in the black area of the windows pass, the fields in the white area are set to zero.



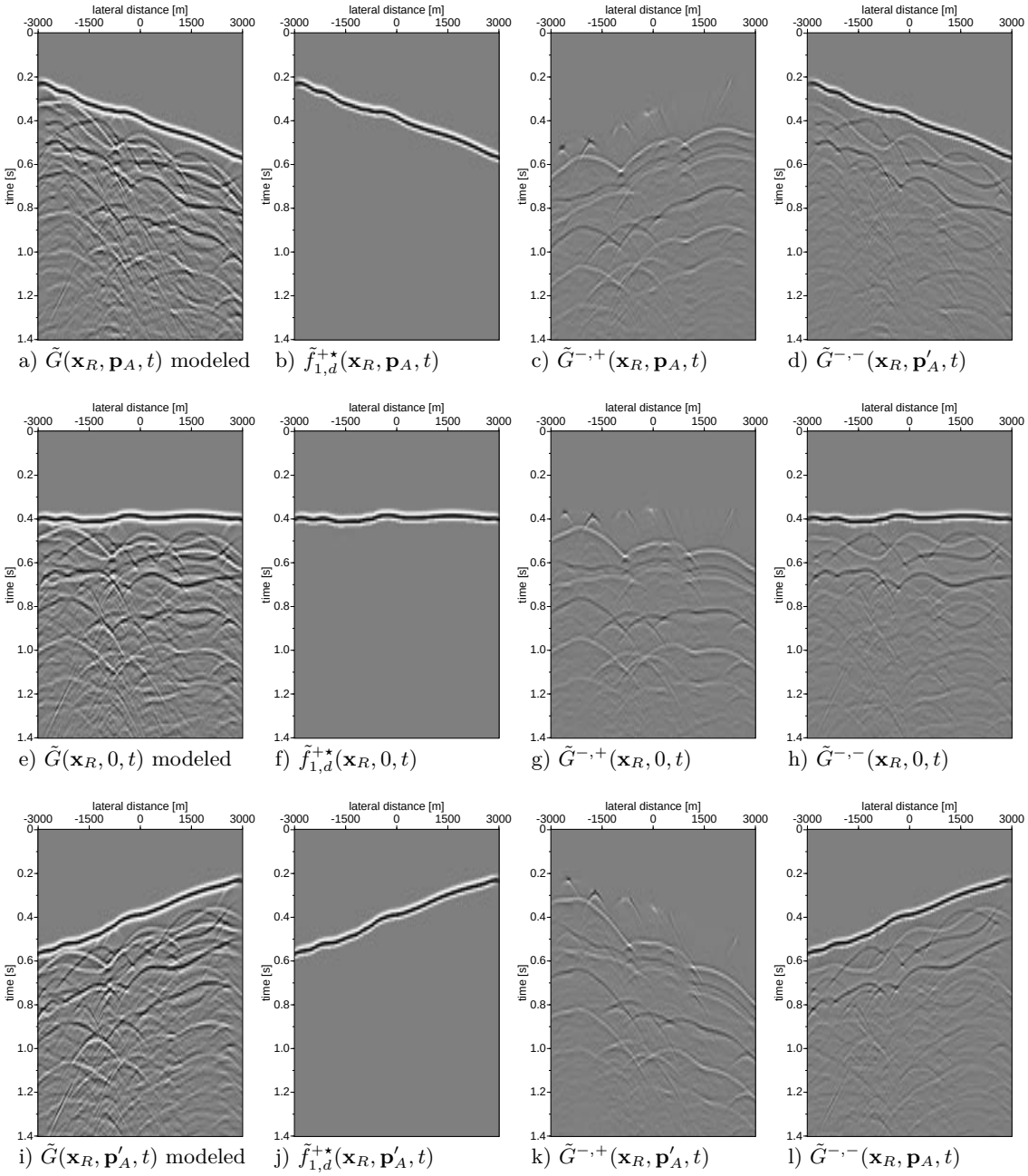
a) first iteration b) second iteration c) $\tilde{f}_1^-(\mathbf{x}_R, \mathbf{p}_A, t)$ d) $\tilde{f}_1^+(\mathbf{x}_R, \mathbf{p}_A, t)$

Figure 36: Basic plane-wave Marchenko results for a plane-wave with an angle of incidence of +5 degrees. Note, that the results of the first iteration (a) is dipping in the opposite direction as the second iteration (b) and the algorithm uses the time windows, designed for dipping plane-waves as given in equations 17 and 18, to take this into account.

In the Marchenko algorithm the iterations are alternating between a convolution of the focusing functions with R, or a correlation with R (Thorbecke et al., 2017). In the first step, correlation by equation 16, the wavefield is shifted backward in time related to the times of $\tilde{t}_d$, and in the second step, convolution by equation 15, the wavefield is shifted forward in time related to the times of $\tilde{t}'_d$. In Figure 36a the result of the first iteration (correlation) is shown and in Figure 36b the result of the second iteration (convolution) is shown. In Figure 36a we can see that the first event, that starts at negative time, shows an imprint of the undulation of the first reflector and has an opposite dip compared to the plane-wave in Figure 37a.
The result of the first iteration in Figure 36a is windowed in time (with the window in Figure 35a) to mute $\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}_A, t)$, followed by time-reversal and convolution with R (equation 16). The result is shown in Figure 36b. In this second iteration the convolution step brought the data back to the arrival times corresponding to reflection times in the forward modeled plane-wave response of Figure 37a. Note that the arrival time in Figure 36b, starting from the left at time 0.2 s dipping to the right to 0.5 s, is the same as the first arrival time $\tilde{t}_d$ in Figure 37a and 37b. The result of this second iteration is muted in time (with the window in Figure 35b) to remove $\tilde{G}^{-,-}(\mathbf{x}_R, \mathbf{p}'_A, t)$ and the first arrival event at $\tilde{f}^{+}_{1,d}(\mathbf{x}_R, \mathbf{p}_A, t)$, as indicated in Figure 29b. The odd iteration(s) are building up $\tilde{f}^{-}_{1}(\mathbf{x}_R, \mathbf{p}_A, t)$ and $\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}_A, t)$ and the even iteration(s) are building up $\tilde{f}^{+}_{1}(\mathbf{x}_R, \mathbf{p}_A, t)$ and $\tilde{G}^{-,-}(\mathbf{x}_R, \mathbf{p}'_A, t)$. Figure 36c and 36d show $\tilde{f}^{+}_{1}(\mathbf{x}_R, \mathbf{p}_A, t)$ and $\tilde{f}^{-}_{1}(\mathbf{x}_R, \mathbf{p}_A, t)$ respectively after 16 iterations.
In Figure 37 three plane-wave responses are shown with angles of -5, 0 and 5 degrees after 16 iterations. Comparing these three plane-wave responses for $\tilde{G}^{-,-}$ and $\tilde{G}^{-,+}$ shows that each angle illuminates different parts of the medium. This is clearly seen in the events that arrive later than 1.2 seconds. By combining different plane-wave responses into one image a fully illuminated image can be constructed by using only a few migrations (Meles et al., 2018). Plane-wave imaging, that suppress internal multiples, can use the same strategies as point-source Marchenko, for example double focusing as described in van der Neut et al. (2017); Staring et al. (2018), or Multi Dimensional Deconvolution as discussed in Ravasi et al. (2016). Almobarak (2021) discusses different plane-wave imaging methods and shows that the Marchenko Green's function plane-wave response is a computational efficient imaging method.
Figure 38 shows horizontal plane-wave images from the Troll field data-set located west of Norway, that was kindly provided by Equinor. This data set is part of a time-lapse monitoring set. We have selected a small part of this data-set, with source and receiver positions on the same locations. The receiver and source spacing is 12.5 meter and traces are recorded with a time sampling of 4 ms. The data-set has been pre-processed to remove free-surface multiples and deconvolve the wavelet (Qu and Verschuur, 2020).
The imaging is carried out according to the imaging method described in Meles et al. (2018). In the basic imaging method, a forward modeled plane-wave response is computed at each depth level in an estimated smooth macro model of the data. This plane-wave depth response is correlated with the point-source responses of the recorded data and integrated over the receiver coordinate for each point-source response. This creates the plane-wave depth response of the data (Rietveld et al., 1992). This plane-wave response of the data is correlated with the same modeled plane-wave response at each depth level and the imaging condition $t = 0$ is used to construct the image for all depth levels. Combining these depth levels gives the left side picture in Figure 38 labeled 'standard'.
To compute the Marchenko based image, the first arrivals of the forward modeled plane-wave response at each depth level is input to the plane-wave Marchenko algorithm to create $\tilde{G}^{-,+}$. The computed $\tilde{G}^{-,+}$ is, similar to the standard imaging method, correlated with the forward modeled plane-wave response for each depth level, and the imaging condition at $t = 0$ constructs the image for all depth levels. For a horizontal plane-wave ($\mathbf{p}_A = (\mathbf{p} = \mathbf{0}, x_{3,A})$) at one depth level $x_{3,A}$ this imaging condition is represented by

$$\tilde{I}(\mathbf{x}_R, \mathbf{p}_A) = \int_t \tilde{f}^{+}_{1,d}(\mathbf{x}_R, \mathbf{p}_A, t)\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}_A, t)dt. \tag{19}$$

The advantage of using $\tilde{G}^{-,+}$ is that this field does not contain downgoing internal multiples from the layers above the focal/imaging depth. The Marchenko method has separated the internal multiples in up- and downgoing parts in the Greens functions by applying the computed focal functions to the reflection data. Alternative strategies to compute an image without internal multiples can be based on MME. Thorbecke et al. (2021) illustrate the working of the internal multiple elimination of MME directly on reflection data in Figure 9 and equations (12)-(20). Staring et al. (2018) illustrate source-receiver Marchenko redatuming and imaging on field data using an adaptive double-focusing method. Figure 4 of that paper shows internal multiples that are first predicted and then subtracted from the data.

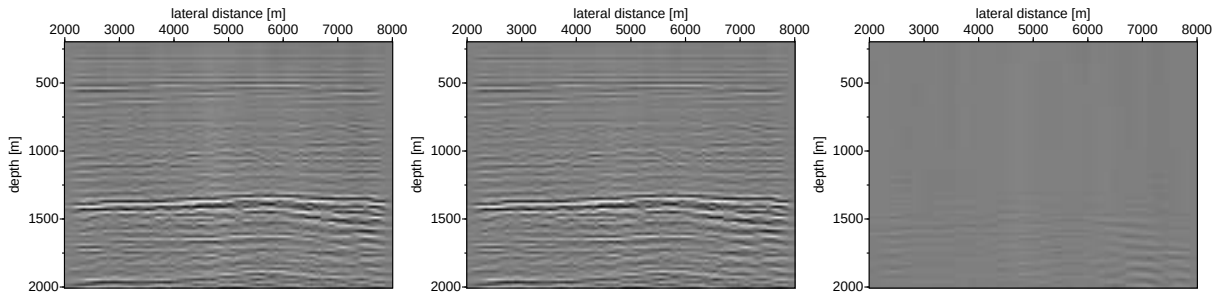


a) $\tilde{G}(\mathbf{x}_R, \mathbf{p}_A, t)$ modeled b) $\tilde{f}^{+\star}_{1,d}(\mathbf{x}_R, \mathbf{p}_A, t)$ c) $\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}_A, t)$ d) $\tilde{G}^{-,-}(\mathbf{x}_R, \mathbf{p}'_A, t)$

e) $\tilde{G}(\mathbf{x}_R, 0, t)$ modeled f) $\tilde{f}^{+\star}_{1,d}(\mathbf{x}_R, 0, t)$ g) $\tilde{G}^{-,+}(\mathbf{x}_R, 0, t)$ h) $\tilde{G}^{-,-}(\mathbf{x}_R, 0, t)$

i) $\tilde{G}(\mathbf{x}_R, \mathbf{p}'_A, t)$ modeled j) $\tilde{f}^{+\star}_{1,d}(\mathbf{x}_R, \mathbf{p}'_A, t)$ k) $\tilde{G}^{-,+}(\mathbf{x}_R, \mathbf{p}'_A, t)$ l) $\tilde{G}^{-,-}(\mathbf{x}_R, \mathbf{p}_A, t)$

Figure 37: Marchenko computed plane-wave responses for angles at 5 (a-d), 0 (e-h), and -5 (i-l) degrees. Note the difference in illumination in the decomposed Green's function for different dipping angles of the plane-wave. Analog to Figure 34 the addition of the Marchenko computed up- (d) and downgoing (k) Green's function gives the forward modeled response in a.



a) Standard at 0 degrees b) Marchenko at 0 degrees c) difference

Figure 38: Plane-wave images of the Troll field data-set for a horizontal plane-wave using standard imaging (left) and Marchenko based imaging (middle). All images are displayed with the same clipping factor.

The middle picture of Figure 38 shows the Marchenko-created image. The difference between the standard image and the Marchenko-based image is shown in Figure 38c. From this difference plot it is observed that the Marchenko method predicts and removes internal multiples. In this data-set the effect of the multiple removal on the image is small. van IJsseldijk et al. (2023) show that Marchenko multiple removal on this dataset improves the confidence of the effects of small time-lapse changes.

5.4 Conclusions

The plane-wave Marchenko method is a straightforward extension of the point-source Marchenko method. A counter-intuitive aspect of the plane-wave method is that the retrieved up- and down-going Marchenko Green's functions have opposite dipping angles. This is taken into account in the time windows that separate the Green's function from the focusing function. In case one would like to get the total Green's function for a specific dip angle, one would have to run the whole procedure twice, for opposite dip angles. In this paper, the use of these time windows is illustrated with numerical and field data examples. The plane-wave Marchenko method can give a computational advantage over the point-source method. Specially for imaging applications with 3-dimensional datasets that have moderate lateral changes, in that case only a few plane-wave migrations are needed to compute a well illuminated image.

Acknowledgment

The authors thank Equinor (formerly Statoil A.S.) for providing the marine dataset of the Troll Field. This research was funded by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement no. 742703).

5.5 Code availability

- Name of the code/library: OpenSource code for Finite Difference, Marchenko algorithms and processing utilities
- Contact: j.w.thorbecke@tudelft.nl
- Hardware requirements: tested on x86_64 and aarch64 processors
- Program language: C and Fortran
- Software required: C compiler, Fortran compiler, GNU Make, tested only on Linux/Unix OS. The display and generation of the figures is done with Seismic Unix and is available at https://github.com/JohnWStockwellJ
- Program size: 147 MB

The source codes are available for downloading at the link: https://gitlab.com/geophysicsdelft/OpenSource.git
The scripts to reproduce the results in this manuscript can be found in `.../OpenSource/marchenko/demo/planewave`.
The README in that directory explains all the steps to reproduce the results in the manuscript. For the reproduction of the measured data example please contact us directly, we will ask the owner of the data if we can share the data.

A Appendix A

A.1 Plane-waves

To model a plane-wave with a tilted angle in finite difference Thorbecke and Draganov (2011) a ray-parameter value p [s/m] is defined by a chosen velocity and a plane-wave angle positioned at a horizontal depth-level. The plane-wave is triggered at all grid-points that are positioned at the same horizontal depth-level in the finite-difference grid. To simulate a dipping plane-wave each source position that defines the dipping plane-wave gets a different time-delay t_p (=p*distance) that depends linearly on the distance from the rotation point. The rotation point of the plane-wave, that defines the source at $t = 0$, is chosen at the central position ($\mathbf{x}_c$) of the plane-wave.

$$\mathbf{p} = sin(\alpha)/c_p, \tag{20}$$

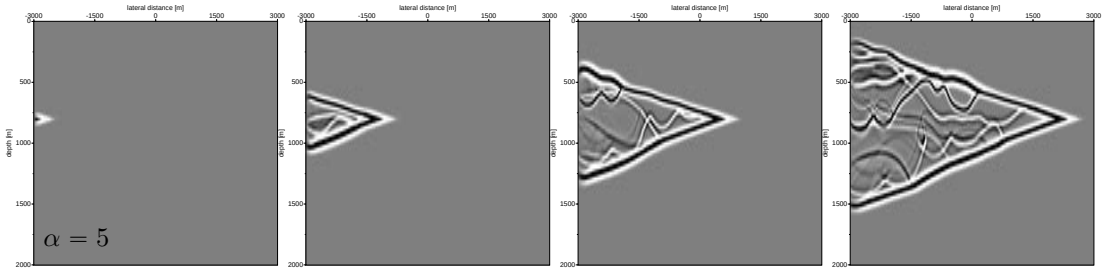
$$\mathbf{x}_p = (\mathbf{x} - \mathbf{x}_c) * dx, \tag{21}$$

$$t_p(\mathbf{x}) = \mathbf{x}_p * \mathbf{p}. \tag{22}$$

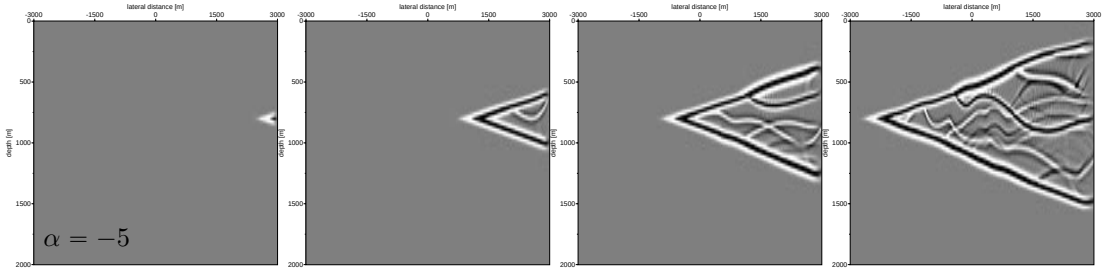
where α is the dip angle, c_p is the propagation velocity of the medium, and $\mathbf{x_c} = (x_{1,c}, x_{2,c})$ are the horizontal coordinates of the central location of the plane-wave source. By including these coordinates in the definition of $\mathbf{x}_p$, we ensure that the center of the plane-wave source always has an emission time of $t = 0$ Brackenhoff et al. (2022).
In a homogenous model a plane-wave, modeled with these time-delays on a horizontal level, is equal to a plane-wave on a slanted line with `src_angle(`$= \alpha$`)` in a medium with velocity `src_velo(`$= c_p$`)`. In Figure 39a to 39d four snapshots are shown of a propagating plane-wave with an angle of +5 degrees and a propagation velocity of 1500 m/s. The snapshots of the +5 degrees dipping plane-wave, between x-position -3000 and 0, have a negative time delay, the shot in the middle ($x = 0$) starts at $t = 0$. The shot that is initiated first (at negative time) is positioned at the first lateral position (most left position at -3000 m in Figure 39a) and has a time-delay of -0.1742 s. To account for the negative times that occur in the modeling of dipping plane-waves, the finite difference modeling code Thorbecke and Draganov (2011) is adapted. To model a negative angle the shot that is initiated first is positioned at the last spatial sample in Figure (most right position at +3000 m in Figure 39e to 39h) with the same time-delay of -0.1742 s.
An alternative implementation of a dipping plane-wave is achieved by placing each source at a different grid-point that also varies in depth. All sources are activated at the same time instant $t = 0$ without any time-delays between the sources. Figure 39i to 39l shows the snapshots of this implementation. The disadvantage of this implementation is that in solving the Marchenko equations for a plane-wave with a defined angle α in depth, the ray-parameter(s) of $\tilde{G}^{-,-}(\mathbf{p}_A)$ are not related anymore to $\tilde{G}^{-,+}(-\mathbf{p}_A)$, since $\mathbf{p}$ is position dependent.

A.2 Time-shifts in Marchenko equations

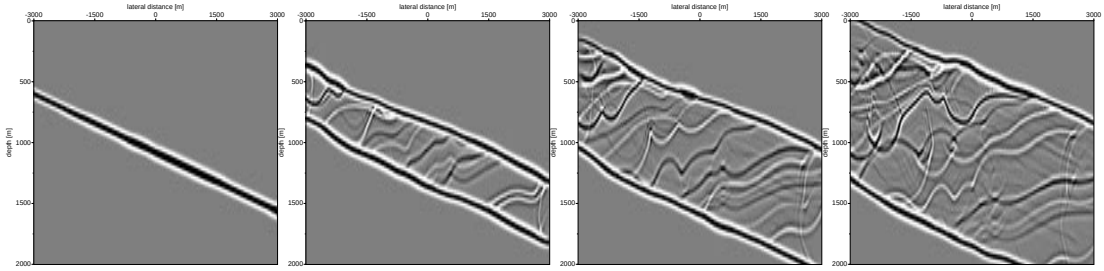
In the use of tilted plane-waves time-shifted sources play an important role and it is worthwhile to understand the effects of a time-shift in the regular Marchenko scheme with point sources. In Figure 40a to 40e the standard Marchenko results are shown for the model of Figure 30 with a focal-point at 900 meter depth. In Figure 40f the forward modeled operator is shifted +0.3 seconds forward in time, hence the time-reverse of that forward modeled operator, $f^+_{1,d}$ is shifted -0.3 seconds backward in time. The Marchenko results in Figure 40g to 40j show that the solution of the Marchenko equation does not change; the same fields are computed, but shifted in time. The fields f^-_1, f^+_1 and G^- are shifted backward in time and G^+ is shifted forward in time.
To compute the time-shifted Marchenko results the time-windows Θ that are in use in the Marchenko equations, to separate the focal- from the Green's function, must be adjusted accordingly. This means that for even and odd iterations different time-windows have to be designed to take into account the constant-time shift of 0.3 seconds.
The results of the time-shifted Marchenko can of course also be used for further processing, but one has to be careful. For example $G^{-,+}$ can be back-propagated into the medium to build an image at focal depth, but before back-propagating $G^{-,+}$ into the medium the time-shifted response has to be shifted back to the original $t = 0$ of $f^+_{1,d}$, otherwise the $G^{-,+}$ will not focus at the focal depth of $f^+_{1,d}$.



a) $t = -0.174$ b) $t = -0.074$ c) $t = 0.026$ d) $t = 0.126$



e) $t = -0.174$ f) $t = -0.074$ g) $t = 0.026$ h) $t = 0.126$



i) $t = 0.1$ j) $t = 0.2$ k) $t = 0.3$ l) $t = 0.4$

Figure 39: Time snapshots for different implementations to model a tilted plane-wave. Pictures a-d show the time-delayed implementation at time instances -0.174, -0.074, 0.026 and 0.126 seconds for an angle of +5 degrees. Pictures e-h show the time-delayed implementation at the same time instances for an angle of -5 degrees. Pictures i to l show a titled grid position implementation at different snapshot times.

A.3 Time wrap-around

The time wrap-around events only occur for deep focusing levels and are not observed in the Figure 29 in the paper that are chosen not that deep. To illustrate the wrap-around events a similar picture as Figure 29 is shown in Figure 41 with a focal level at 1900 m. depth. The left picture in Figure 41 (a) shows time wrap-around for times earlier than $-\tilde{t}_d + \varepsilon$. After padding with 2x the number of time samples (from 1024 to 2048 time samples) the time wrap-around is not visible anymore in picture (b).

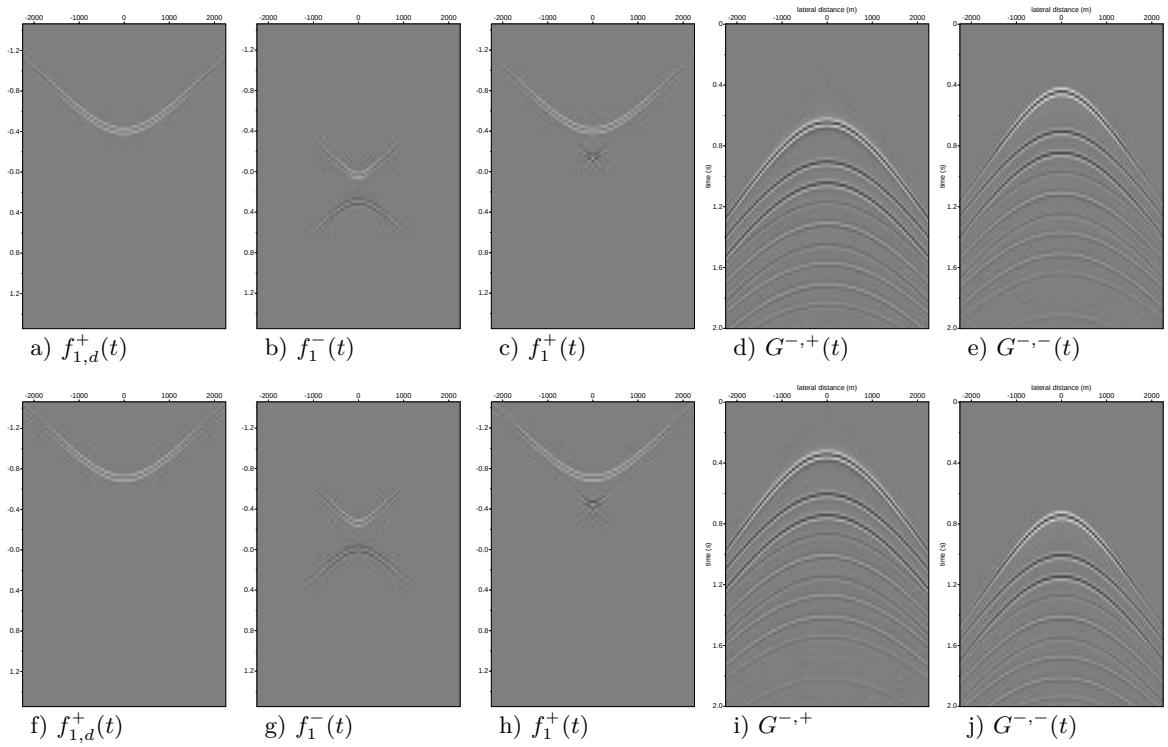
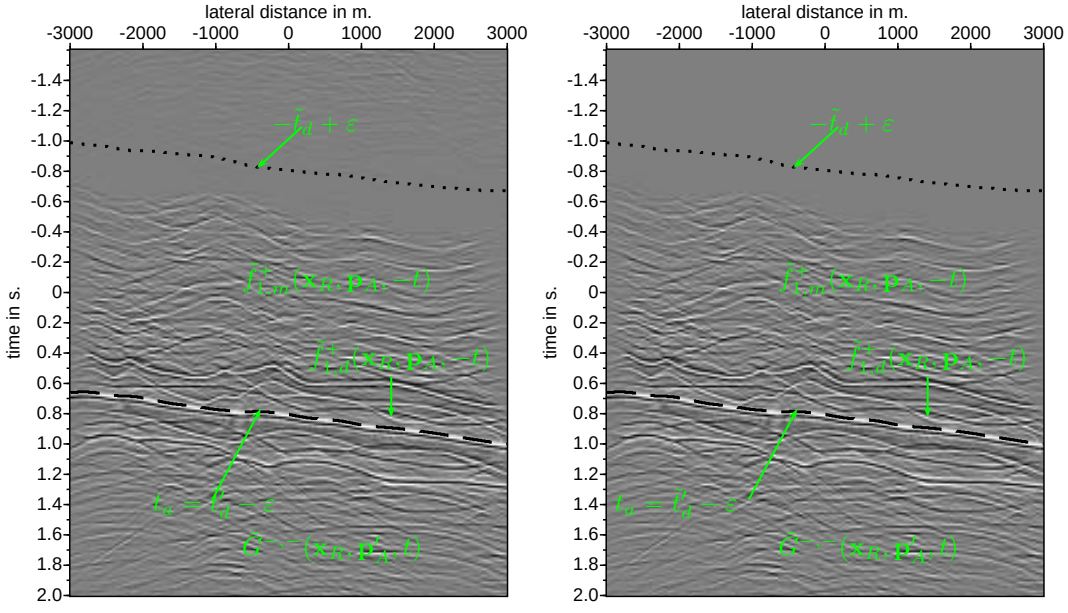


Figure 40: Standard and time-shifted Marchenko results for a focal-point at z=900 in the 4 layer 1D model of Figure 30. The applied shift is +0.3 s. forward in time on the forward modeled field $(f^+_{1,d})^*$.



a) Separation without time-padding.

b) Separation with time-padding.

Figure 41: Illustration of the time-window function to the plane-wave Green's function $\tilde{G}^{-,-}(\mathbf{p}'_A)$ from the focusing function $\tilde{f}^{+\star}_1(\mathbf{p}_A)$. The dashed black lines indicate the separation line of the time-window and are indicated with white arrows. The dotted black line indicates the time-window that suppresses time wrap-around.

A.4 Scripts to reproduce the figures in this chapter.

```
--------------------------
The instructions below explain how to reproduce the results in the paper:

"Design, implementation and application of the Marchenko Plane-wave algorithm"

by Jan Thorbecke, Mohammed Almobarak, Johno van IJsseldijk, Joeri Brackenhoff, Giovanni Meles, Kees Wa
--------------------------

#Figure 1: model lateral varying created by Giovanni Meles
cd marchenko/demo/planewave/twoD
# this creates the gridded models
./model.scr
# to create the eps figures
./epsModel.scr model

#Figure 2: time-window seperation point-source algorithm
cd marchenko/demo/planewave/twoD
#First generate all 601 shots to model the reflection data R from the model of Figure 1
./shots.scr #this script is based on slurm
#model direct field
./direct.scr
#subtract direct field from modeled shots to create Reflection data in shots/refl_cp.su
./remove_direct.scr

#Figure 2
# initial modeling for first arrival
./IniPoint.scr
#run the marchenko algorithm for a point in the middle at 800 m depth and create figure 2
./sumGandFpoint.scr

#Figure 4 same as figure 2 but now for plane-waves
#first model the first arrivals of the plane-wave response
./IniFocus.scr
#run plane-wave marchenko with plane-wave of 5 degrees and generate picture
sumGandF.scr

#Figure 5 (1D model + middle shot)
cd marchenko/demo/planewave/oneD
./model.scr
./epsModel.scr

#Figure 6 & 7 : plane wave snapshots angle 0 and 3 in 1D medium
cd marchenko/demo/planewave/oneD
./initialFocusPlane.scr
./marchenkoPlane.scr
./backpropf1plusPlane.scr
./epsPlane.scr plane
./epsPlane.scr snapshots

#Figure 8 (see figure 1 for generating data)
cd marchenko/demo/planewave/twoD
./model.scr
./IniFocus.scr (for 0 5 and -5)
./marchenkoPlaneIter.scr (for 0 5 and -5)
#back propagating snapshots through Giovanni's Model
```

```
./backpropf1plusPlane.scr
./epsPlane.scr snapshots

#Figure 9
./epsPlane.scr shots0

#Figure 10: use fmute with returnmask=1
cd marchenko/demo/planewave/twoD
./muteW.scr
./epsModel.scr mute

#Figure 11
cd marchenko/demo/planewave/twoD
./epsPlane.scr shots5
./epsPlane.scr iter

#Figure 12
cd marchenko/demo/planewave/twoD
./epsPlane.scr shots5
./epsPlane.scr shots0
./epsPlane.scr shots-5


#Figure 13 Troll data field
Not provided have to ask permission for sharing data to Equinor
Data stored at /palmyra/data/jthorbecke/Johno/data

#Imaging bash script
iangle=0
nangle=1
startangle=0
while (( iangle < nangle ))
do
  (( angle = ${startangle} + ${iangle}*${dangle} ))
  echo angle=$angle
  filegm=gmplanes$angle.su
  filefd=fdplanes$angle.su
  invangle=$((angle * -1))

  # data with internal multiples
  syn2d file_syn=fdplanes$angle.su file_shot=$Rdata nshots=481 file_cfp=Rfocus$angle.su fmax=110 verbo
  fconv file_in1=fdplanes$invangle.su file_in2=Rfocus$angle.su file_out=Rimage$angle.su fmax=110 verbo
  # scale with 5e-5 * dt (0.004) = 2e-7
  sugain < Rimage$angle.su scale=2e-7 > nep.su
  mv nep.su Rimage$angle.su
  suwind < Rimage$angle.su itmin=0 itmax=0 | sustrip > Rimage366x481_$angle.bin

  # data after Marchenko
  fconv file_in1=fdplanes$invangle.su file_in2=gmplanes$angle.su file_out=Gimage$angle.su fmax=110 ver
  suwind < Gimage$angle.su itmin=0 itmax=0 | sustrip > Gimage366x481_$angle.bin
  (( iangle = $iangle + 1))
done

#generate eps files
for angle in -3 0 3
do
for file in Rimage366x481_$angle.su Gimage366x481_$angle.su GRdiff366x481_$angle.su
```

```
do
    file_base=${file%.su}
    supsimage < $file hbox=3 wbox=4 labelsize=12 linewidth=0.0 \
          n1tic=2 x1beg=0 x1end=2005 f1=200 f1num=0 d1=5 d1num=500 \
          label1="depth [m]" label2="lateral distance [m]" \
          f2=2000 f2num=2000 d2num=1000 d2=12.5 clip=1e8 > $file_base.eps
done
done

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

Description of files:

1) ./model.scr computes the gridded velocity/density model

2) ./shots.scr computes 601 shots using slurm arrays: ~100 s runtime for each shot

3) ./direct.scr compute 1 shot that contains direct wave only

4) ./remove_direct.scr removes the direct wave from the data created in 3 and places all shots in one

   Figures of the model and the middle shot are generated by
   ./epsModel.scr model

5) ./IniFocus.scr compute plane wave responses at 800 m depth with angles -5 0 and 5 degrees.

   To compute the plane wave respone with the sources all starting at t=0 on a slanted line in the mod

   ./IniFocus.scr
```

6 3D Marchenko algorithms

7 Multi Dimensional Deconvolution

1 Introduction

2 Parameters in program MDD

3 Examples to run the code

A Source and directory structure

```
.
├── Common_Public_License.txt
├── SU_LEGAL_STATEMENT.txt
├── CODE_OF_CONDUCT.md
├── README.md
├── INSTALL ..... Short instructions to install the code package
├── REPRODUCE ..... Summary how to reproduce the examples in the papers mentioned
├── Make_include_template ..... Template to create your own Make_include
├── Make_include ... File with system specific setting and can be adapted for specific Unix systems
├── Makefile ..... Controls the compilation and linking of the programs in the subdirectories
├── FFTlib ..... Library for FFT transformation routines used by the programs
├── doc ..... where you can find this manual
├── include ..... Directory for the include file from the FFT library
├── lib ..... Directory where the FFT library is placed
├── bin ..... Directory for the binaries compiled and linked using the Makefile
├── fdelmodc ..... This directory contains all source code for the program fdelmodc
│   ├── FiguresPaper . The bash-script to generate the Figures from in Geophysics paper Thorbecke
│   │   and Draganov (2011)
│   └── demo ..... Bash-script which demonstrate the possibilities of fdelmodc
├── fdemmodc
├── extrap
├── extrap3d
├── marchenko
├── marchenko3D
├── raytime3d
├── fdelmodc3D
├── fdacrtmc
├── zfp
├── utils Source code for programs to generate models, wavelets, basic processing can be found here
├── raytime
├── marchenko_applications
├── corrvir
├── MDD
├── MatInv
├── 3DFD
├── movies
└── scripts

marchenko
├── writeDataIter.c
├── par.h
├── segy.h
├── fmute.c
├── readTinvData.c
├── marchenko_primaries.c
├── readShotData.c
└── applyMute.c
```

```
├── marchenko.c
├── applyMute_tshift.c
├── marchenko
├── synthesis.c
├── marchenko_tshift
├── marchenkojan.c
├── findFirstBreak.c
├── writeData.c
├── verbosepkg.c
├── docpkge.c
├── synthesis_cgemm.c
├── getpars.c
├── getFileInfo.c
├── wallclock_time.c
├── readData.c
├── atopkge.c
├── Makefile
├── marchenko_primaries
├── fmute
├── marchenko_tshift.c
└── name_ext.c
demo
    ├── README
    └── oneD
        ├── epsMarchenkoIter.scr
        ├── figAppendix.scr
        ├── conv.gnp
        ├── p5all.scr
        ├── referenceShot.scr
        ├── model.scr
        ├── primariesFrame.scr
        ├── epsPrimaries.scr
        ├── README
        ├── primariesPlane.scr
        ├── primaries.scr
        ├── marchenkoPlaneReg.scr
        ├── initialFocus1300.scr
        ├── iterations.scr
        ├── marchenkodt.scr
        ├── epsModel.scr
        ├── backProp_f2sum_movie.scr
        ├── line3
        ├── initialFocus.scr
        ├── epsBackprop.scr
        ├── epsIterwithLabels.scr
        ├── clean
        ├── marchenkoIter.scr
        ├── marchenkoPlane.scr
        ├── primariesFocus.scr
        ├── initialFocusPlane.scr
        ├── test.scr
        ├── migr.scr
        ├── epsCompare.scr
        ├── marchenko.scr
        ├── f2Plreg.su
        ├── model2.scr
        └── backpropf2.scr
```

```
├── windowA60W10.txt
├── epsWindows.scr
├── primaries_skiptest.scr
├── 3D
│   ├── marchenko.scr
│   └── marchenkoIter.scr
├── twoD
│   ├── check.scr
│   ├── clean
│   ├── direct.scr
│   ├── initialFocus_slurm.scr
│   ├── model.scr
│   ├── remove_direct.scr
│   ├── shots_pbs.scr
│   ├── shots_slurm.scr
│   ├── backpropf2.scr
│   ├── backProp_makemovie.scr
│   ├── eps.scr
│   ├── marchenko.scr
│   ├── referenceShot.scr
│   ├── homgview.scr
│   ├── marchenko_ray.pbs
│   ├── marchenko_ray.scr
│   ├── initialFocus_pbs.scr
│   ├── epsPrimaries.scr
│   ├── homg_reference.scr
│   ├── initialPlane.scr
│   ├── primaries.scr
│   ├── README
│   ├── initialFocus.scr
│   ├── epsPlane.scr
│   ├── homgpng.scr
│   ├── marchenkoPlane.scr
│   └── rayvsp.scr
├── ScientificReports
│   ├── README
│   ├── back_injrate_planes.scr
│   ├── backpropf2.scr
│   ├── check.scr
│   ├── clean
│   ├── direct.scr
│   ├── epsBack.scr
│   ├── initialFocus.scr
│   ├── marchenko.scr
│   ├── model.scr
│   ├── remove_direct.scr
│   ├── shotslurm.scr
│   └── NatureSnapshots.tex
├── primaries
│   ├── marchenkojan.scr
│   ├── marchenko.scr
│   ├── marchenko_invisible.scr
│   └── marchenkoGiovanni.scr
└── invisible
    ├── marchenko.scr
    ├── clean
    └── p4all.scr
```

```
        ├── README
        ├── model.scr
        ├── eps.scr
        └── primaries.scr
    ├── WS15
        ├── job.pbs
        ├── README.1
        ├── README.2
        ├── README.3
        ├── README.4
        ├── README.5
        ├── setup.sh
        └── MarchenkoWorkshop.pdf
    ├── mme
        ├── epsPrimaries.scr
        ├── iterations.scr
        ├── model.scr
        ├── primariesPlane.scr
        ├── primaries.scr
        ├── README_PRIMARIES
        ├── epsModel.scr
        ├── clean
        ├── primariesTestuv.scr
        └── strongContrast
            ├── epsPrimaries.scr
            ├── model.scr
            └── iterations.scr
└── Papers
    ├── MelesGJI2018.pdf
    ├── ThorbeckeGPY2017.pdf
    ├── WapenaarSR2018.pdf
    └── ZhangGPY2019.pdf

/
├── bin/
    ├── basop ..... Executable for basic operations (shift, envelope, ..) on seismic data
    ├── extendModel Executable to extends the edges of a file with first and last trace and/or sample
    ├── fconv ..... Executable for auto-, cross-correlation, deconvolution or convolution computation
    ├── fdelmodc ..... Executable for elastic acoustic finite-difference wavefield modeling
    ├── green ..... Executable for the calculation of 2D Greens function in homogeneous media
    ├── makemod ..... Executable for building gridded subsurface models
    └── makewave ..... Executable to generate wavelets
├── /include
    └── genfft.h ..... Include file for the FFT library
├── /lib
    └── libgenfft.a ..... Library which contains the objects of the FFT routines

/
└── fdelmodc/
    ├── Makefile ..... controls the compilation and linking of the program fdelmodc
    ├── fdelmodc.h ..... header file which defines structures used modeling
    ├── par.h ..... header file from SU for reading in program parameters
    ├── SUsegy.h ..... adjusted segy header file, which defines ns as an integer
    ├── segy.h ..... original segy header from SU
    ├── acoustic2.c ..... Kernel of acoustic FD using 2'nd order derivatives
    ├── acoustic4.c ..... Kernel of acoustic FD using 4'th order derivatives
    ├── acoustic6.c ..... Kernel of acoustic FD using 6'th order derivatives
    └── applySource.c ..... Routine which adds source amplitude(s) to the wavefield grids
```

- atopkge.c ..... converts ascii to arithmetic from SU
- CMWC4096.c ..... random number generator
- defineSource.c ..... computes, or read from file, the source signature
- docpkge.c .....function for self-documentation, from SU
- elastic4.c ..... Kernel of elastic FD using 4'th order derivatives
- fdelmodc.c .....main FD modeling program, contains self-doc
- fileOpen.c .....file handling routines to open SU files
- gaussGen.c .....generate a Gaussian distribution of random numbers
- getBeamTimes.c ..... stores energy fields (beams) in arrays at certain time steps
- getModelInfo.c .....reads gridded model file to compute min/max and sampling intervals
- getParameters.c ..... reads in all parameters to set up a FD modeling
- getRecTimes.c .....stores the wavefield at the receiver positions
- getWaveletInfo.c reads source wavelet file and computes maximum frequency and sampling
- getpars.c ..... functions to get parameters from the command line, from SU
- name_ext.c .....inserts a character string after the filename, before the extension
- readModel.c .... reads gridded model files and computes medium parameters for FD kernels
- recvPar.c .....calculates the receiver positions based on the input parameters
- spline3.c ..... computes interpolation based on third order splines
- taperEdges.c .....tapers the wavefield to suppress unwanted reflections from the edges
- verbosepkg.c .....functions to print out verbose, error and warning messages to stderr
- viscoacoustic4.c .....Kernel of visco-acoustic FD using 4'th order derivatives
- viscoelastic4.c .....Kernel of visco-elastic FD using 4'th order derivatives
- wallclock_time.c .....function used to calculate wallclock time
- writeRec.c .....writes the receiver array(s) to output file(s)
- writeSnapTimes.c ..... writes gridded wavefield(s) at a desired time to output file(s)
- writeSrcRecPos.c .....writes the source and receiver positions into a gridded file
- writesufile.c .....writes an 2D array to a SU file

References

Almobarak, Mohammed, 2021, Plane-wave Marchenko imaging method: applications: MSc. Thesis, Delft University of Technology.

Behura, J., K. Wapenaar and R. Snieder, 2014, Autofocus imaging: Image reconstruction based on inverse scattering theory: Geophysics, **79** (3), A19–A26.

Brackenhoff, J., 2016, Rescaling of incorrect source strength using Marchenko Redatuming (M.Sc. thesis): TU Delft Repository, Delft University of Technology.

Brackenhoff, J., Thorbecke, J., and Wapenaar, K., 2019, Virtual sources and receivers in the real Earth - considerations for practical applications: Journal of Geophysical Research - Solid Earth, **124** (11), 802–821.

Brackenhoff, J., Thorbecke, J., Meles, G., Koehne, V., Barrera, D., and Wapenaar, K., 2022, 3D Marchenko applications: Implementation and examples: Geophysical Prospecting, Vol. **70**, p. 35–56.

Broggini, F., K. Wapenaar, J. van der Neut and R. Snieder, 2014, Data-driven Green's function retrieval and application to imaging with multidimensional deconvolution: Journal of Geophysical Research - Solid Earth, Vol. **119**, p. 425–441.

Broggini, F., R. Snieder, and K. Wapenaar, 2014, Data-driven wave field focusing and imaging with multidimensional deconvolution: Numerical examples from reflection data with internal multiples: Geophysics, **79** (3), WA107–WA115.

Cohen, J. K. and J. W. Stockwell, 2016, CWP/SU: Seismic Un*x Release No. 43R4: an open source software package for seismic research and processing: Center for Wave Phenomena, Colorado School of Mines.

Costa Filho, C. A. da, M. Ravasi, A. Curtis, and G.A. Meles, 2014, Elastodynamic Green's function retrieval through single-sided Marchenko inverse scattering: Physical Review E, **90**, 063201.

Costa Filho, C. A. da, G.A. Meles, A. Curtis, M. Ravasi and A. Kritski, 2017, Imaging strategies using Marchenko focusing functions: 79th Annual International Meeting, European Association of Geoscientists and Engineers, Expanded Abstracts, Tu P9 15.

Dukalski, M.S., and K. de Vos, 2017, Marchenko inversion in a strong scattering regime including surface-related multiples: Geophysical Journal International, **212** (2), 760–776.

Jia, X., Guitton, A. and Snieder, R., 2018, A practical implementation of subsalt Marchenko imaging with a Gulf of Mexico data set: Geophysics, **83** (5), S409–S419.

Lomas, A.,and A. Curtis, 2019, An introduction to Marchenko methods for imaging: Geophysics, **84** (2), F35–F45.

Lomas, A., S. Singh, and A. Curtis, 2020, Imaging vertical structures using Marchenko methods with vertical seismic-profile data: Geophysics, **85** (2), S103–S113.

Matias, M. A., R. Pestana, and J. van der Neut, 2018 Marchenko imaging by unidimensional deconvolution: Geophysical Prospecting, **66** (9), 1653–1666.

Meles, G.A., K. Löer, M. Ravasi, A. Curtis and C.A. da Costa Filho, 2015, Internal multiple prediction and removal using Marchenko autofocusing and seismic interferometry: Geophysics, **80** (1), A7–A11.

Meles, G.A., C.A. da Costa Filho and A. Curtis, 2017, Synthesising singly-scattered waves (primaries) from multiply-scattered data: 79th Annual International Meeting, European Association of Geoscientists and Engineers, Expanded Abstracts, Tu P4 11.

Meles, G.A., K. Wapenaar, and J. Thorbecke, 2018, Virtual plane-wave imaging via Marchenko redatuming: Geophysical Journal International, **214** (1), 508–519.

Meles, G.A., L. Zhang, J. Thorbecke, K. Wapenaar and E. Slob, 2020 Data-driven retrieval of primary plane-wave responses: Geophysical Prospecting, **68**, 1834–1846.

Mildner, C., F. Broggini, K. de Vos and J.O.A. Robertsson, 2017, Source Wavelet Amplitude Spectrum Estimation Using Marchenko Focusing Functions: 79th Annual International Meeting, EAGE, Expanded Abstracts, We B2 01.

Mildner C., F. Broggini, K de Vos, and J.O.A. Robertsson, 2019 Accurate source wavelet estimation using Marchenko focusing functions: Geophysics, **84** (6), Q73–Q88.

Pereira, R., Ramzy, M., Griscenco, P., Huard, B., Huang, H., Cypriano, L. and Khalil, A., 2019, Internal multiple attenuation for OBN data with overburden/target separation: 89nd Annual International Meeting, Society of Exploration Geophysicists, Expanded Abstracts, 4520–4524.

Qu, S. and Verschuur, D. J. 2020, Simultaneous joint migration inversion for high-resolution imaging/inversion of time-lapse seismic datasets: Geophysical Prospecting, **68**, p. 1167–1188.

Ravasi, M., I Vasconcelos, A. Kritski, A. Curtis, C.A. da Costa Filho and G.A. Meles, 2016, Target-oriented Marchenko imaging of a North Sea field: Geophysical Journal International, **205** (1), 99–104.

Ravasi, M., 2017, Rayleigh-Marchenko redatuming for target-oriented, true-amplitude imaging: Geophysics, **82** (6), S439–S452.

Ravasi, M., and I. Vasconcelos, 2020, PyLops - A linear-operator Python library for scalable algebra and optimization: SoftwareX, **11**, 100361.

Rietveld, W., G. Berkhout, and K. Wapenaar, C., 1992, Optimum seismic illumination of hydrocarbon reservoirs: Geophysics, Vol. **57** (10), p. 1334–1345.

Singh, S., R. Snieder, J. Behura, J. van der Neut, K. Wapenaar and E. Slob, 2015, Marchenko imaging: Imaging with primaries, internal multiples, and free-surface multiples: Geophysics, **80** (5), S165–S174.

Singh, S., J. van der Neut, K. Wapenaar and R. Snieder, 2016, Beyond Marchenko - Obtaining virtual receivers and virtual sources in the subsurface: 86th Annual International Meeting, SEG, Expanded Abstracts, p. 5166–5171.

Singh, S., R. Snieder, J. van der Neut, J. Thorbecke, E. Slob, and K. Wapenaar, 2017, Accounting for free-surface multiples in Marchenko imaging: Geophysics, **82** (1), R19–R30.

Slob, E., K. Wapenaar, F. Broggini and R. Snieder, 2014, Seismic reflector imaging using internal multiples with Marchenko-type equations: Geophysics, **79** (2),S63–S76.

Slob, E., 2016, Green's function retrieval and Marchenko imaging in a dissipative acoustic medium: Physical Review Letters, **116** (16), 164301.

Sripanich, Y., Vasconcelos, I., and Wapenaar, K., 2019, Velocity-independent Marchenko focusing in time- and depth-imaging domains for media with mild lateral heterogeneity: Geophysics, **84** (6), Q57–Q72.

Staring, M., J. van der Neut and K. Wapenaar, 2016, An interferometric interpretation of Marchenko redatuming including free-surface multiples: 86th Annual International Meeting, SEG, Expanded Abstracts p. 5172–5176.

Staring, M., R. Pereira, H. Douma, J. van der Neut, and K. Wapenaar, 2017, Adaptive double-focusing method for source-receiver Marchenko redatuming on field data: 87th Annual International Meeting, SEG, Expanded Abstracts, p. 4808–4812.

Staring, M., R. Pereira, H. Douma, J. van der Neut, and K. Wapenaar, 2018, Source-receiver Marchenko redatuming on field data using an adaptive double-focusing method: Geophysics, **83** (6), S570–S590.

Thomsen, H., 2016, Investigating the robustness of Green's function retrieval via Marchenko focusing and Seismic Interferometry: MSc Thesis, ETH Zürich.

Thorbecke, J. and D. Draganov, 2011, Finite-difference modeling experiments for seismic interferometry: Geophysics, **76** (6), H1–H18.

Thorbecke, J., J. van der Neut and K. Wapenaar, 2013, Green's function retrieval with Marchenko equations: A sensitivity analysis: 83th Annual International Meeting, SEG, Expanded Abstracts, SPMI 6.3.

Thorbecke, J., E. Slob, J. Brackenhoff, J. van der Neut, and K. Wapenaar, 2017, Implementation of the Marchenko method: Geophysics, **82** (6), WB29–WB45.

Thorbecke, J., and J. Brackenhoff, 2020, OpenSource code for Finite Difference, Marchenko and utilities: https://github.com/JanThorbecke/OpenSource, doi 10.5281/zenodo.3374728.

Thorbecke, J. and L. Zhang and K. Wapenaar and E. Slob, 2021, Implementation of the Marchenko multiple elimination algorithm: Geophysics, **86** (2), F9–F23.

van IJsseldijk, J., J. van der Neut, J. Thorbecke, and K. Wapenaar, 2023, Extracting small time-lapse traveltime changes in a reservoir using primaries and internal multiples after Marchenko-based target zone isolation: Geophysics, accepted, 2023

van IJsseldijk, J., J. Brackenhoff, J. Thorbecke, and K. Wapenaar, 2023, Time-lapse applications of the Marchenko method on the Troll field: under review for Geophysical Prospecting, 2023

Van der Neut, J., J. Thorbecke, K. Wapenaar and E. Slob, 2015a. Inversion of the multidimensional Marchenko equation: 77th Annual International Meeting, EAGE, Extended Abstracts, We N106 04.

Van der Neut, J., I. Vasconcelos and K. Wapenaar, 2015b, On Green's function retrieval by iterative substitution of the coupled Marchenko equations: Geophysical Journal International, **203** (2), 792–813.

Van der Neut, J., K. Wapenaar, J. Thorbecke, E. Slob and I Vasconcelos, 2015c, An illustration of adaptive Marchenko imaging: The Leading Edge, Vol. **34**, p. 818–822.

Van der Neut, J., and K. Wapenaar, 2016, Adaptive overburden elimination with the multidimensional Marchenko equation: Geophysics, **81** (5), T265–T284.

van der Neut, J., Johnson, J.L., van Wijk, K., Singh, S., Slob, E., and Wapenaar, K., 2017, A Marchenko equation for acoustic inverse source problems: J. Acoust. Soc. Am., Vol. 141 (6), 4332-4346.

Verschuur, E., A. Berkhout, and K. Wapenaar, 1992, Adaptive surface-related multiple elimination: Geophysics, **57** (9), 1166–1177.

Wapenaar, K., F. Broggini and R. Snieder, 2012, Creating a virtual source inside a medium from reflection data: heuristic derivation and stationary-phase analysis: Geophysical Journal International, Vol. **190** (2), p. 1020–1024.

Wapenaar, K., F. Broggini, E. Slob and R. Snieder, 2013, Three-dimensional single-sided Marchenko inverse scattering, data-driven focusing, Green's function retrieval, and their mutual relations: Physical Review Letters, Vol. 110 (8), 084301.

Wapenaar, K., 2014, Single-sided Marchenko focusing of compressional and shear waves: Physical Review E, **90** (6), 063202.

Wapenaar, K., J. Thorbecke, J. van der Neut, F. Broggini, E. Slob and R. Snieder, 2014a, Green's function retrieval from reflection data, in absence of a receiver at the virtual source position: The Journal of the Acoustical Society of America, **135** (5), 2847–2861.

Wapenaar, K., J. Thorbecke, J. van der Neut, F. Broggini, E. Slob and R. Snieder, 2014b, Marchenko imaging: Geophysics, **79** (3), WA39–WA57.

Wapenaar, K., J. Thorbecke, and J. van der Neut, 2016a, A single-sided homogeneous Green's function representation for holographic imaging, inverse scattering, time-reversal acoustics and interferometric Green's function retrieval: Geophysical Journal International, Vol. **205**, p. 531–535.

Wapenaar, K., J. Thorbecke, J. van der Neut, E. Slob and R. Snieder, 2017, Virtual sources and their responses, Part II: Data-driven single-sided focusing: Geophysical Prospecting, **65** (6), 1430–1451.

Wapenaar, K., and van IJsseldijk, J., 2020, Discrete representations for Marchenko imaging of imperfectly sampled data: Geophysics, **85** (2), A1–A5.

Wapenaar, K., Brackenhoff, J., Dukalski, M., Meles, G., Reinicke, C., Slob, E., Staring, M., Thorbecke, J., van der Neut, J., and Zhang, L., 2021, Marchenko redatuming, imaging, and multiple elimination and their mutual relations: Geophysics, Vol. **86** (5), p. WC117–WC140.

Zhang, L., and M. Staring, 2018, Marchenko scheme based internal multiple reflection elimination in acoustic wavefield: Journal of Applied Geophysics, **159** (9), 429–433.

Zhang, L., and E. Slob, 2019, Free-surface and internal multiple elimination in one step without adaptive subtraction: Geophysics, **84** (1), A7–A11.

Zhang, L., J. Thorbecke, K. Wapenaar, and E. Slob, 2019, Transmission compensated primary reflection retrieval in data domain and consequences for imaging: Geophysics, **84** (4), Q27–Q36.

Zhang, L., and E. Slob, 2020a, A field data example of Marchenko multiple elimination: Geophysics, **85** (2), S65–S70.

Zhang, L., and E. Slob, 2020b, A fast algorithm for multiple elimination and transmission compensation in primary reflections: Geophysical Journal International, **221** (1), 371–377.

Zhang, L., and E. Slob, 2020c, Marchenko multiple elimination of a laboratory example: Geophysical Journal International, **221** (2), 1138–1144.

Index