WSJointInv2D-MT-DCR
version 1.0

User Manual

Puwis Amatyakul
Department of Physics, Faculty of Science
Mahidol University
Rama 6 Rd., Rachatawee, Bangkok 10400
THAILAND
apuwis@gmail.com

Chatchai Vachiratienchai
Curl-E Geophysics Co. Ltd., Bangkok, Thailand

Weerachai Siripunvaraporn
Department of Physics, Faculty of Science
Mahidol University, Bangkok, THAILAND
wsiripun@gmail.com

Copyright © 2016 Weerachai Siripunvaraporn. All rights reserved.
1. INTRODUCTION

1.1 What is Joint2D-MT-DCR?

WSJointInv2D-MT-DCR (Amatyakul et al., 2017) is a joint inversion program for the 2-D magnetotelluric data and the 2-D direct current resistivity (DCR) data. Its basic inversion algorithm is based mainly on the data space Occam’s inversion (Siripunvaraporn and Egbert, 2000; Siripunvaraporn et al., 2005; Boonchaisuk et al., 2008; Siripunvaraporn and Egbert, 2009; Vachiratienchai et al., 2010; Vachiratienchai and Siripunvaraporn, 2013) in which the objective of the inversion is to seek for the smoothest minimum structure model subject to an appropriate fit to the data. Because the Occam’s inversion is stable and converges to the desired misfit in relatively small number of iterations compared to most other methods, it is therefore preferable.

The code is written with Fortran 95. To run the code you need to have a Fortran compiler. It is mostly tested on Linux operating system, but should work on any systems.

1.2 What can the WSJointInv2D-MT-DCR do?

- Forward modeling for the MT data, the DCR data, and both types.
- Invert just the 2-D MT data.
- Invert the 2-D DCR data.
- Jointly invert the 2-D MT & DCR data.

1.3 Before using the code

Please read the conditions of use on the next pages
Conditions of Use

1. I will use WSJointInv2D-MT-DCR (hereinafter referred to as "the codes") for academic and non-profit purpose only. I will not accept any payment for use of the WSJointInv2D-MT-DCR program. For commercial use, users must contact the developer to get the license.

2. I will not redistribute the WSJointInv2D-MT-DCR program directly to any other person, including colleagues, professors or students. Any third party wishing to use the code should be referred to this website.

3. I will not modify the code without permission of the code developer.

4. I will send a copy or copies of papers (PDFs) resulting from use of the WSJointInv2D-MT-DCR program to the developer’s email.

5. In any publications resulting from use of this code, I will cite the code appropriately. Suggested citations are as follows.

Using the WSJointInv2D-MT-DCR program,

Referring to the data space Occam’s inversion used for MT and DCR data,
- Boonchaisuk., S., Vachiratienchai., C., Siripunvaraporn, W., 2008. Two-dimensional direct current (DC) resistivity inversion: data space Occam’s approach. Phys Earth Planet Inter. 168, 204–211.

6. I will report bugs to the developer.
7. If I move to another institution, change affiliations or email addresses, and wish to continue using the codes, I will report my change of address to the developer.
8. Violation of any of the conditions of use here will result in suspension of the right to use the codes for any future purposes. I will delete the code immediately if requested by the developer.
9. The developer is not responsible for any damages to the user caused by usage of this code.

Thank you very much.

Puwis Amatyakul, Chatchai Vachiratienchai, and Weerachai Siripunvaraporn

Copyright © 2016 Weerachai Siripunvaraporn
2. SYSTEM CONFIGURATION AND CODE COMPILATION

2.1 Code compilation

The code was written in Fortran 95 standard. It was tested on GNU’s gfortran compiler under Unix system (Linux/Mac terminal/Windows’ cygwin/minGW). Therefore, to run and compile the code, make sure you have all the following in your computer.

1. Your preferred Unix system.
2. GNU’s gfortran which included in CGG bundle
   (see https://gcc.gnu.org/wiki/GFortran)
3. GNU’s make package.
   (see https://www.gnu.org/software/make)
4. BLAS and LAPACK libraries
   (see www.netlib.org/blas/ and www.netlib.org/lapack/)

After installing all the required packages, you can proceed to the code compilation step. To configure the compilation on different systems, see the Makefile (on the code root folder). Type `make` on the code’s root folder. The example given in the Makefile should work in most of the system.

2.2 Compiled code testing

The executable file `wsjointinv2d_mt_dcr` should be generated out of the compilation if the compilation went correctly. There is an example folder, TEST, which contains files needed to recreate the synthetic case I in the publication (Amatyakul et al., 2017). Change to one of the subfolder, for example `sl-te-tm`, type `./././wsjointinv2d_mt_dcr` (or path to the executable file if you moves it into another folder). The program should start and run, as shown in Figure 1.
Figure 1: WSJointInv2D-MT-DCR start program interface.

NOTE: It is possible that you may not be able to run the code even the compilation is success. Limitation of the code depends greatly on the memory (RAM) of your machine. If you do not have large memory but try to run on a large data set (e.g., N > 50,000) with a large grid discretization (M > 200 x 100 mesh), it is likely to crash. Before applying the code, please know your RAM size. The memory used in this code depends greatly on the NN + MN, where N is the number of data and M is the number of grid discretization. Make sure your RAM size exceeds the NN + MN, at least more than 30%.
3. INPUTS

There are several input files that need to be prepared before running WSJoint2D-MT-DCR. The required input files are

- Startup file (auto or any name)
- Data file (.pmt for magnetotelluric data and .pdc for DC-resistivity data)
- Model file (.rmd file)

There are also additional files to manually adjust some of the inversion parameters.

- setparameter file (to set inversion parameters from the default value)
- .cm file (to set covariance matrix to each data set)

3.1 Startup file

The Startup file lists all the input files. You could save the startup file in

1. any arbitrary name: the program wsjointinv2d-mt-dcr will ask for the file.
2. ‘auto’: the program will start automatically after wsjointinv2d-mt-dcr is execute in the folder.

Example of Startup file

```
PROGRAM_TYPE inv
NSET 3
DATAFILE sl.pdc
DATATYPE 1
DATAFILE te.pmt
DATATYPE 2
DATAFILE tm.pmt
DATATYPE 2
REF_MODEL initm.rmd
START_MODEL initm.rmd
```
**Description of startup file**

<table>
<thead>
<tr>
<th>PROGRAM_TYPE</th>
<th>fwd or inv</th>
</tr>
</thead>
</table>

This will tell the program to execute the forward only mode (fwd) or run the inversion (inv). For the forward mode, \_fwd file will be generated after the program finish running.

<table>
<thead>
<tr>
<th>NSET</th>
<th>integer</th>
</tr>
</thead>
</table>

NSET is the number of data sets that will be used in the program. This number is the same number of the data input file.

<table>
<thead>
<tr>
<th>DATAFILE</th>
<th>filename</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATATYPE</td>
<td>1 or 2</td>
</tr>
</tbody>
</table>

Filename is the file’s name of your input data. DATATYPE indicates the data type whether it is magnetotelluric or DC-resistivity data.

- DATATYPE 1 for DC-resistivity data
- DATATYPE 2 for magnetotelluric data

You can import any numbers of data sets into the program by do this DATAFILE/DATATYPE data input block repeatedly.

<table>
<thead>
<tr>
<th>REF_MODEL</th>
<th>filename</th>
</tr>
</thead>
</table>

This file name is linked to the reference model file (.rmd file).

<table>
<thead>
<tr>
<th>START_MODEL</th>
<th>filename</th>
</tr>
</thead>
</table>

This file name is linked to the initial model file (.rmd file).

**3.2 Data file**

Currently, the program can be applied with the DC resistivity and magnetotelluric data. Please see the following file format (and see the example in the TEST folder).
3.2.1 DC-resistivity file

<table>
<thead>
<tr>
<th>TITLE Fwd_Response_4E</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA_COLLECTING_ELECTRODE separated</td>
</tr>
<tr>
<td>DATA_RESPONSE 1</td>
</tr>
<tr>
<td>DATA_ERROR 0.050</td>
</tr>
<tr>
<td>DATA_ARRANGE A B M N APP ERR</td>
</tr>
<tr>
<td>0.00 15.00 5.00 10.00 94.474045 1.051271</td>
</tr>
<tr>
<td>5.00 20.00 10.00 15.00 95.025740 1.051271</td>
</tr>
<tr>
<td>10.00 25.00 15.00 20.00 93.941167 1.051271</td>
</tr>
</tbody>
</table>

The detail in each parts of the .pdc file are listed below:

<table>
<thead>
<tr>
<th>TITLE text</th>
</tr>
</thead>
<tbody>
<tr>
<td>TITLE is a string input and without white spaces</td>
</tr>
</tbody>
</table>

| DATA_COLLECTING_ELECTRODE separated |
| DATA_RESPONSE 1 |

These two lines tell the program to read to input file in arbitrary 4-electrodes configuration (separated) with apparent resistivity input (1)

<table>
<thead>
<tr>
<th>DATA_ERROR 0.05</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATA_ERROR indicates the error floor of the input data in percent (for example, 0.05 means 5% error floor)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DATA_ARRANGE A B M N APP ERR</th>
</tr>
</thead>
</table>

DATA_ARRANGE indicates how the input data is formatted (data column). The typical 4-electrode format is A B M N APP ERR. However, you can swap the data column to suit your need. For pole-pole configuration, you may need to format the data according to A M APP ERR. APP is the apparent resistivity data in a unit of Ohm meter. ERR is the error of the input apparent resistivity data in a unit of Ohm meter.
### 3.2.2 Magnetotelluric file

The format of magnetotelluric data file is inherited from REBOCC (and DASOCC) of Siripunvaraporn and Egbert (2000). The detail is listed below.

```plaintext
MODE_TYPE te or tm
NUMBER_OF_RESPONSE 1 or 2
NUMBER_OF_PERIOD 13

...  
NUMBER_OF_STATION 21

...  
DATA_RESPONSE_NO_1 app

...  
ERROR_RESPONSE_NO_1 0.01

...  
DATA_RESPONSE_NO_1 phsdeg

...  
ERROR_RESPONSE_NO_1 0.01
```

The detail of each part of the file can be found in the following section.

- **MODE_TYPE te or tm**
  
  **MODE_TYPE** indicates model of 2-D magnetotelluric data, TE or TM.

- **NUMBER_OF_RESPONSE 1 or 2**
  
  **NUMBER_OF_RESPONSE** indicates which MT response(s) are being used, apparent resistivity in Ohm meter and/or phase in degree.

- **NUMBER_OF_PERIOD 13**
  
  **NUMBER_OF_PERIOD** is the number of period being used.

- **NUMBER_OF_STATION 21**
  
  **NUMBER_OF_STATION** is the number of magnetotelluric station being used.
DATA_RESPONSE_NO_1/2   app/phsdeg
...
ERROR_RESPONSE_NO_1/2   0.01
...

DATA_RESPONSE_NO_1 or 2 is the data section in the dimension of
NUMBER_OF_PERIOD (row) × NUMBER_OF_STATION (column).

App is the apparent resistivity in a unit of Ohm meter and phsdeg is the phase in a unit of degree. The unit of the data is also similar in ERROR_RESPONSE_NO_1/2.

3.3 Model file

The .rmd indicates the model file used in this code. Its format is almost similar to the model file of REBOCC and DASOCC of Siripunvaraporn and Egbert (2000) with minor changes.

nx   nz
Horizontal
...
LeftOffset   offset
Vertical
...
Model index 0 or 1 or integer
...

The detail of each part can be found here:

nx   nz

nx and nz indicate the horizontal and vertical grid size respectively (number of model blocks).

Horizontal
...

The horizontal grid (block) section needs the grid size from left to right including the extended model block to preserve modeling accuracy.
Offset indicates the total length of the left extended blocks. The coordinate 0 of the DC resistivity electrode and magnetotelluric station starts from this position.

The vertical grid section also requires the grid size in vertical direction. The recommendation on the vertical grid discretization is to increase to grid size logarithmically to preserve the modeling accuracy. We also recommend extending the total depth of the model to a very deep depth, > 3 -5 times of the profile length (the deeper block would be bigger according logarithmic growth).

Model index 0 or 1

For half-space model, you could use this style of input:

Model 1
100
which means the half-space model with homogenous resistivity of 100 Ohm meter.

For arbitrary model, you could use this style of input:

Model 0
... real values in nz (row) x nx (column)
You need to put matrix of the size nz x nx with resistivity values to represent your model.

For discrete resistivity model, you could use this style of input:

Model 3
10 100 1000
... integers in nz (row) x nx (column)
If you need to run a forward modeling from your desired model with 3 discrete resistivity values. First, put 3 in the model line. The next line will be the values listed for those 3
parts. Next, put the matrix of integers (1 to 3) to represent the model with those discrete values.

### 3.4 Inversion parameter

The inversion’s parameters are stored in `mathconst.f95`. You can change some of the parameter and recompile the code to get the new executable file. Or select some of the parameter and list them in `setparameter` file (global change). Without recompilation, the program will read those values and use them in the project in the executed folder. Model covariance parameters for each data set can be adjusted by put `.cm` after the data file name.

**Example of `setparameter` and `.cm` file**

```plaintext
CM_ROUTINE 5
DLENGTH_HOR 20
DLENGTH_VER 20
DTIME_STEP 1
ERR_FLOOR 0.05
```
4. OUTPUTS

There are several output files generated by the program: data, and for each iteration, the model and response files. In addition, the log file is also generated. The format of each file is inherited from the input data file.

- The **model file** is generated for each iteration. It will be named as `model_i.rmd`, where `i` is the iteration number. Note that `model_00.rmd` is also generated which is the initial model in arbitrary resistivity format. To read the model output, use the following routine:

```fortran
DO icol = 1:nx
  DO irow = 1:nz
    Model_output(irow,icol) = READ(value)
  END DO
END DO
```

- **Response file** will be generated for each iteration in the same format as the input data file (see input file section). The program will add `_i` suffix after the data file name, where `i` is the iteration number. For example, if the data files are `sl.pdc` and `te.pmt` respectively, in each iteration, there will be `sl_01.pdc` and `te_01.pdc` which are the responses calculated from the `model_01.rmd`. Note that there is also `sl_00.pdc` and `te_00.pdc` generated. They are the response of the starting model. The error presented in the response file is the error used by the program (filter by the error floor) to calculated the RMS.

- The **inversion summary** (log file) will be generated as `inversion_summary.txt` which includes RMS, mode norm and Lagrange multiplier of each iteration. Here, is the example of the summary file.
WSJointInv2D-MT-DCR (Version: 1.0)

Number of data sets : 3
Data set 1: DC data (sl.pdc) = 496 data
Data set 2: MT data (te.pmt) = 546 data
Data set 3: MT data (tm.pmt) = 546 data
Total number of data : 1588
Model size : 34 x 70 ( nz x nx )
Starting RMS = 9.71

Iteration, Phase, RMS, MNorm, Lagrange
1, 1, 3.24, 3.55, 41.50
2, 1, 1.81, 12.07, 7.43
3, 2, 0.98, 22.80, 2.80
4, 2, 1.00, 21.43, 3.53
5, 3, 1.02, 21.04, 3.72

> WSJointInv2D-MT-DCR is finished. End of execution...!
> Time elapsed is 381.30999999982305 sec.

In addition, the useful plotting script in Matlab for data and model visualization will be available at http://geophysics.sc.mahidol.ac.th/software/WSJointInv2D-MT-DCR.

Please contact apuwis@gmail.com for further bug report or inquiries.

Thank you very much.
Puwis Amatyakul, Chatchai Vachiratienchai, and Weerachai Siripunvaraporn

Copyright © 2016 Weerachai Siripunvaraporn