User Manual: Downward Continuation to the seafloor of WAS data

Author: Clara Estela Jiménez Tejero. Developed at BCSI group at ICM-CSIC (Barcelona, Spain).

Abstract

We present a user friendly open source HPC software for redatuming 2D WAS field data (recorded at OBS's) to the sea bottom for any seafloor relief. The main ingredient is the acoustic wave equation used backward in time, allowing the redatuming of the OBS gathers.

As input, OBS data are required in a series of SU file (one file for each OBS), and the bathymetry of the shots in a ascii file. Also, a series of parameters have to be provided through an input file which is read in the execution line. By default, the p-wave velocity model for the column water is considered constant but also XBT data can be provided to build a specific Vp water model.

1 Software requirements

- The software presented here is an open source code developed under fortran 90 and HPC architecture built with open MPI.
- It is needed parallel compilation using MPI (mpif90 command installed for compilation and mpirun command installed for execution).
- Seismic Unix tool [1] installed and a minimum of knowledge working with it. It is a open source software necessary to work/convert/visualize seismic binary data files in SU format.
- For time consuming reasons, the best is to run it on a cluster however it is possible to work in a local computer too.

2 Installation

- Type in the terminal, inside the 'src/' folder:
  make

- You can install the software manually:
  - Compilation of the modules first:
    mpif90 -c modules.f90
  - Compilation of all and generation of executable DC_WAS:
    mpif90 *.f90 -o DC_WAS
- After it is properly installed, include the path in your .bashrc, so that the executable is accesible from any location:
  export PATH="/home/user/path/to/src:$PATH"

3 Quick start

- Give a value to all the parameters in the input parameter file as specified in section 4.1.
- Provide in input folder:
  - The WAS field data. Each OBS gather recording in a separate file and be named followed by the number of the OBS (example for 3 OBS's: name1, name2 and name3). Also, it is necessary to convert each data file to SU format.
  - The bathymetry information for all the OBS data is needed in a single ascii file. Navigation has to be included too, if it is not specified in SU headers (see section 4.2).
- Run the program (DC_WAS), where the specific Parfile is located:
  mpirun -np numtasks DC_WAS Parfile
  (Here 'numtasks' refers to the number of cores which will be used to parallelize the calculation. The most efficient and maximum value for 'numtasks' is equal to the number of OBS's.)

4 Input data

4.1 Input parameter file

The input parameter file is a ascii file and its structure consist on different lines, each line containing the specific parameter name, followed by ':' and at least one spacebar followed by the correspondent parameter value:

parameter_1: value_1
parameter_2: value_2

...
parameter$_n$: value$_n$

The file can be named as the user want. Also, it is convenient to be familiar with the header structure of the SU files, as information is obtained from the headers, if this is specified in the parameters file (see https://wiki.seismic-unix.org/sudoc:su_data_format). The list of parameters are described in the following lines (all of them might not be necessary):

1. endianness_machine: integer variable
   - It should be set to 0 (little endian) or 1 (big endian).
   - If not included, the value by default is little endian, 'endianness_machine: 0'.
2. endianness_data: integer variable
   - It should be set to 0 (little endian) or 1 (big endian).
   - If not included, the value by default is big endian, 'endianness_data: 1'.
3. input_folder: character variable
   - Path to the folder where the input files are located.
   - Example, folder_input: '/home/user/DCtest/data/input'.
   - Notice that input and output folder can be the same directory. It is the user choice to separate input and output data or place all data in same directory.
4. output_folder: character variable
   - Path to the folder where the output files will be stored.
   - Example, folder_output: '/home/user/DCtest/data/output'.
   - Notice that input and output folder can be the same directory. It is the user choice to separate input and output data or place all data in same directory.
5. su_file: character variable
   - Name of the binary SU files containing the OBS data. For example, for 10 OBS's, user should provide the SU files containing the OBS data named as: name_1, name_2, ..., name_10. In this case, "su_file: name_".
6. nav_file: character variable
   - Name of the bathymetry ascii file, which contain the bathymetry information and also might contain the navigation (position of shots).
   - More information in section 4.2.

7. sx_sy_header: integer variable

   - If 'sx_sy_header: 1', the position of the shots are read from the headers of the SU files (referred as 'sx' and 'sy' in the header). Also in this case, the parameter 'scalco' is automatically read in the headers to correctly obtain sx and sy parameters (UTM coordinates, in meters).
   - If not included, the value by default is 'sx_sy_header: 0'. In this case, the position of each shot is read from the ascii file (explained in section 4.2).

8. dt: real variable.
   Time sampling of the OBS fiel data (seconds).

9. nt: integer variable
   Number of time steps.

10. NumOBS: integer variable
    Number of OBS's.

11. dshots: real variable
    Distance between shots (meters).

12. shot_init: integer variable
    shotID of the first shot provided in the su_files.

13. shot_fin: integer variable
    shotID of the last shot provided in the su_files.

14. shot_depth: real variable
    Depth of the shotgathers (meters).

15. dmodel: real variable

    - Space sampling of the p-wave velocity model (meters).
    - If not specified, it is considered by default, 'dmodel: 25'.

16. water_velocity: real variable

    - Water velocity model (meters/second).
    - If not included, it is considered by default, water_velocity: 1500.

17. vp_file: character variable

    - Name of the file which contains the XBT data available at each specific OBS gather.

- It needs to be included if user wants to describe the water column with a realistic velocity model.
- If 'vp_file: vp.dat', this is an example of a valid content for the file vp.dat:

  1008 vp_1.dat
  5400 vp_2.dat
  9821 vp_3.dat

  In this example, there is data available for three different shotgathers with shotID: 1008, 5400 and 9821. Each of the files, vp_1.dat, vp_2.txt and vp_3.dat, must contain 2 columns; the first column indicates the depth (meters), and the second column indicates the p-wave velocity at each available depth.
- There are different ways to calculate the p-wave velocity from XBT data (temperature and salinity).
- The software interpolates the given data into the resolution required for the model (dmodel). In the case of not providing data up to the real depth at each shot gather, the software uses the deepest value provided.
- Nevertheless, building a realistic velocity model with XBT data is not a requirement to properly redatume our data from the surface to the bottom of the sea. We show in [1] that using a realistic homogeneous value for the water column, the results are very similar even for deep waters.
- If 'vp_file' parameter is not included, the water column is considered homogeneous using the velocity value specified in the parameter 'water_velocity'.

18. save_gnuplot_txt: integer variable

    - Activate this parameter as 'save_gnuplot_txt: 1', to save the shotgathers in ascii gnuplot format: X(1:NumRec), Y(1:nt), Z=shot gather.
    - If not included, this parameter is not activated by default, 'save_gnuplot_txt: 0'.

19. save_matlab_txt: integer variable

    - Activate this parameter as 'save_matlab_txt: 1', to save shotgathers in ascii matlab format: shot gather(nt,NumRec).
    - If not included, this parameter is not activated by default, 'save_matlab_txt: 0'.

4.2 Bathymetry and navigation information

The bathymetry information of the shotgathers must be given in an ascii file. But depending on the options selected in the parameter input files, the navigation information has

also to be provided together with the bathymetry in the same ascii file. Depending on the activation or not of the parameters 'sx_sy_header':

- If 'sx_sy_header: 0', the navigation (in UTM coordinates) has to be included in the ascii file.
- If 'sx_sy_header: 1', the navigation (in UTM coordinates) is extracted from the SU headers and don't need to be included here.

Therefore there are two options to build the bathymetry/navigation file:

- If 'sx_sy_header: 0', the file must be a 4-column structure:

  shotID_1 X_1(UTM, meters) Y_1(UTM, meters) Z_1(meters)
  shotID_2 X_2(UTM, meters) Y_2(UTM, meters) Z_2(meters)
  ...
  shotID_n X_n(UTM, meters) Y_n(UTM, meters) Z_n(meters)

- If 'sx_sy_header: 1'. The file must be 2-column structure:

  shotID_1 Z_1(meters)
  shotID_2 Z_2(meters)
  ...
  shotID_n Z_n(meters)

Notes:

- The units for each shot position, UTM, refer to the 'Universal Transverse Mercator' coordinates (in meters).
- The parameters, $\text{shotID}_i > 0$, is the shot number, normally same value than for the parameter 'fldr' in SU header.
- The depth of the seafloor at each shot gather position, Z_i, in meters, can be expressed as positive or negative numbers, the easiest for the user. It is always used as $|Z_i|$.

5 Output data

The files located in the output folder after the execution of the program:

- DC_OBS_i (SU format). OBS gather results after step DC, where "i" is the OBS number, from 1 to n. For just one OBS, it will be just named as "DC_OBS".

- bathymetry_meters.txt (ascii file).
  This file contains the bathymetry interpolated to the grid of the model. It contains 2 columns: x-axis (grid model in meters) and y-axis (bathymetry).

- Vp_model.txt (ascii file).
  This file contains the 2D-Vp water model in case of using XBT data to characterize the water column (in case of using, 'vp_file: 1').

References

[1] Murillo, Alejandro E. and J. Bell. "Distributed Seismic Unix: a tool for seismic data processing." Concurrency and Computation: Practice and Experience 11 (1999): 169-187. Seismic Unix tool: https://wiki.seismic-unix.org/doku.php.