E3D

A Program Library for Forward Modelling and Inversion of Frequency Domain EM Data over 3D Structures

Version 1.0

Developed under the consortium research project:

COOPERATIVE INVERSION OF GEOPHYSICAL AND GEOLOGICAL DATA

Release date: 7 September 2012

UBC - Geophysical Inversion Facility 1988 – 2012

Table of Contents

1 Package overview

1.1 Introduction

This manual provides instruction and background for the e3d program library for the forward modelling and inversion of frequency domain electromagnetic survey data. In order to decrease computational time and increase accuracy by mesh refinement in areas of interest, e3d models are discretized on an Octree mesh. For ease of use the program library includes several utilities

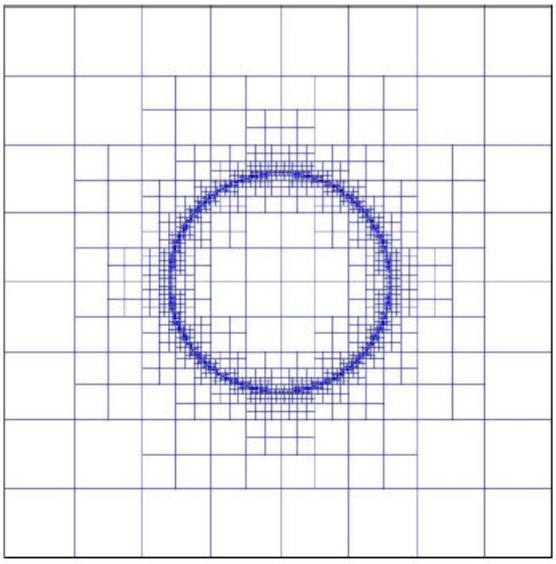


Figure 1: An example of 2D adaptive mesh refinement.

which generate a regular base mesh, enabling the user to construct initial or simulation models on a regular mesh and then convert to an octree mesh. From the users point of view the software operates in much the same way as previous GIF codes. This version is currently run through the command line only.
The program library provides codes to do the following:

1. Construct models on a rectangular mesh, where each cell is assigned a constant value of conductivity, and transfers the model to an octree mesh.
2. Forward model electric and magnetic field anomaly responses to a 3D volume of contrasting conductivity, on and octree mesh.
3. Convert from and octree mesh to a regular base mesh.
4. Invert of surface, airborne, and/or borehole EM data to generate 3D conductivity models:
    - The inversion is solved as an optimization problem with the simultaneous goals of (i) minimizing an objective function dependent on the model and (ii) generating synthetic data that match observations to within a degree of misfit consistent with the statistics of those data.

- To counteract the inherent lack of information, the formulation incorporates reference model and smoothing by regularization.
- Capacity for the user to directionally weight smoothing and reference model influence as well as overall influence of regulariztion on objective function minimization. Explicit prior information may also take the form of upper and lower bounds on the conductivity contrast in any cell.
- The regularization parameter (controlling relative importance of objective function and misfit terms).

The initial research underlying this program library was funded principally by the mineral industry consortium "Joint and Cooperative Inversion of Geophysical and Geological Data" (1991 - 1997) which was sponsored by NSERC and the following 11 companies: BHP Minerals, CRA Exploration, Cominco Exploration, Falconbridge, Hudson Bay Exploration and Development, INCO Exploration & Technical Services, Kennecott Exploration Company, Newmont Gold Company, Noranda Exploration, Placer Dome, and WMC.

Since then, improvements have been implemented as time and resources permit.

1.2 e3d program library content

1. **e3d executable programs**. For performing 3D forward modelling and inversion of EM data on an octree mesh.
   The e3d library (Windows or Linux platforms) consists of three major programs:
   - create_octree_mesh_e3d
   - e3dfwd: Forward Problem
   - e3dinv: Inverse Problem
2. **Octree utilities**. A collection of utilities to created weight files, generate models on the base regular mesh and convert between regular and octree meshes.
   - create_weight_file
   - face_weights
   - octree_cell_centre
   - octreeTo3D
   - refine_octree
   - remesh_octree_model
   - surface_electrodes
3. **graphical user interface**. There is not yet a GUI for e3d, but other GUI's are available for the Windows platforms **only**. Facilities include:
   - EH3Dinv-gui: this can be used to view data, or the fields generated by the forward problem.
   - MESHTOOLS3D: a utility for displaying resulting 3D models as volume renderings.
4. **Example data sets** are provided in "EXAMPLES" directory.

1.3 Licensing

A **constrained educational version** of the program is available with the IAG package (please visit UBC-GIF website for details). The educational version is fully functional so that users can learn how to carry out effective and efficient 3D inversions of magnetic data. **However, RESEARCH OR COMMERCIAL USE IS NOT POSSIBLE because the educational version will NOT work with more than 200 data points or 12,000 cells in the 3D mesh**.

Licensing for an unconstrained academic version is available - see the Licensing policy document.

NOTE: All academic licenses will be **time-limited to one year**. You can re-apply after that time. This ensures that everyone is using the most recent versions of codes.

Licensing for commercial use is managed by distributors, not by the UBC-GIF research group. Details are in the Licensing policy document.

1.4 Installing e3d

There is no automatic installer currently available for the e3d. Please follow the following steps in order to use the software:

1. Extract all files provided from the given zip-based archive and place them all together in a new folder.
2. Add this directory as new path to your environment variables.
3. If you are running the software on a cluster of computers, please install the Message Pass Interface (MPI) on your computer and add it to your path in addition from
4. Make sure to create a separate directory for each new inversion, where all the associated files will be stored. Do not store anything in the bin directory other than executable applications and Graphical User Interface applications (GUIs).

MPI can be downloaded here.

2 Theoretical Background for e3d

This section aims to provide the user with a basic review of the physics, discretization, and optimization techniques used to solve the frequency domain electromagnetics problem. It is assumed that the user has some background in these areas. For further reading see (Nabighian (2006)).

2.1 Physics

Maxwell's equations provide the starting point from which an understanding of how electromagnetic fields can be used to uncover the substructure of the Earth by determining changes it's electrical properties. In the frequency domain Maxwell's can be written as as:

$$\nabla \times \mathbf{E} + i\omega\mu\mathbf{H} = 0 \tag{1a}$$

$$\nabla \times \mathbf{H} - \sigma\mathbf{E} = s \tag{1b}$$

$$\widehat{\mathbf{n}} \times \mathbf{H} = 0 \tag{1c}$$

where $\mathbf{E}$ and $\mathbf{H}$ are the electric and magnetic fields, s is some external source, and μ, σ, and ω are the magnetic permeability, conductivity, and angular frequency respectively. This formulation assumes a quasi-static mode so that the system can be viewed as a diffusion equation (Weaver,1994; Ward and Hohmann,1988 in (Nabighian (2006)). By doing so some difficulties arise when solving the system;

- the curl operator has a non-trivial null space making the resulting linear system highly ill-conditioned
- the conductivity σ varies over several orders of magnitude
- the fields can vary significantly near the sources, but smooth out at distance thus high resolution is required near sources

These issues are addressed by considering a Helmholz potential formulation to deal with the null space of the curl operator, harmonic averaging and mesh refinement to combat large jumps in conductivities, and the ability to refine the mesh near the sources to improve resolution.

2.2 Octree mesh

By using an Octree discretization of the earth domain, the areas near sources and likely model location can be give a higher resolution while cells grow large at distance. In this manner, the necessary refinement can be obtained without added computational expense. Figure(2) shows an example of an OcTree mesh, with nine cells, eight of which are the base mesh minimum size When working with OcTree meshes, the underlying mesh is defined as a regular 3D orthogonal grid where the number of cells in each dimension are $2^{m_1} \times 2^{m_2} \times 2^{m_3}$, with grid size h. This underlying mesh is the finest possible, so that larger cells have lengths which increase by powers of 2 multiplied by h. The idea is that if the recovered model properties change slowly over a certain volume, the cells bounded by this volume can be merged into one without losing the accuracy in modelling, and are only refined when the model begins to change rapidly.

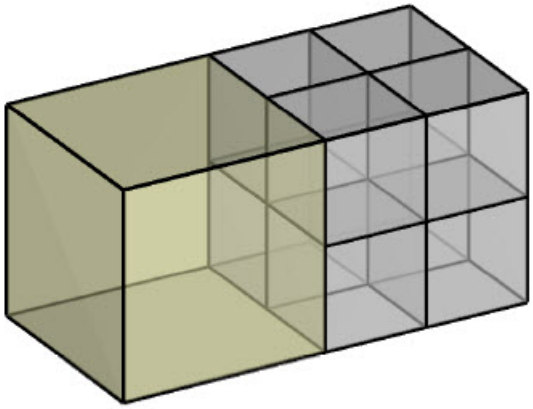


Figure 2: An example of a 3D OcTree mesh.(Haber et al. (2012))

2.3 Discretization of the Operators

The operators div, grad, and curl are discretized using a finite volume formulation. Although div and grad do not appear in (1), they are required for the solution of the system. The divergence operator is discretized in the usual flux-balance approach, which by Gauss' theorem considers the current flux through each face of a cell. The nodal gradient (operates on a function with values on the nodes) is obtained by differencing adjacent nodes and dividing by edge length. The discretization of the curl operator is computed similarly to the divergence operator by utilizing Stokes theorem by summing the magnetic field components around the edge of each face. Please see (Haber et al. (2012)) for a detailed description of the discretization process.

2.4 Solutions to the Forward Problem

The solution for the fields $\mathbf{H}$ and $\mathbf{E}$ can either be computed iteratively or directly depending on the number of sources. If the number of sources is small than an iterative method (BiCGstab) is used. Because of the null space of the curl operator a discrete Helmholz decomposition is used to write the electric field as

$$\mathbf{E} = \mathbf{A} + \nabla\phi$$

The problem is then solved by eliminating the curl operator and solving for $\mathbf{A}$ and ϕ.

If on the other hand if there are many sources, it is more efficient to directly decompose the forward matrix system by $\mathbf{LU}$ factorization. By doing so, many systems can be solved with a single factorization. MUMPS (Amestoy et al. (2001)) is used for the factorization and can be downloaded here.

The forward problem of simulating data can now be written in the following form. Let $\mathbf{A}(\mathbf{m})$ be the discrete linear system obtained by the discretiztion of Maxwell's equations, where $\mathbf{m} = \log(\sigma)$. Assuming there n_s sources $\mathbf{S} = i\omega(\mathbf{s}_1, ..., \mathbf{s}_{n_s})$ and that $\mathbf{P}^T$ is a projection matrix from edges to

receivers then the simulated EM field data can be written as

$$\mathbf{D}(\sigma) = \mathbf{P}^T \mathbf{A}(\mathbf{m})^{-1} \mathbf{S} \tag{2}$$

2.5 Solutions to the Inverse Problem

Solving the non-linear EM inverse problem for electric conductivity is akin to minimizing the following objective function

$$\mathcal{J}_{mis}(\mathbf{m}) = \frac{1}{2} ||\Sigma \odot (\mathbf{D}(\sigma) - \mathbf{d}^{\text{obs}})||_F^2$$

where Σ is a matrix of the inverse standard deviation for each measured data point $\mathbf{d}^{\text{obs}}$. Due to the ill-posedness of the problem, there are no stable solutions and a regularization is needed. The regularization used penalizes for both smoothness, and likeness to a reference model $\mathbf{m}_{\text{ref}}$ supplied by the user.

$$\mathcal{J}_{reg}(\mathbf{m} - \mathbf{m}_{\text{ref}}) = \frac{1}{2} ||\nabla(\mathbf{m} - \mathbf{m}_{\text{ref}})||_2^2$$

An important consideration comes when discretizing the regularization. The gradient operates on cell centered variables in this instance. Applying a short distance approximation is second order accurate on a domain with uniform cells, but only $\mathcal{O}(1)$ on areas where cells are non-uniform. To rectify this a higher order approximation is used Haber et al. (2012). The discrete regularization operator can then be expressed as

$$\begin{aligned} \mathcal{J}_{reg}(\mathbf{m}) &= \frac{1}{2} \int |\nabla m|^2 dV \\ &\approx \alpha \frac{1}{2} \mathbf{m}^T \mathbf{G}_c^T \text{diag}(\mathbf{A}_f^T v) \mathbf{G}_c \mathbf{m} \end{aligned}$$

where $\mathbf{A}_f$ is an averaging matrix from faces to cell centres, $\mathbf{G}$ is the cell centre to cell face gradient operator, and v is the cell volume For the benefit of the user, let WTW be the weighting matrix given by

$$\text{WTW} = \alpha \mathbf{G}_c^T \text{diag}(\mathbf{A}_f^T v) \mathbf{G}_c$$

which can be also be written as

$$\text{WTW} = \begin{pmatrix} \alpha_{\text{x}} & & \\ & \alpha_{\text{y}} & \\ & & \alpha_{\text{z}} \end{pmatrix} \begin{pmatrix} \mathbf{G}_{\text{x}}^T & \mathbf{G}_{\text{y}}^T & \mathbf{G}_{\text{z}}^T \end{pmatrix} \text{diag}(v_f) \begin{pmatrix} \mathbf{G}_{\text{x}} \\ \mathbf{G}_{\text{y}} \\ \mathbf{G}_{\text{z}} \end{pmatrix}$$

In the code the WTW matrix is stored as a separate matrix so that individual model norm components can be calculated . Now, if a cell weighting is used it is applied to the entire norm, that is, there is a w for each cell.

$$\text{WTW} = \text{diag}(w) \text{WTW} \text{diag}(w)$$

There is also the option of choosing a cell interface weighting. This allows for a weight on each cell FACE. The user must supply the weights (w_x, w_y, w_z) for each weighted cell. When the interface

weighting option is chosen and the value is less than 1, a sharp discontinuity will be created. When the value is greater than 1, there will be a smooth transition. To prevent the inversion from putting "junk" on the surface, the top X and Y face weights should have a large value.

$$\text{WTW} = \alpha_{\text{x}}\mathbf{G}_{\text{x}}^T\text{diag}(w_x v_f)\mathbf{G}_{\text{x}} + \alpha_{\text{y}}\mathbf{G}_{\text{y}}^T\text{diag}(w_y v_f)\mathbf{G}_{\text{y}} + \alpha_{\text{z}}\mathbf{G}_{\text{z}}^T\text{diag}(w_z v_f)\mathbf{G}_{\text{z}} + \alpha_{\text{s}}\text{diag}(w_s v_f)$$

The resulting optimization problem is then

$$\begin{aligned} \min_m \quad & \mathcal{J}_{mis}(\mathbf{m}) + \alpha\mathcal{J}_{reg}(\mathbf{m} - \mathbf{m}_{\text{ref}}) \\ \text{s.t.} \quad & \mathbf{m}_L \leq \mathbf{m} \leq \mathbf{m}_H \end{aligned}$$

where α is a regularization parameter, and $\mathbf{m}_L$ and $\mathbf{m}_H$ are upper and lower bounds provided by some a prior geological information.
A simple Gauss-Newton optimization method is used to where the system of equations is solved using ipcg (incomplete preconditioned conjugate gradients) to solve for each G-N step. For more information refer again to Haber et al. (2012) and references therein.

3 Program Library

This section provides a description of each program in the e3d library.

1. **e3d Executable programs**

   - create_octree_mesh_e3d: creates an octree mesh around a source-receiver array
   - e3dfwd: Forward Problem: computes electric and magnetic response to a 3D conductivity model
   - e3dinv: Inverse Problem: computes a conductivity model by inverting EM data (fields)

1. **Octree utilities**.

   - create_weight_file: creates the weighting on each cell in the model
   - face_weights: creates weights on the faces of cells
   - octree_cell_centre: computes the cell centres of each octree cell
   - octreeTo3D: converts and octree mesh to a 3D base mesh
   - refine_octree: refine the octrees
   - remesh_octree_model: converts the octree model to base mesh
   - surface_electrodes: a utility to place surface electrodes

2. **GUI's**.

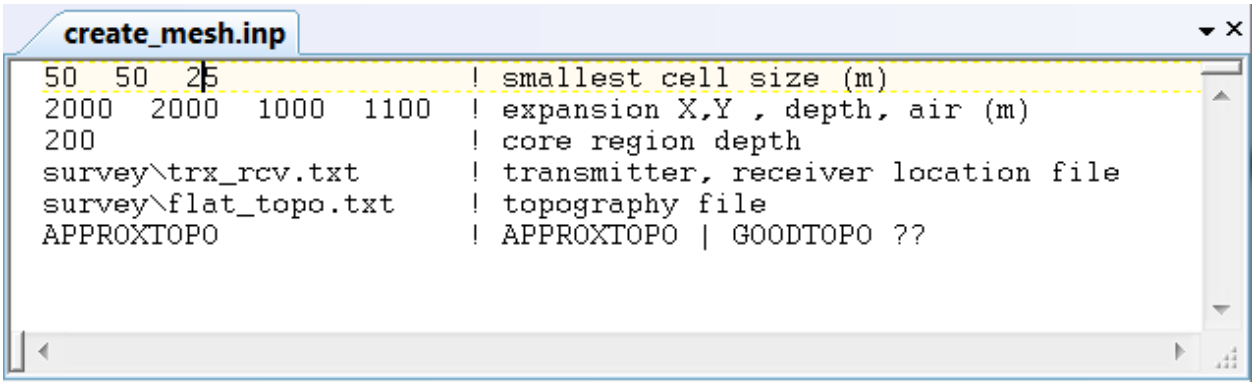
   - EH3Dinv-gui: this can be used to view data, or the fields generated by the forward problem at the surface
   - MESHTOOLS3D: a utility for displaying resulting 3D models as volume renderings.

3.1 OctTree Mesh Generation

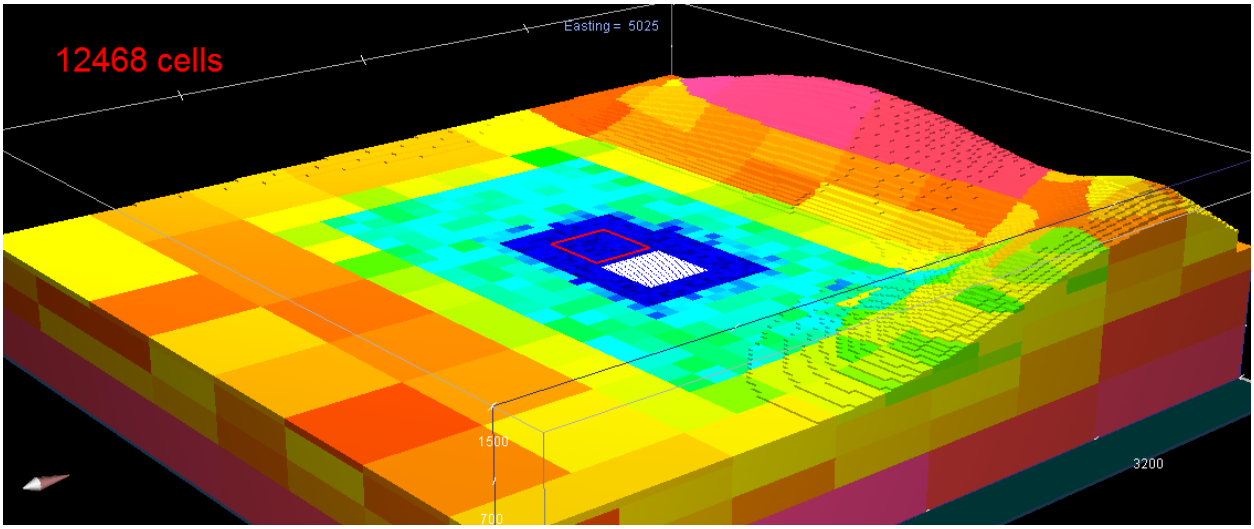
By providing a survey design, an OcTree mesh can be generated for the domain with the application **Create_octree_mesh_e3d.exe** where all input is specified in **Create_mesh.inp**. This file can be viewed and manipulated in a simple text editor.

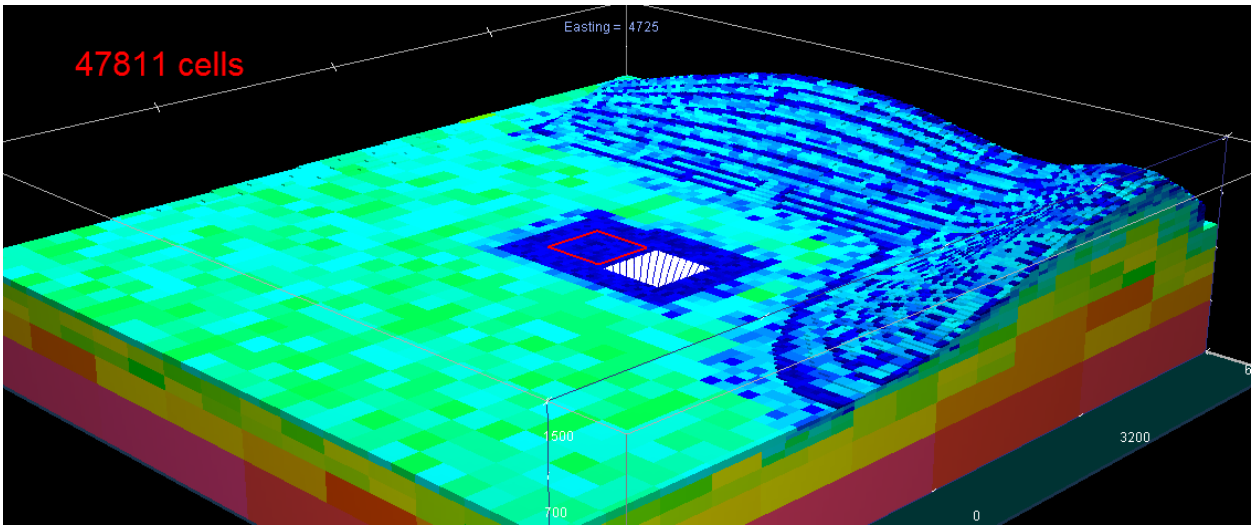
1. **Create_octree_mesh_e3d.exe**
   - **Create_mesh.inp**
     - trx_rcv.txt
     - flat_topo.txt
   - **Create_octree_mesh_e3d output files**
     - 3D_mesh.txt
     - 3D_core_mesh.txt
     - octree_mesh.txt
     - active_cells.txt
     - create_mesh.log

The OcTree mesh 'grows' from a fine regular base mesh to extents in x, y, z and elevation above ground, so the smallest cell size for the underlying mesh is specified in line 1 of **Create_mesh.inp** and the extents are specified in line 2. All cell lengths, extents, and depths are in metres (m). On line 3 the depth of the core region (region of interest) below the survey area can be specified. This area will retain the finest mesh cells, as accuracy in this area is important.



The pathway to the file **trx_rcv.txt** is specified on line 4 of the input file, and the pathway to the topography file, named **flat_topo.txt** in this example, is specified on line 5. These files contain information about the topography of the surface and location of transmitters and receivers. In the last line the user can decide whether to interpolate the topography information exactly or approximately. If APPROXTOPO is chose, there will only be fine cells close to the survey, whereas GOODTOPO will place fine cells everywhere on the surface. Pictured below are images of an approximated topography and fully discretized topography. Not the significant difference in number of cells between the two.





trx_rcv.txt contains all of the survey design information. The number of frequencies are specified on line 1. TRX_ORIG tells the code the number of corner points defining a closed loop transmitter and their coordinates (x,y,z). Next the frequency is set and the number of receivers are specified after N_RECV. The (x,y,z) coordinates of each receiver is then entered.

trx_rcv.txt

```
N_TRX 2

TRX_ORIG
5
-1650.0  -475.0  12.5
 -650.0  -475.0  12.5
 -650.0   475.0  12.5
-1650.0   475.0  12.5
-1650.0  -475.0  12.5

FREQUENCY 100

N_RECV 1681
         -500.000      -500.000          0.000
         -475.000      -500.000          0.000
         -450.000      -500.000          0.000
         -425.000      -500.000          0.000
         -400.000      -500.000          0.000
         -375.000      -500.000          0.000
         -350.000      -500.000          0.000
         -325.000      -500.000          0.000
         -300.000      -500.000          0.000
```

If there is more than one frequency, simply add another frequency and set of receiver points below the first.

flat_topo.txt contains all topography information. The first line gives the number of points of known elevation followed by the list of the elevations (x,y,z). For example if there is no available knowledge about topology all the z coordinates can be set to zero as in the following example

flat_topo.txt

```
4
-600 -600 0
-600  600 0
 600  600 0
 600 -600 0
```

Enter the following in the command line to generate the OcTree mesh.

```
>>C:\e3d> create_octree_mesh_e3d create\_mesh.inp
```

Create_octree_mesh_e3d.exe will output the following five files in the same directory. These files contain information about underlying meshes and serve as an input for forward modelling.

- **3D_mesh.txt** - regular underlying mesh, number of cells in each direction (powers of 2)
- **3D_core_mesh.txt** - the same as 3D_mesh.txt, summary for the core region

- **octree_mesh.txt** - octree mesh, list
- **active_cells.txt** - if value 1 is assigned the cell is active (defined on the OcTree mesh)
- **create_mesh/log** - log file

3.2 Forward Problem

Generating simulated EM field data from a prescribed conductivity model is a three step process. First a model is generated on a regular grid, converted to and OcTree mesh, and finally run in **e3dfwd.exe**. The following list contains all the input and output files required and generated in the three step process.

1. Create a conductivity model: **blk3cell.exe**
   - **blk3cell.inp**
     - 3D_core_mesh.txt
   - **blk3cell.exe output file** :
     - 3D_model.con (can be a different name)
2. Convert model to OcTree: **3DModel2Octree.exe**
   - **model2octree.inp**
     - octree_mesh.txt
     - 3D_core_mesh.txt
     - 3Dmodel.con
     - octree_mesh_block.txt (named in the input file)
     - block_octree.con (named in the input file)
   - **3DModel2Octree.exe output file** :
     - octree_mesh_block.txt
     - block_octree.con
3. Forward Code: **e3dfwd.exe**
   - **e3d_octree_fwd.inp**
     - octree_mesh_block.txt
     - trx_rcv.txt
     - block_octree.con
   - **e3dfwd.exe output file** :
     - e3d_data.txt
     - e3d_octree_fwd.log

3.2.1 Creating the model/ conductivity field

Creating the conductivity model is a two phase process, it is first defined on a standard 3D mesh then converted to an OcTree mesh. In the input file **blk3cell.inp** blocks of differing conductivity are defined. The value in line 1 specifies the background conductivity, and in line 2, the number of blocks. each of them is described in the following lines based on extent (x_{start} x_{end} y_{start} y_{end} z_{start} z_{end}) and conductivity value. The first block every time represents the background conductivity field.

blk3cell.inp

```
1.e-8  ! background air
2  ! number of blocks
-10000 10000  -10000 10000  0 -10000  0.005  ! xx, yy, zz, background conductivity
-125 125  -250 250  -50 -150  1              ! xx, yy, zz, block conductivity
```

This file together with the 3D_core_mesh.txt serve as an input for the application **blk3cell.exe**. The name of the output file for the model has to be given in the command line before running the application (it should have a suffix '.con', e.g. **3D_model.con** and will contain a list of all cell conductivities).

```
>>C:\e3d> blk3cell.exe 3D_core_mesh.txt blk3cell.inp 3Dmodel.con
```

3.2.2 Convert model to OcTree

To convert the conductivity model into an OcTree model run **3DModel2Octree.exe**, which has an input file **Model2Octree.inp**.

model2octree.inp

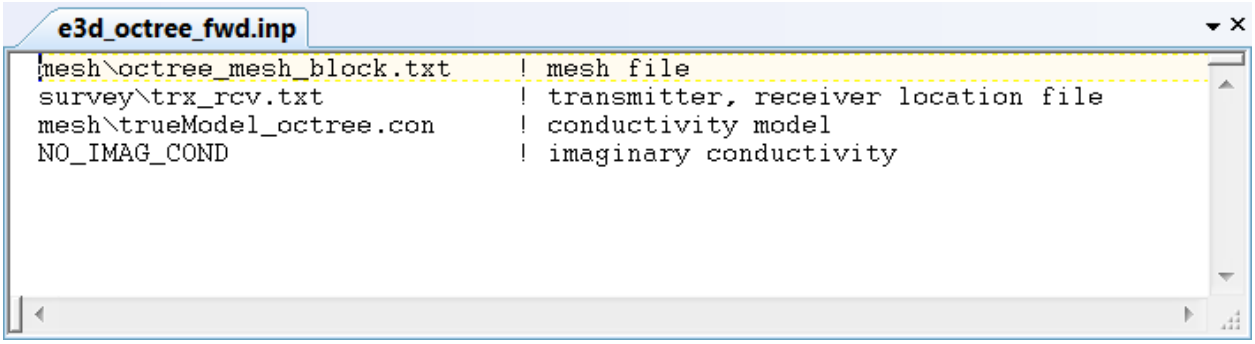
```
LOG_MODEL              ! LOG_MODEL | LIN_MODEL
input\octree_mesh.txt      ! input octree mesh
input\3D_core_mesh.txt     ! input 3d mesh file
input\trueModel.con          ! input 3D model
input\octree_mesh_block.txt ! USE_INPUT_MESH | output octree mesh
input\trueModel_octree.con     ! output octree model
```

This file specifies the paths to the names of the input and output files and in the first line the user can specify a logarithmic or linear conductivity model. The input files are **octree_mesh.txt**, **3D_core_mesh.txt**, and **3D_model.con**.

```
>>C:\e3d> 3DModel2Octree.exe model2octree.inp
```

3.2.3 The forward simulation

The input file **e3d_octree_fwd.inp** gives the names of the files for the OcTree mesh and model, transmitters and receivers location, last line gives a choice whether to enable imaginnary conductivity or not.



The command line:

```
>>C:\e3d> e3dfwd.exe e3d_octree_fwd.inp
```

The simulation results will be stored in

- **e3d_data.txt** - the 'observed' data (potentials)
- **e3d_octree_fwd.log** - log file of the simulation

3.3 Inverse Problem

The inversion of 3D EM data on an OcTree grid for electric conductivity is performed using **e3dinv.exe**. The input file and other required files are listed below. The user will need to generate a mesh based on the survey design, and make sure that their data file is in the correct format, including standard deviations. To generate an OcTree and base mesh for the survey, see section (3.1).

1. InverseCode: **e3dinv.exe**
    - **e3d_octree_inv.inp**
        - octree_mesh.txt
        - data.txt
        - active_cells.txt
        - interface_weights.txt
        - cell_weights.txt
    - **e3dinv.exe output file** :
        - inv.con
        - dpred.txt
        - e3d_octree_inv.log
        - e3d_octree_inv.out
2. Utilities
    - **blk3cell.exe**
    - **3DModel2Octree.exe**
    - **interface_weights.exe**

For the inversion all the necessary files and parameters are defined in a single input file **e3d_octree_inv.inp**. The user must provide a pathway to the OcTree mesh and data files in lines 1 and 2. The initial model and reference model conductivity values are specified in lines 3 and 4, or the user must provide pathways to files containing an initial and/or reference models if applicable (something more complex than a constant background value). When creating the OcTree mesh (section 3.1), **Create_octree_mesh_e3d.exe** will output the file **active_cells.txt**, which depends on topography. A pathway to this file is entered on line 5.

e3d_octree_inv.inp

```
../octree_mesh.txt      ! mesh file
../data.txt             ! data file
VALUE 0.005           ! initial model
VALUE 0.005           ! reference model
../active_cells.txt ! active cell file for topography | ALL_ACTIVE
ALL_ACTIVE            ! model active cell file | ALL_ACTIVE
NO_WEIGHT             ! cell weight file | NO_WEIGHT
interface_weights.txt     ! interface weighting file | NO_FACE_WEIGHT
200.  1.e-4    0.4  ! DEFAULT ! beta_max  beta_min  beta_factor
1.e-5  1.  1.  1.   ! alpha_s  alpha_x  alpha_y  alpha_z
1.                  ! chifact
1.e-2  1.e-3  3     ! tol_nl  mindm  iter_per_beta
1.e-2  20           ! tol_ipcg  max_iter_ipcg
CHANGE_MREF         ! CHANGE_MREF | NOT_CHANGE_MREF
SMOOTH_MOD_DIF      ! SMOOTH_MOD | SMOOTH_MOD_DIF
BOUNDS_NONE         !BOUNDS_CONST 1.e-10  1.e+10    ! bounds
```

The following is a breakdown of the inversion parameters specified on lines 6 through 16

- Line 6: either provide a pathway to a file specifying which cells in the model are active, or choose ALL_ACTIVE if no file is required
- Line 7: NO_WEIGHT: either provide a pathway to a file specifying cell weights, or choose NO_WEIGHT if there is no cell weighting
- Line 8: NO_FACE_WEIGHT: either provide a pathway to an interface weights file or choose NO_FACE_WEIGHT
- Line 9: BETA cooling parameter, either choose DEFAULT or enter the bounds; [beta_max, beta_min] and the decrease rate beta_factor.
- Line 10: Alpha parameters (section 2.5). User specifies 'how much' each smoothing and/or reference model in each direction contributes to the inversion
- Line 11: chifactor
- Line 12: tol_nl (newton iteration tol, how close the gradient is to zero) mindm(minimum model perturbation δm allowed) iter_per_beta (number of iterations per beta value)
- Line 13: tol_ipcg (now well the iterative solver does when solving for δm), max_iter_ipcg (maximum iterations of incomplete-preconditioned-conjugate gradient)
- Line 14: CHANGE_MREF: updates the reference model to the last inversion step result. Choose NOT_CHANGE_MREF if the reference model is to remain static
- Line 15: Choosing SMOOTH_MOD is essentially the same as have $\mathbf{m}_{ref} = 0$, If there is a reference model chose SMOOTH_MODEL_DIF for $(\mathbf{m} - \mathbf{m}_{ref})$.

- Line 16: Bound constraints on the recovered model: Choose BOUNDS_CONST and enter the values of the minimum and maximum model conductivity. Enter BOUNDS_NONE if the inversion is unbounded, or if there is no a-prior information about the subsurface model.

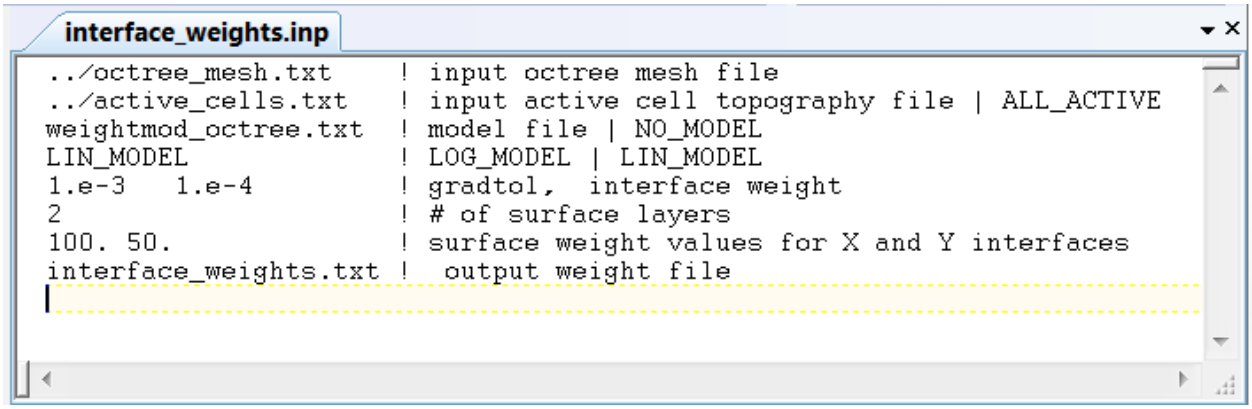
The inversion is executed through a command:

```
>>C:\e3d> e3dinv.exe e3d_octree_inv.inp
```

GENERATING INTERFACE WEIGHTS FILE:
If choosing to use an interface weighting, the file is generated in the following way

1. **Create weight interfaces on a 3D mesh.** The interface weights apply to a model (reference model or perhaps a fault).
2. **Convert to a 3D mesh**. Use **3DModel2Octree.exe** (see section 3.2.2)
3. **Create the interface weight file using interface_weights.exe**
   - interface_weights.inp
     - octree_mesh.txt
     - active_cells.txt
     - weightmod_octree.txt
   - interface_weights.inp OUTPUT
     - interface_weights.txt (can be a different name)

The input file for the interface weighting utility **interface_weights.exe** requires pathways to the mesh file **octree_mesh.txt** on line 1, and the file **active_cell.txt** on line 2. A model file can be specified on line 3, or if there is no model file where the weights are acting on a region of the mesh NO_MODEL is entered. On line 4 a linear of logarithmic model is specified.

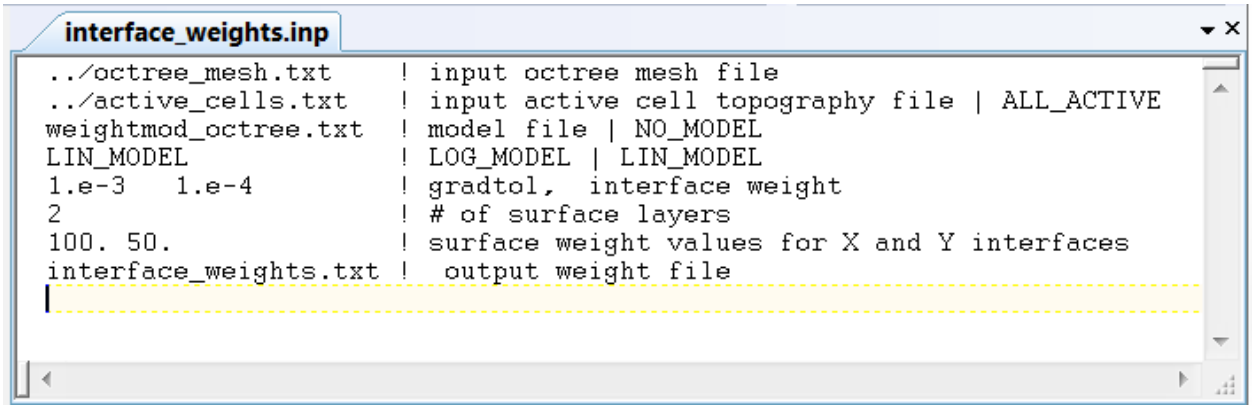


The gradient tolerance and interface weight value is entered on line 5. The weights are set to the interface weight value when $|\nabla \mathbf{m}| >$ gradient tolerance. If the interface weight value is less than 1,

a sharp discontinuity will be generated. If the value is greater than 1, then there will be a smooth transition. To prevent the inversion from putting 'junk' on the surface, the top X and Y interface weights should have a large value. The number of surface layers are entered on line 6. Adding more layers will result in smoother transitions. On line 7 the surface weight values are entered. The output file name is entered on line 8. **GENERATING CELL WEIGHTS FILE:**
If choosing do use an cell weighting, the file is generated in the following way

1. **Create weight interfaces on a 3D mesh.** The interface weights apply to a model (reference model or perhaps a fault).
2. **Convert to a 3D mesh**. Use **3DModel2Octree.exe** (see section 3.2.2)
3. **Create the interface weight file using interface_weights.exe**
   - interface_weights.inp
     - octree_mesh.txt
     - active_cells.txt
     - weightmod_octree.txt
   - interface_weights.inp OUTPUT
     - interface_weights.txt (can be a different name)

The input file for the interface weighting utility **interface_weights.exe** requires pathways to the mesh file **octree_mesh.txt** on line 1, and the file **active_cell.txt** on line 2. A model file can be specified on line 3, or if there is no model file where the weights are acting on a region of the mesh NO_MODEL is entered. On line 4 a linear of logarithmic model is specified.



The inversion output model can be viewed in MeshTools, and the predicted data in the EH3Dinv gui.

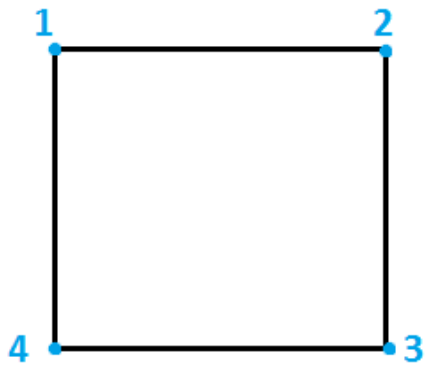
4 Example 1: Block in a half space

Folder: Demo. This is a simple block in a half space example designed to familiarize the user with the work flow required to simulate and invert data.

4.1 Octree mesh generation from a survey design

To generate a mesh to a transmitter/receiver configuration must be specified. This applies both to the forward and inverse problems.

Generate a mesh from transmitter-reciever file and topography
Define the location of a closed loop transmitter and a set of receivers in **trx_rcv.txt**, in this example there is one transmitter described in the first paragraph of the file: the three columns give x,y and z coordinates of each corner of the closed loop (1,2,3,4,1). The first frequency is set to 100Hz and it is followed by the number of receivers: 1681 and a list of their x,y,z coordinates.



Closed loop Transmitter

The topography file in this example it flat. This can be adjusted by adding z values in the example file or importing other data and converting to the format in flat_topo.txt.

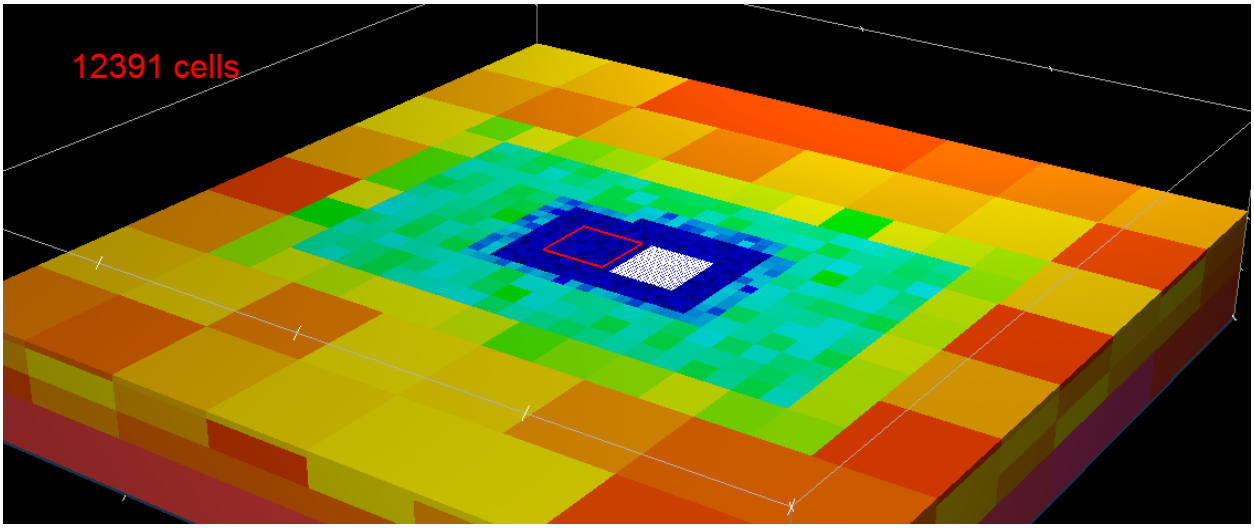
create_mesh.inp

```
50  50  25              ! smallest cell size (m)
2000  2000  1000  1100  ! expansion X,Y , depth, air (m)
200                     ! core region depth
survey\trx_rcv.txt      ! transmitter, receiver location file
survey\flat_topo.txt    ! topography file
APPROXTOPO              ! APPROXTOPO | GOODTOPO ??
```

Command line:

```
>>C:\...\Demo> create_octree_mesh_e3d create_mesh.inp
```

The output mesh files are created in the same directory. The mesh can be viewed with MeshTool3D. To overlay the transmitters and receivers, copy the x,y,z locations to a text file and import by loading data in MeshTools3D.



4.2 Forward problem

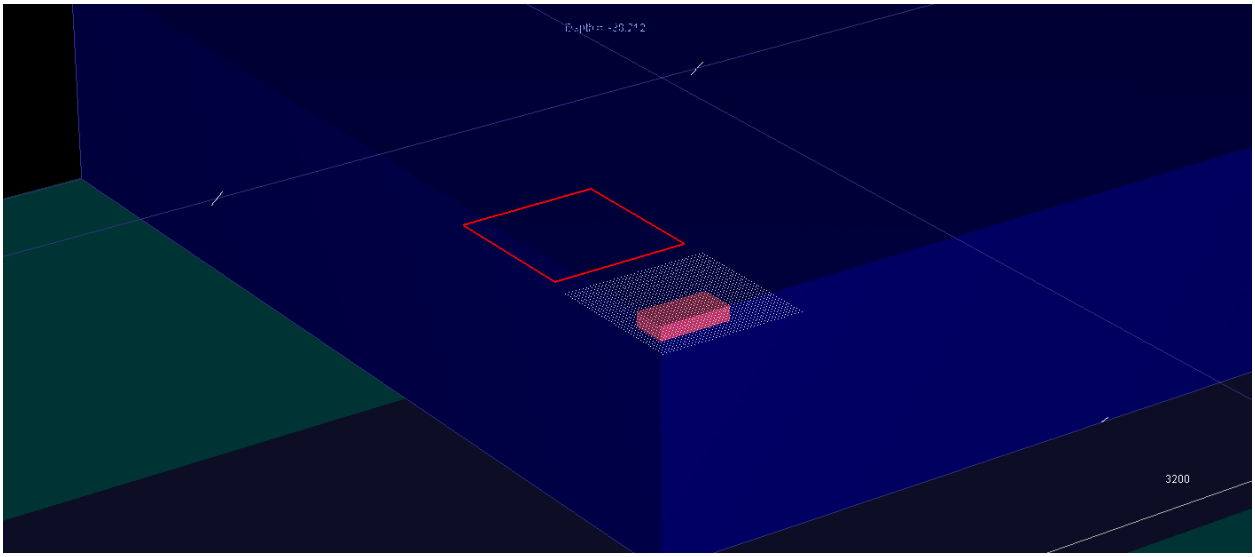
Create the conductivity model
Here **blk3cell.exe** is used to generate a block on the regular core mesh. In this example there is one block whose centre is at (0 ,0 ,-100) with conductivity = 1.

blk3cell.inp

```
1.e-8  ! background air
2  ! number of blocks
-10000 10000  -10000 10000  0 -10000  0.005  ! xx, yy, zz, background conductivity
-125 125  -250 250  -50 -150  1              ! xx, yy, zz, block conductivity
```

Command line:

```
>>C:\...\Demo> blk3cell.exe 3D_core_mesh.txt blk3cell.inp trueModel.con
```



Convert the model to OcTree
The model is now converted to an OcTree mesh from the regular mesh, using **3DModel2Octree.exe**, with input file **model2octree.inp**:

model2octree.inp

```
LOG_MODEL                ! LOG_MODEL | LIN_MODEL
input\octree_mesh.txt        ! input octree mesh
input\3D_core_mesh.txt       ! input 3d mesh file
input\trueModel.con            ! input 3D model
input\octree_mesh_block.txt ! USE_INPUT_MESH | output octree mesh
input\trueModel_octree.con      ! output octree model
```

Command line:

```
>>C:\Demo> 3DModel2Octree.exe model2octree.inp
```

4.3 Forward Model Data

There is one input file required for **e3d_octree_fwd.exe**, in which all input information is specified. Enter pathways in **e3d_octree_fwd.inp** to the mesh, transmitter/receiver file and the conductivity model.

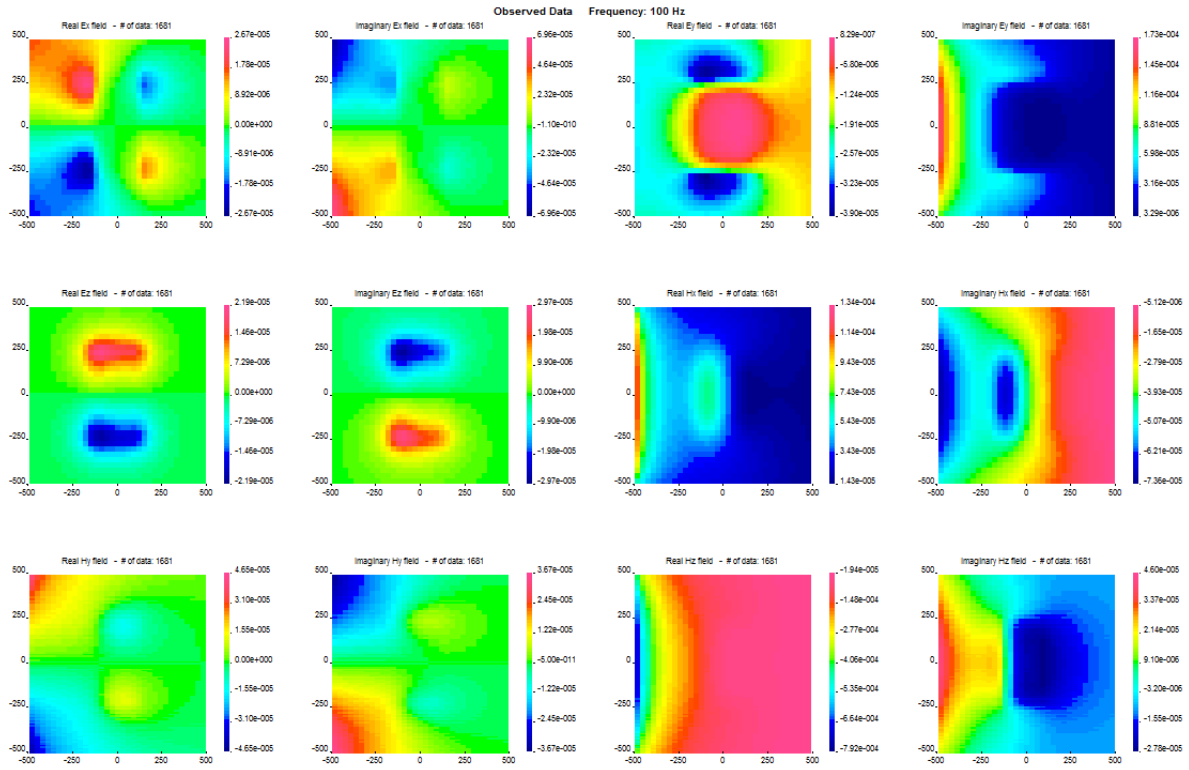
e3d_octree_fwd.inp

```
mesh\octree_mesh_block.txt    ! mesh file
survey\trx_rcv.txt            ! transmitter, receiver location file
mesh\trueModel_octree.con     ! conductivity model
NO_IMAG_COND                  ! imaginary conductivity
```

Command line:

```
>>C:\Demo> e3dfwd.exe e3d_octree_fwd.inp
```

The simulated data are easily viewed using the EH3Dinv GUI.



4.4 Inverse Problem

In this example we invert some data generated from the same block model. The mesh was generated a second time for the same survey so that it would differ from the forward modelling mesh, and no inverse crimes would be committed. The results of three inversions are shown, to demonstrate the effect that different types of weightings have on the inversion results. The data file has a specific structure, which includes the transmitter_receiver coordinates (see section 3.1)

data.txt

```
IGNORE -0

N_TRX 2

TRX_ORIG
5
 -1650.0 -475.0  12.5
  -650.0 -475.0  12.5
  -650.0  475.0  12.5
 -1650.0  475.0  12.5
 -1650.0 -475.0  12.5

FREQUENCY 10.     ! frequency
N_RECV 1681

!   x          y          z          Ex r            Ex r sd         Ex i            Ex i sd         Ey r            Ey r sd         Ey i

    -500.00    -500.00    0.00    -4.59332E-007    3.14241E-007    8.48348E-006    1.87255E-007    -6.42184E-007    3.76497E-007    9.85713
    -475.00    -500.00    0.00    -7.73329E-007    2.85517E-007    7.66974E-006    1.70844E-007    -5.95790E-007    3.54913E-007    9.94308
    -450.00    -500.00    0.00    -9.39969E-007    2.65492E-007    6.90609E-006    1.59406E-007    -9.93220E-007    3.33327E-007    9.10756
    -425.00    -500.00    0.00    -5.32715E-007    2.45468E-007    6.56149E-006    1.47976E-007    -8.94862E-008    3.16908E-007    8.72843
```

Also note that the real and imaginary field components are included and assigned a standard deviation.

4.4.1 Inversion with no weights

The input file for **e3dinv.exe** is described in detail in section 3.3. Here there are no weight files specified.

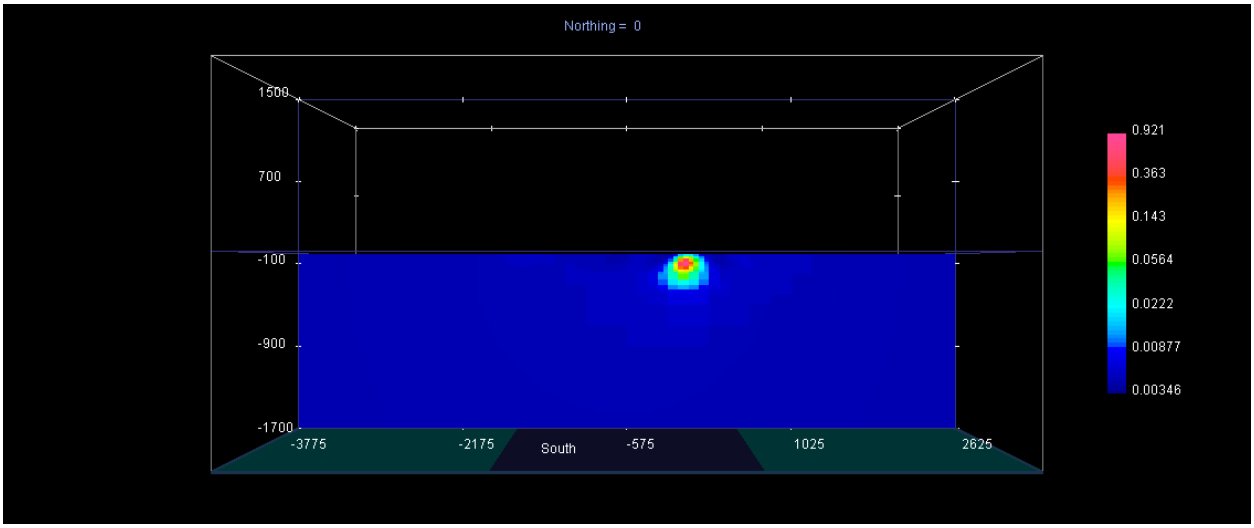
e3d_octree_inv.inp

```
octree_mesh.txt      ! mesh file
data.txt             ! data file
VALUE 0.005          ! initial model
VALUE 0.005          ! reference model
active_cells.txt     ! active cell file for topography | ALL_ACTIVE
ALL_ACTIVE           ! model active cell file | ALL_ACTIVE
NO_WEIGHT            ! cell weight file | NO_WEIGHT
NO_FACE_WEIGHT       ! interface weighting file | NO_FACE_WEIGHT
200.  1.e-4    0.4   ! DEFAULT ! beta_max  beta_min  beta_factor
1.e-5  1.  1.  1.    ! alpha_s  alpha_x  alpha_y  alpha_z
1.                   ! chifact
1.e-2  1.e-3  3      ! tol_nl  mindm  iter_per_beta
1.e-2  20            ! tol_ipcg  max_iter_ipcg
CHANGE_MREF          ! CHANGE_MREF | NOT_CHANGE_MREF
SMOOTH_MOD_DIF       ! SMOOTH_MOD | SMOOTH_MOD_DIF
BOUNDS_NONE          !BOUNDS_CONST 1.e-10  1.e+10    ! bounds
```

If more than one processor is to by used, the command line entry should include an instruction to mpi to use a certain number of processors:

```
C:\...\mpiexec.exe -localonly 4 -priority 1  e3dinv.exe
```

After 4 iterations the recovered model is shown in the following plot with the receiver locations overlaid in MeshTools3D.



4.4.2 Inversion with interface weights

The interface_weights file is created as explained in section 3.3. The only change to the **e3d_octree_inv.inp** file is the interface_weights.txt file

e3d_octree_inv.inp

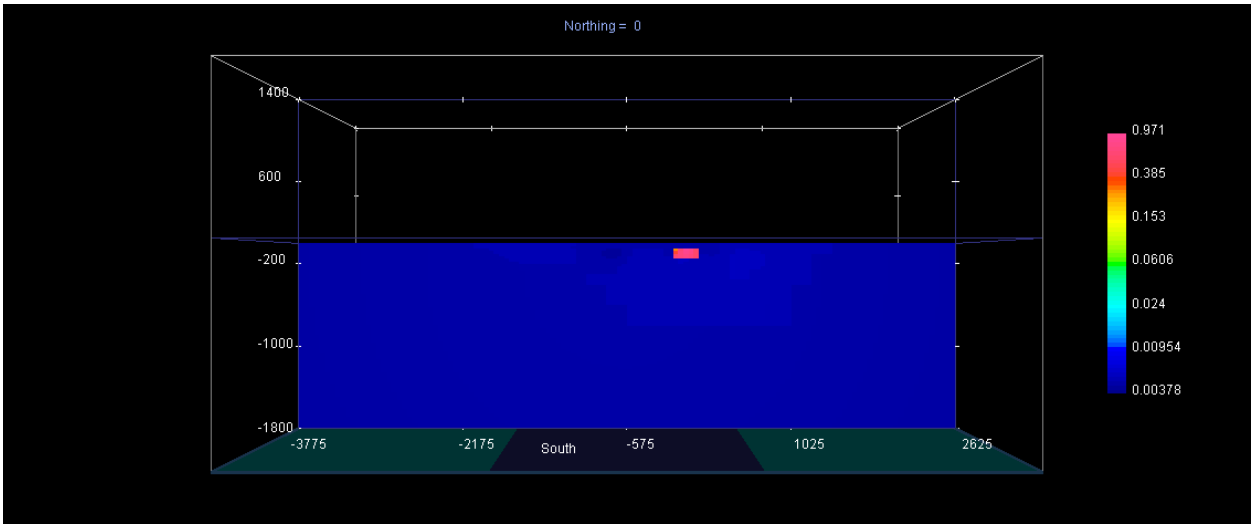
```
octree_mesh.txt      ! mesh file
data.txt             ! data file
VALUE 0.005          ! initial model
VALUE 0.005          ! reference model
active_cells.txt ! active cell file for topography | ALL_ACTIVE
ALL_ACTIVE           ! model active cell file | ALL_ACTIVE
NO_WEIGHT            ! cell weight file | NO_WEIGHT
interface_weights.txt    ! interface weighting file | NO_FACE_WEIGHT
200.  1.e-4    0.4   ! DEFAULT ! beta_max  beta_min  beta_factor
1.e-5  1.  1.  1.    ! alpha_s  alpha_x  alpha_y  alpha_z
1.                   ! chifact
1.e-2  1.e-3  3      ! tol_nl  mindm  iter_per_beta
1.e-2  20            ! tol_ipcg  max_iter_ipcg
CHANGE_MREF          ! CHANGE_MREF | NOT_CHANGE_MREF
SMOOTH_MOD_DIF       ! SMOOTH_MOD | SMOOTH_MOD_DIF
BOUNDS_NONE          !BOUNDS_CONST 1.e-10  1.e+10    ! bounds
```

This inversion result after 4 beta iterations has much sharper edges than the result without interface weights. The 4th beta iteration model is chosen because of a significant decrease in the relative gradient and conductivity value closest to the true model value of 1.

e3d_octree_inv.out

% beta	iter	misfit	phi_d	phi_m	phi	norm g	g rel
2.00000E+02	0	5.70127E+06	2.85064E+06	0.00000E+00	2.85064E+06	2.08964E+05	1.00000E+00
2.00000E+02	1	3.86229E+06	1.93115E+06	7.58542E+01	1.94632E+06	1.41586E+05	6.77562E-01
2.00000E+02	2	2.11020E+06	1.05510E+06	2.56257E+02	1.10635E+06	8.43018E+04	4.03428E-01
2.00000E+02	3	9.21872E+05	4.60936E+05	5.13439E+02	5.63624E+05	3.17912E+04	1.52138E-01
8.00000E+01	0	9.21872E+05	4.60936E+05	0.00000E+00	4.60936E+05	4.36714E+04	1.00000E+00
8.00000E+01	1	5.11725E+05	2.55863E+05	9.24084E+01	2.63255E+05	9.70428E+03	2.22211E-01
8.00000E+01	2	4.56861E+05	2.28430E+05	1.83220E+02	2.43088E+05	1.43861E+03	3.29416E-02
8.00000E+01	3	4.57283E+05	2.28642E+05	1.75513E+02	2.42683E+05	2.39977E+02	5.49505E-03
3.20000E+01	0	4.57283E+05	2.28642E+05	0.00000E+00	2.28642E+05	1.81627E+04	1.00000E+00
3.20000E+01	1	4.29236E+05	2.14618E+05	1.59510E+02	2.19722E+05	5.16529E+02	2.84390E-02
3.20000E+01	2	4.29211E+05	2.14606E+05	1.59184E+02	2.19700E+05	3.85686E+01	2.12351E-03
1.28000E+01	0	4.29211E+05	2.14606E+05	0.00000E+00	2.14606E+05	1.66714E+04	1.00000E+00
1.28000E+01	1	4.06773E+05	2.03386E+05	3.43499E+02	2.07783E+05	4.97569E+02	2.98456E-02
1.28000E+01	2	4.06492E+05	2.03246E+05	3.50645E+02	2.07734E+05	6.81934E+01	4.09043E-03
5.12000E+00	0	4.06492E+05	2.03246E+05	0.00000E+00	2.03246E+05	1.52881E+04	1.00000E+00
5.12000E+00	1	3.82803E+05	1.91402E+05	9.88323E+02	1.96462E+05	5.60893E+02	3.66881E-02
5.12000E+00	2	3.83092E+05	1.91546E+05	9.53769E+02	1.96429E+05	3.54811E+01	2.32082E-03
2.04800E+00	0	3.83092E+05	1.91546E+05	0.00000E+00	1.91546E+05	1.39505E+04	1.00000E+00
2.04800E+00	1	3.52557E+05	1.76278E+05	3.46502E+03	1.83375E+05	1.19990E+03	8.60113E-02
2.04800E+00	2	3.54265E+05	1.77133E+05	3.00227E+03	1.83281E+05	1.36544E+02	9.78779E-03
8.19200E-01	0	3.54265E+05	1.77133E+05	0.00000E+00	1.77133E+05	1.20529E+04	1.00000E+00
8.19200E-01	1	3.15767E+05	1.57884E+05	1.13136E+04	1.67152E+05	2.00815E+03	1.66611E-01
8.19200E-01	2	3.19821E+05	1.59910E+05	8.40506E+03	1.66796E+05	5.68297E+02	4.71501E-02
8.19200E-01	3	3.18586E+05	1.59293E+05	9.10180E+03	1.66749E+05	1.71547E+02	1.42328E-02

In the file **e3d_octree_inv.out** all information from the inversion is recorded for each beta value and iteration. When the value of beta gets small (after 10+ iterations) the inversion is fitting the noise in the data, and will return poor model estimates.



5 References

Amestoy, P., I. Duff, J.-Y. LExcellent, and J. Koster, 2001, Mumps: A general purpose distributed memory sparse solver, *in* Applied Parallel Computing. New Paradigms for HPC in Industry and Academia: Springer Berlin / Heidelberg, volume **1947** *of* Lecture Notes in Computer Science, 121–130. (10.1007/3-540-70734-4_16).

Haber, E., D. Oldenburg, R. Shekhtman, and C. Schwarzbach, 2012, An adaptive mesh method for electromagnetic inverse problems: Geophysics.

Nabighian, M. N., 2006, Electromagnetic methods in applied geophysics: Society of Exploration Geophysicists, **1**, no. 3.