Manual for the software iEMmod

Jürg Hunziker[1], Jan Thorbecke[2], Joeri Brackenhoff[3] and Evert Slob[4]
Delft University of Technology, Delft, The Netherlands

Delft University of Technology, Delft, The Netherlands
[1]jurg.hunziker@unil.ch
[2]j.w.thorbecke@tudelft.nl
[3]j.a.brackenhoff@student.tudelft.nl
[4]e.c.slob@tudelft.nl

Contents

1 Introduction

The software iEMmod is a non-linear conjugate-gradient algorithm to invert frequency-domain electromagnetic data for a layered-Earth model. The forward solver used is our open-source layered-Earth electromagnetic modeling code EMmod (Hunziker et al., 2015). For the explanation of the forward code, we refer the user to the manual attached with the code to the previously cited paper. The implementation of the non-linear conjugate-gradient algorithm follows Shewchuk (1994).
In the next section the input parameters are explained and their related options and limitations discussed. Note, that some of the parameters are identical to those of the forward modeling software EMmod. Therefore, some parts of this manual are almost identical to the manual of the forward modeling code EMmod. Subsequently, we show how the results can be loaded and plotted in Matlab.

2 Input parameters

Several input parameters are expected by iEMmod. These input parameters can be specified using a script. Below the script found in the file `refTE_newmod_iso_01.scr` is shown. This script is included in the distribution of which this manual is also part of. The input parameters specified in this script will carry out an inversion of the TE-mode reflection response for the subsurface conductivity distribution.

```
#!/bin/bash

iemmod \
        freqvec=0.5 \
        file_weight=/data_a/hunziker/testing/oneweight.bin \
        file_in=/data_a/hunziker/testing/newmod_iso_xsize512_88.bin \
        path_out=/data_a/hunziker/testing/refTE_newmod_iso_01 \
        nx=512 \
        ny=512 \
        zsrc=200 \
        zrcv=200 \
        dx=80 \
        dy=80 \
        z=-1,0,200,1200,1220 \
        known=1,1,0,0,0 \
        econd=1,1,0.5,0.5,0.5 \
        eperm=17,17,17,2.1,17 \
        mperm=1,1,1,1,1 \
        component=88 \
        nd=1000 \
        startlogx=-6 \
        deltalogx=0.003125 \
        nlogx=4096 \
        kmax=0.628625 \
        c1=-0.00083333 \
        c2=-0.0029167 \
        maxpt=5000 \
        xdirect=1 \
        whichstep=2 \
        whichdir=1 \
        eps=1e-7 \
```

```
maxit=500 \
restartgrad=5 \
ngradrestart=4 \
epsa=1e-5 \
maxita=20 \
maxnogood=5 \
```

Subsequently, we will discuss all input parameters in order of appearance in this script.

2.1 freqvec

The parameter `freqvec` defines the frequency of the signal in Hz. If several components are inverted jointly, the frequency for each supplied data-file needs to be specified. In that case, freqvec becomes a vector of frequencies whereby each value is separated by a comma. Note that the amount of specified frequencies, components, weighting files and data-files must be equal. If this parameter is not specified, iEMmod will assume that only one data-file with a frequency of 0.5 Hz is used. Thus, the default value for this parameter is `freqvec=0.5`.

2.2 file_weight

This parameter specifies the location of the binary file containing the data weights. It is a string of maximum length of 512 characters. The file needs to have the same size as the input data. It specifies a weight for each receiver position. The current version uses only real weights. However, since the file needs to have the same format as the input parameter, the datatype needs to be complex. The imaginary part is simply ignored. For a joint inversion of several files, for each dataset a weighting-file needs to be specified. The different files are listed for the same parameter separated by commas. Note that the amount of specified frequencies, components, weighting files and data-files must be equal.

2.3 file_in

The parameter `file_in` specifies the location of the input file in which the data are stored. It is a string of maximum length of 512 characters. The file has to be in binary format organized in the same way as the output of EMmod (see the manual attached to Hunziker et al., 2015). For a joint inversion of several files, for several datasets need to be specified. The different files are listed for the same parameter separated by commas. Note that the amount of specified frequencies, components, weighting files and data-files must be equal.

2.4 path_out

Specify the path in which at each iteration the latest conductivity model as well as the misfit corresponding to that conductivity model are stored. Note, that the folder needs to exist before the program is started. Files from previous runs of iEMmod will be overwritten.

2.5 nx and ny

The two parameters `nx` and `ny` specify the dimensions of the grid of the supplied datafile. The same grid will be used to forward model the data during the inversion. Thereby specifies `nx` the amount of datapoints in the x-direction (inline direction) and `ny` specifies the amount of datapoints in the y-direction (crossline direction). Note that values for `nx` and `ny` need to be even. If an uneven number is entered, the program will add 1 to make it even. Thus, the smallest grid that can be used is a grid of two by two elements. Note also that the coordinate vectors are symmetric with respect to the zero coordinate with one more datapoint on the negative axis. If an axis has only two datapoints, they will be at minus `dx` (or minus `dy`) and 0. It is crucial that the supplied datafile is organized in the same way, otherwise the forward modeled data and the supplied data will not match.

2.6 zsrc and zrcv

The two parameters `zsrc` and `zrcv` specify the depth in meters of the source and the receivers, respectively. The vertical separation between the two affects the quality of the forward modeling step. The smaller the vertical separation, the broader is the bandwidth of the direct field. If the source and the receivers are at the same depth, the bandwidth of the direct field is infinite. Solving the integral of the Hankel transformation numerically becomes then impossible. Thus, if the source and the receivers are at the same depth, it is crucial to compute the direct field in the space domain. This can be done by setting the option `xdirect=1`. If the source and the receivers are at the same depth and also at an interface, not only the direct field but also the specular reflection of that interface have an infinite bandwidth. Since there is no option implemented to compute the specular reflection differently, in that case a small vertical offset between the source and the receivers has to be introduced. Consider also to increase the band of wavenumbers used for the computation by increasing the parameters `kmax` and `nd`.

2.7 dx and dy

The two parameters `dx` and `dy` define the sampling in meters in x-direction and y-direction, respectively. Note that the two parameters do not need to be equal.

2.8 z, known, econd, eperm and mperm

These five parameters are all arrays and are required to have the same amount of elements. Otherwise, iEMmod will produce an error and stop. The parameter `z` contains the depth of the layer-interfaces of the medium in meters. Note that the very first value of the `z`-array will be disregarded. We recommend to just put the value -1 there. Note also, that if a source or a receiver depth coincides with an interface, the corresponding source or receiver will be considered above the interface. The parameter `econd` contains the electric conductivity of the starting model for each layer in Siemens per meter. The parameter `eperm` specifies the electric permittivity relative to the value of free space and `mperm` specify the magnetic permeability relative to the value of free space. These two quantities are unitless because they are relative quantities. Note also that they are not updated in the inversion process, because at low frequencies usual for CSEM they hardly influence the data. The array `known` is 1 if the parameter is known and does not need to be estimated. The user should put a 0 for every layer he wants the algorithm to estimate the conductivity.

2.9 component

The parameter `component` specifies the orientation of both, the receivers and the source. This parameter consists of two numbers. The first one refers to the receiver while the second one refers to the source. The values 1, 2 and 3 represent an electric dipole antenna oriented in the inline-direction, the crossline-direction or the vertical direction, respectively. Options 77 and 88 refer the transverse magnetic (TM-mode) and the transverse electric (TE-mode) reflection response, respectively. All combinations except 77 and 88 are listed in table 1. Entry of any other combination will produce zeros in the forward model. The program will run but do nothing. If this parameter is not specified, EMmod will assume `component=11`. For a joint inversion, more than one component can be specified separated by commas. Note that the amount of specified frequencies, components, weighting files and data-files must be equal.

2.10 nd

The integral of the Hankel-Transformation of the forward modeling step is solved using a 61-point Gauss-Kronrod integration routine. This algorithm can not compute the complete integral in one go.

	inline electric source	crossline electric source	vertical electric source
inline electric receiver	11	12	13
crossline electric receiver	21	22	23
vertical electric receiver	31	32	33

Table 1: Choices for the parameter component to compute the electromagnetic field. Inline refers to parallel to the x-direction, while crossline refers to parallel to the y-direction. Note, that the choices 77 and 88, which refer to the TM-mode and TE-mode reflection responses, are not represented in the table.

Therefore, the integral is divided into subdomains. The parameter `nd` specifies how many subdomains. If this parameter is not specified, EMmod will assume `nd=1000`.

2.11 startlogx, deltalogx and nlogx

These three parameters define the coordinates in the space domain for which the Hankel transformation is carried out in the first iteration of the while loop of the forward modeling step. Thereby, `startlogx` specifies the logarithm of the coordinate of the first integration point greater than zero. The default value, which will be chosen if nothing is entered, is -6. This means, that the first coordinate is 10^{-6} meters. The parameter `deltalogx` specifies the logarithmic sampling rate. The default value is 0.025. Finally, the parameter `nlogx` defines the amount of points of this coordinate vector. The default value is 512. It should be ensured, that the largest point of this coordinate vector is larger than the largest radial distance in the grid specified with the parameters `dx`, `nx`, `dy` and `ny`.

2.12 kmax

This parameter defines the upper boundary of the integral of the Hankel transformation in the forward modeling step. If the electromagnetic field in the wavenumber domain is large also at large wavenumbers, for example if the vertical distance between the source and the receivers is small or if the source and the receivers are close to an interface, this integration boundary has to be shifted to larger wavenumbers. It is crucial to also increase accordingly the parameter `nd`. Naturally, if the complete integral spreads over a larger amount of wavenumbers, it needs to be divided into a larger amount of subdomains `nd`. The default value is `kmax=0.628625` m^{-1}, which is twice the Nyquist wavenumber for a spatial sampling rate of 10 m.

2.13 c1 and c2

These two parameters define the maximum allowed error in the forward modeling step for the optimization process searching for the points in the space domain for which the Hankel transformation has to be evaluated. These two parameters can be computed using the following Matlab script:

```
clear all; close all; clc;

xdata = [-6 -10.5 -15]; % logarithm of base 10 of the field-amplitude
ydata = [0.0025 0.005 0.01]; % maximum allowed error
eps = 10^(-6); % Stabilization parameter for the line-fitting process
fs = 18; % Fontsize
lw = 2; % Linewidth

% Find the two parameters of the linear regression
A(:,1) = xdata.';
```

```
A(:,2) = ones(size(xdata.'));
c = (A'*A+eps*eye(2,2))\A'*ydata.';

% Compute the function based on the two parameters previously determined
xvec = linspace(min(xdata),max(xdata),100);
yvec = c(1).*xvec+c(2);

figure;
plot(xdata,ydata*100,'ok','Linewidth',lw);
hold on;
plot(xvec,yvec*100,'k','Linewidth',lw);
hold off;
grid on
xlabel('log_{10} of Field-Amplitude','Fontsize',fs)
ylabel('Largest allowed error [%]','Fontsize',fs)
if sign(c(2)) == 1
    tempstring = '+';
else
    tempstring = '-';
end
c2string = [tempstring,' ',num2str(abs(c(2)))];
title(['y = ',num2str(c(1)),'x ',c2string],'Fontsize',fs)
legend('datapoints','determined function')
set(gca,'Fontsize',fs)
print('-depsc2','error_model.eps')
```

The array `xdata` of this Matlab script defines the logarithmic amplitude of the electromagnetic field for which the maximum allowed error is specified in the array `ydata`. The Matlab script fits a line through the specified points. The iEMmod parameters `c1` and `c2` define the slope and the offset of this linear function, respectively. Figure 1 shows the plot produced by the Matlab script printed above. The circles are the entered points with the required precision and the red line is the function found to fit these plots. The equation with parameters `c1` and `c2` is given in the header of the plot. The idea behind the choice made here for the precision is, that a smaller relative error is required if the amplitude of the electromagnetic field is large and a larger relative error is allowed if the amplitude of the electromagnetic field is small. For a constant error, `c1` is set to 0.0 and `c2` to the desired error. For example for a constant error of 5%, `c2` is set to 0.005. If the error is chosen large, less points in the space domain are required and, therefore, the algorithm runs faster, but the forward modeled electromagnetic field is not so precise. If the error is chosen small, a very precise forward modeled electromagnetic field can be expected. However, it is also possible, that the desired accuracy can not be achieved and the algorithm iterates until the maximum amount of points specified with the parameter `maxpt` is reached. If no value is specified by the user, `c1` is set to 0.0 and `c2` is set to 0.01.

2.14 maxpt

Under certain conditions the forward modeling process does not stop requiring more and more points in the space domain. This can for example be the case for components with a vertical electric antenna involved. These components decay typically at smaller offsets into the numeric noise. If that is the case, the algorithm will only fit numeric noise and will, therefore, never reach the required precision in that area. This can also happen if the precision parameters `c1` and `c2` are chosen to strictly. For

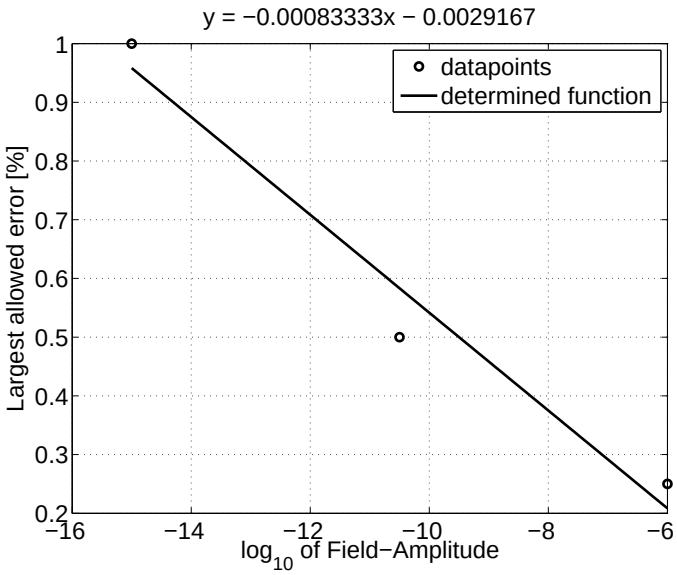

Figure 1: Error model: the circles are the datapoints for the required precision and the line is the determined function. The equation with parameters c1 and c2 is given above the figure.

these cases, the user can define a maximum amount of points for which the Hankel transformation is computed. If no value is specified by the user, `maxpt` is set to 500.

2.15 xdirect

In the forward modeling step, the direct field from the source to the receiver can either be computed in the wavenumber domain and numerically Hankel transformed to the space domain or it can be computed directly in the space domain. If it is computed in the wavenumber domain, the code is slightly faster, but it is then impossible to put the source and the receivers at the same depth. If that is the case or if the vertical distance between the source and the receivers is small, which requires a larger wavenumber spectrum to be integrated, it is recommend to compute the direct field in the space domain. Note, if both, the source and the receivers, are close to the same interface, it is still necessary to chose a large wavenumber spectrum for the integration even if the direct field is computed in the space domain, because the specular reflection features a high wavenumber content. `xdirect=1` computes the direct field in the space domain, `xdirect=0` computes it in the wavenumber domain. The default parameter is 1. It has been observed, that computing the direct field in the space domain can lead to artifacts in the final output, because the direct field is computed on a different precision level as the rest of the electromagnetic field.

2.16 whichstep

Once the direction for the next update of the conjugate gradient method has been fixed, the algorithm needs to determine how big the step is, that is taken in that direction. There exists a step length which leads to the smallest misfit. One can assume that the misfit as a function of the step length approximates a parabola. If the misfit is computed for three different step lengths, a parabola can be fitted through these three points. The step length used is at the minimum of the parabola and, thus, at the minimum misfit. If the parabola-assumption does not hold, it is better to search for the step length that minimizes the misfit through a line search. The parabola fitting requires a maximum of three forward simulations and is, therefore, faster than the line search, which might require many more forward simulations. However, the line search generally produces a better estimate of the ideal step length. If the parameter `whichstep` is set to 1, the step length is determined by fitting of a parabola. If the parameter `whichstep` is set to 2, a line search is carried out. The default is `whichstep=2`.

If the parabola fitting delivers a misfit that is larger than the previous misfit, the parabola-assumption obviously was not valid. The algorithm then switches automatically to the line search for the next iteration. Afterwards, parabola-fitting is employed again.

2.17 whichdir

This parameter specifies which method is used to determine the direction for the next model update. If the parameter `whichdir` is set to 1, the Fletcher-Reeves formula is employed. If the parameter `whichdir` is set to 2, the Polak-Ribière formula is used. For more information about the determination of the direction, the user is refered to Shewchuk (1994).

2.18 eps

The parameter `eps` defines a threshold for the model update. If the change of the conductivity between two iterations is smaller than the value specified in `eps`, the algorithm has possibly converged. But it could also be, that the direction was not good. Therefore, if this happens, a stop counter is increased and the search direction of the next iteration is determined by using the negative gradient. The whole inversion algorithm stops if the stop counter is larger than the parameter `ngradrestart`. The default value is `eps=1e-5`.

2.19 maxit

This parameter specifies the maximum amount of iterations allowed. The inversion algorithm will stop after this amount of iterations no matter if convergence has been reached or not. The default is `maxit=500`.

2.20 restartgrad

Normally the search direction is determined by one of the two methods specified with the parameter `whichdir`. However, from time to time a restart of the search direction is required, which means that the search direction is set to the negative gradient. This parameter gives the amount of iterations after which the search direction is regularly restarted. Default is `restartgrad=5`.

2.21 ngradrestart

This parameter fixes the amount of extraordinary restarts of the gradient before the program stops. In other words, it specifies how many times the model update is allowed to be smaller than the value specified in the parameter `eps`. The default is `ngradrestart=4`.

2.22 epsa

If the parameter `whichstep` is set to 2, the step length is determined using a line search. The parameter `epsa` specifies the threshold to stop this line search. To be preciese, if the change in the step length is smaller than `epsa`, the line search is stopped and the best step length found is used for the next update of the conductivities. Default is `epsa=1e-5`.

2.23 maxita

If the parameter `whichstep` is set to 2, the step length is determined using a line search. This line search either stops when the update of the step length is smaller than the parameter `epsa` or if the maximum amount of iterations specified by the parameter `maxita` have been reached. The default is `maxita=20`.

2.24 maxnogood

If the misfit during the line search for the step length has not improved for the amount of iterations specified by `maxnogood`, the line search is stopped and the previously best step length is selected. The default is `maxnogood=5`.

3 Load and plot results in Matlab

In the folder specified by the parameter `path_out` a binary file is created for each completed iteration. This file contains as many 64-bit real numbers with little-ending byte ordering as there are layers in the model plus one additional number. These numbers are the conductivity for each layer plus the misfit that correspond to this model. The number in the filename corresponds to the iteration number.
To load and plot the data in these files, the user can run the included Matlab script `load_iemmod_ref.m`. This script allows to plot the convergence history up to the latest iteration or up to a specified iteration. Also the corresponding subsurface conductivity model can be plotted together with the true model and the starting model.
If a Windows-system is used, the folder (line 12), where the files to load are located, must be specified using backward slashes and not forward slashes as it is the case on Linux-based systems.

4 The tool ascii2bin

The program iEMmod reads binary files as input, which are the data to be inverted as well as the weighting files. These files have the same format as the binary output of EMmod. As SEG does not accept binary files as a part of a software package, these files first need to be created from ASCII files included in the package. Therefore, we supplied a small program called ascii2bin, with which you can convert the included ASCII files to binary files on your machine.
First compile the tool by typing the following in the Unix console:
`gcc ascii2bin.c -DDOUBLE -o ascii2bin`
The flag `-DDOUBLE` ensures that double precision is used. Now you can use the program ascii2bin by typing `./ascii2bin file_in number_of_elements file_out`. The first argument, `file_in`, is the input ASCII file, which you would like to convert to a binary file. The second argument, `number_of_elements`, specifies how many elements have to be read from the ASCII file and subsequently written to the binary file. If the dataset has 512 times 512 datapoints, as it is the case for the example, this parameter is 262144. The third and final argument, `file_out`, is the name of the output binary file. All three arguments are required.

5 List of files

The following list of files more or less follows the logical structure of the program.

- `iemmod.c`: Main file containing input and output of data, but also the non-linear conjugate-gradient algorithm.
- `emmod.c`: Gateway between iemmod.c and kxwmod.c doing two things: 1.) Check if the medium is a homogeneous fullspace or a layered medium and 2.) determine in which layer the source and the receivers are located.
- `kxwmod.c`: Computes the forward-problem and the gradient for the various source-receiver combinations. This file includes the computation of the electromagnetic fields or the reflection responses in the wavenumber domain as well as the Hankel-transformation to the space domain.
- `zqk61_setup_grid.F90`: To determine the points that are necessary to evaluate an integral with the function zqk61n.f.

- `Gamma.F90` and `dGamma.F90`: Compute the vertical wavenumber Γ and its derivative, respectively. For mathematical details see Hunziker et al. (2015).

- `Wprop.F90` and `Wpropderiv.F90`: Calculate the propagation of the field or the gradient form the layer boundaries to the receiver position (Hunziker et al., 2015).

- `Rmin.F90`, `Rminderiv.F90`, `Rplus.F90` and `Rplusderiv.F90`:
  Compute the reflection response or its gradient (Hunziker et al., 2015).

- `Pdownmin.F90` and `Pdownminderiv.F90`, `Pdownplus.F90` and `Pdownplusderiv.F90`:
  Calculate the downgoing field or gradient (Hunziker et al., 2015).

- `Pupmin.F90` and `Pupminderiv.F90`, `Pupplus.F90` and `Pupplusderiv.F90`:
  Calculate the upgoing field or gradient (Hunziker et al., 2015).

- `Ptotalx.F90`, `Ptotaly.F90`, `Ptotalz.F90`, `Ptotalref.F90`,
  `dPtotalx.F90`, `dPtotaly.F90`, `dPtotalz.F90` and `dPtotalref.F90`:
  Combine the downgoing and upgoing parts of the field or the gradient with the propagators within a layer and the reflection coefficients to get the complete field or gradient in the wavenumber domain.

- `getcoords.F90`: Set up logarithmic coordinate vector in the space domain.

- `hankeltrans_xx.F90`, `hankeltrans_yy.F90`, `hankeltrans_zz.F90` and `hankeltrans_ref.F90`: Perform the Hankel-transformation for different components.

- `evalpoints.F90`: Evaluate which points are necessary in the space domain to achieve the precision defined with the parameters c1 and c2.

- `Ginxy.F90` and `dGinxy.F90`: The letters x and y can be 1, 2 or 3. This functions compute the direct field and its derivative, respectively, in the space domain for the component specified by x and y.

- `gridit_xx.F90`, `gridit_yy.F90`, `gridit_zz.F90`, `gridit_ref.F90`,
  `dgridit_xx.F90`, `dgridit_yy.F90`, `dgridit_zz.F90` and `dgridit_ref.F90`:
  Compute the space-domain fields or gradients on a 2D grid using symmetry relations and the corresponding data on a line.

- `bessel.c`: Wrapper for the Bessel-functions

- `verbosepkg.c`: Functions for printing messages to the screen.

- `wallclock_time.c`: Functions to measure CPU time.

5.1 Seismic Unix Files

- `atopkge.c`

- `docpkge.c`

- `getpars.c`

5.2 Slatec Files

- `zqk61n.f`: Compute an integral using the 61-point Gauss-Kronrod rule.
- `fdump.f`
- `i1mach.f`
- `j4save.f`
- `xercnt.f`
- `xerhlt.f`
- `xermsg.f`
- `xerprn.f`
- `xersve.f`
- `xgetua.f`

References

Hunziker, J., J. Thorbecke, and E. Slob, 2015, The electromagnetic response in a layered VTI medium: A new look at an old problem: Geophysics, **80**, F1–F18.

Shewchuk, J. R., 1994, An Introduction to the Conjugate Gradient Method Without the Agonizing Pain: School of Computer Science Carnegie Mellon University Pittsburgh, http://www.cs.cmu.edu/˜quake-papers/painless-conjugate-gradient.pdf.