# User Guide:
# Open Source HPC Software for Redatuming 2D WAS Field Data

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

**Abstract**

Introducing a user friendly open-source HPC software designed for efficiently redatuming 2D Wide-Angle Seismic (WAS) field data to the seabed, accommodating any seafloor relief. The software employs the acoustic wave equation in reverse time, enabling redatuming of OBS gathers.

The input consists of OBS data stored in a series of SU files (one per OBS), along with the bathymetry of the shots in a single ASCII file. Furthermore, users can specify various parameters via an input file provided during execution. By default, a constant p-wave velocity model is used for the column water, but the option to create a custom Vp water model using XBT data is also available.

OBS field data should be muted from time 0 to the first arrival, to avoid the propagation of noise. As well, the field data should be cutted at certain time to avoid the propagation of the multiple.

## 1 Software Requirements

The software presented here is an open-source code developed in Fortran 90 for High-Performance Computing (HPC) architecture, built with Open MPI.

- **Parallel Compilation:** The software requires parallel compilation using MPI. Ensure that the mpif90 command is installed for compilation and the mpirun command for execution.
- **Seismic Unix Tool:** To work with the software, you need to have the Seismic Unix tool [1] installed. This open-source software is necessary for handling, converting, and visualizing seismic binary data files in SU format.
- **System Requirements:** For optimal performance, it is recommended to run the software on a cluster. However, it is also possible to run it on a local computer.

## 2 Installation

To install the software, follow these steps:

1. **Automatic Installation:**
   - Open the terminal and navigate to the 'src/' folder.
   - Type the following command to compile the software:

   ```
   make
   ```

2. **Manual Installation:**
   - First, compile the modules:

   ```
   mpif90 -c modules.f90
   ```

   - Next, compile all files and generate the executable 'DC_WAS_run:

   ```
   mpif90 *.f90 -o DC_WAS_run
   ```

3. **Setting up the PATH:**
   - After the software is properly installed, include its path in your .bashrc file to make the executable accessible from any location:

   ```
   export PATH="/home/user/path/to/src:\$PATH"
   ```

   - Replace "/home/user/path/to/src" with the actual path to the 'src/' folder.

## 3 Quick Start

To quickly get started with the software, follow these steps:

1. **Input Parameter File:** Assign values to all the parameters in the input parameter file, as specified in Section 4.1.
2. **Input Folder Contents:** Place the following items in the input folder:

- **WAS Field Data:** For each OBS gather recorded, provide a separate SU file (e.g., OBS_1.su, OBS_2.su, OBS_3.su for three OBSs). Ensure that each data file is converted to SU format (for more information, refer to the SU tutorial [1] or the User Guide for DC streamer data available in the same GitHub account[1]).
- **Bathymetry and Navigation Information:** Prepare a single ASCII file containing bathymetry information for the shots. Please, include here as well navigation data if not specified in SU headers (see Section 4.2 for further details).

3. **Running the Program:** Execute the program (DC_WAS_run) with the specific parfile in the terminal using the following command:

```
mpirun -np numtasks DC_WAS_run parfile
```

   Here, 'numtasks' refers to the number of cores used for parallelizing the calculation. For optimal performance, set 'numtasks' equal to the number of OBSs.

## 4 Input data

### 4.1 Input Parameter File

The input parameter file is an ASCII file with a specific structure. Each line contains a parameter name followed by a colon and at least one space, and then the corresponding parameter value. The list of parameters and their descriptions are as follows:

1. **endianness_machine:** Integer variable
   - Set to 0 (little endian) or 1 (big endian).
   - Default value: 0 (little endian).
2. **endianness_data:** Integer variable
   - Set to 0 (little endian) or 1 (big endian).
   - Default value: 1 (big endian).
3. **input_folder:** Character variable
   - Path to the folder containing input files.
   - Example: `folder_input:  '/home/user/DCtest/data/input'`.
4. **output_folder:** Character variable

[1]https://github.com/ejimeneztejero/DC

- Path to the folder where output files will be stored.
- Example: `folder_output:  '/home/user/DCtest/data/output'`.

5. **obs_file_list:** Character variable

   - Name of ascii file containing the name of the different OBS data files, one file per OBS. For example, for 5 OBS gathers. In this case, 'obs_file_list: data.txt', where the file data.txt is an ascii file where each line corresponds to the name of each SU file containing each OBS gather:
     obs1.su
     obs2.su
     obs3.su
     obs4.su
     obs5.su

6. **byte_shotnumber:** Integer variable

   - Number of byte in SU header where the shotID must be read.
   - Example: "byte_shotnumber: 9" if read in fldr header.
   - Default value: 5 (reading shotID from tracr).

7. **nav_file:** Character variable

   - Name of the bathymetry ASCII file containing bathymetry information and, optionally, navigation data.
   - More information in Section 4.2.

8. **TWT_option**: integer variable

   - If 'TWT_option: 1', the bathymetry information has to be given as the Two-Way-Traveltime (TWT) in the nav_file (in seconds).
   - If not included, the default value is 'TWT_option: 0'. In this case, the bathymetry information must be given in meters in the input nav_file.
   - For more details, check section 4.2.

9. **sx_sy_header:** Integer variable

   - Set to 1 to read shot positions from the headers of the SU files ('sx' and 'sy' in the header). In this case, 'scalco' parameter is automatically read to obtain sx and sy as UTM coordinates (in meters).
   - Default value: 0 (positions read from the ASCII file, explained in Section 4.2).

10. **dt:** Real variable

- Time sampling of the OBS field data (seconds).

11. **nt:** Integer variable
    - Number of time steps.
12. **NumOBS:** Integer variable
    - Number of OBSs.
13. **shot_init:** Integer variable
    - shotID of the first shot.
14. **shot_fin:** Integer variable
    - shotID of the last shot.
15. **shot_depth:** Real variable
    - Depth of the shots (meters).
16. **dmodel:** Real variable
    - Space sampling of the p-wave velocity model (meters).
    - Default value: 25.
17. **water_velocity:** Real variable
    - Water velocity model (meters/second).
    - Default value: 1500.
18. **vp_file:** Character variable
    - Name of the file containing XBT data for each specific OBS gather.
    - Needed to describe the water column with a realistic velocity model.
    - More details in the description.
    - If not included, the water column is considered homogeneous using the velocity value specified in the parameter 'water_velocity'.
19. **save_gmt:** Integer variable
    - Set to 1 to also save OBS gathers in ASCII gmt format: X(1:NumShots), Y(1:nt).
    - Default value: 0 (not activated).
20. **save_matlab:** Integer variable

- Set to 1 to also save OBS gathers in ASCII MATLAB format: OBS gather(1:nt, 1:NumShots).
- Default value: 0 (not activated).

## 4.2 Bathymetry and Navigation Information

The bathymetry information for the shots must be provided in an ASCII file. Depending on the 'sx_sy_header' parameter's activation in the input parameter file:

- If 'sx_sy_header: 0', the navigation (in UTM coordinates) must be included in the same ASCII file.
- If 'sx_sy_header: 1', the navigation (in UTM coordinates) is automatically extracted from the SU headers and doesn't need to be included in this file.

There are two options to structure the bathymetry/navigation file:

- If 'sx_sy_header: 0', the file must have 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 have a 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 (X and Y in UTM) are in meters, using the 'Universal Transverse Mercator' coordinates.
- The 'shotID$_i$' parameter is the shot number, which must be found in some SU header (probably tracr, fldr ...). Remember to fix the parameter byte_shotnumber in the parfile to indicate where to read the shotID. By default, if not specified, shotID will be read from tracr (byte 5).
- The depth of the seafloor at each shot position (Z$_i$) can be expressed as positive or negative numbers. The software uses the absolute value, $|Z_i|$, for computations.

## 5 Output Data

After executing the program, the following files will be located in the output folder:

- **DC_OBS_i (SU format):** OBS gather results after DC, where "i" is the OBS number, ranging from 1 to n. If there is only one OBS, it will be named "DC_OBS".
- **bathymetry_meters.txt (ASCII file):**
  This file contains the bathymetry interpolated to the grid of the model. It has 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 if XBT data is used to characterize the water column (when 'vp_file: 1' is specified).

## References

[1] [SU]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.