DCIP3D OCTREE

A Program Library for Forward Modelling and Inversion of DC/IP data over 3D Structures using Octree meshes

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 DCIP3D OCTREE v1.0: Package overview

1.1 New capabilities

DCIP3D for octree meshes is a program library for carrying out forward modelling and inversion of DC resistivity and induced polarization data over 3D structures. The forward and inverse modeling is done on a pre-defined base (underlaying) mesh, which can be selectively refined as per curvature amplitude, as dictated by property variations. Version 1.0 of the code is a newly developed algorithm, which has been developed for increased computational efficiency and higher level of modeling accuracy. The code is designed to replicate the capabilities available in the old UBC-GIF DCIP3D program library and its functionality, allows working with old data formats, whenever possible. In addition to the forward modeling and inversion routines, the program library includes numerous utility executables, designed to facilitate the transition from regular to octree meshes and to support format exchange between the new and the old code versions.

In addition to the new approach in discretization, the DCIP3D OCTREE has been released with implemented parallelization using OpenMP, optimized for usage on multi-core computers with hyperthreading functionality. For parallel usage on local networks and commodity clusters, DCIP3D OCTREE has been compiled with Message Pass Interface (MPI) using the MUMPS direct solver (http://graal.ens-lyon.fr/MUMPS/). Among the newly implemented modifications to the beta version of the code, the most significant are the capability to invert borehole (subsurface) data, to drape the 2D (XY) survey geometries over 3D topography, the added ability to incorporate a-priori electrical resistivity or chargeability information by utilizing a 3D weighting function (which can be designed to emphasize or suppress some known spatial or directional features of the recovered model or otherwise, to force desired model conditions via property bound constraints), and interface weighting which can be used to define sharp contacts within the reference model (i.e. faults, unconformities, etc.) and laterally smoothing near surface variations in the recovered model.

Boundary constraint is achieved by imposing restrictions on each cell in the mesh to have a model value of m, such that $m^l \leq m \geq m^u$, where the bounds m^l and m^u, the lower and upper bound, respectively, are prescribed by the user. The conjugate-gradient solution implements this through projected gradient techniques.

1.2 Array types and Earth models

All linear survey surface-array types, including non-standard or uneven arrays, as well as their combinations can be inverted. There is no restriction on array geometry or electrode positioning, as long as the electrode locations are within the extent of the mesh. Recently, capability to invert borehole data and combined surface-borehole data sets has been added to the code.

DCIP3D OCTREE considers the subsurface in terms of a mesh of rectangular cells. Numerical accuracy is increased with usage of smaller cells, but this also drastically increases the size of the problem. The idea behind usage of octree meshes is that in order to minimize the computational costs, the discretization of the volume should be adaptive, based on the quickness of recovered property transition in each direction. Figure 1 shows an example of adaptive refinement for a circular structure.

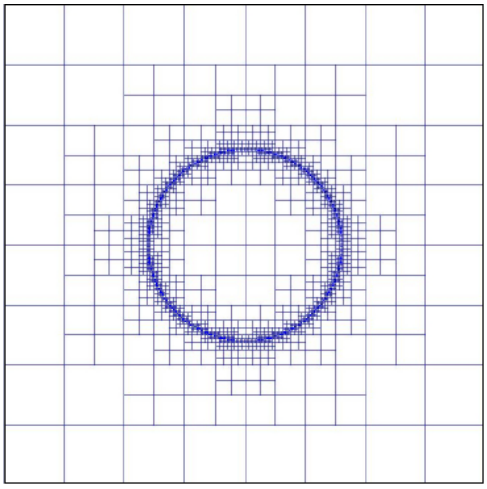

Figure 1: An example of adaptive refinement concept used in octree meshes.

When working with octree meshes, the underlying (base) mesh is defined as a regular 3D discretization with number of cells in each dimension equal to some power of 2. This underlying mesh is the finest possible discretization, which can be used in the inversion at any later stage, without using remeshing procedures. The idea is that if recovered model properties change slowly over a certain volume, then the cells bounded by this volume can be merged into a single cell without losing any accuracy in modelling, and only refined when the model begins changing. The spatial variability of model properties is a measure of the mesh refinement.

1.3 Program library contents

The package that can be licensed includes the following executable programs for performing forward modelling, and inversion of 3D DC resistivity or induced polarization (IP) surveys. Additional functionality is included in supplementary utility programs, which can be used to create and refine octree meshes, calculate octree cell centres, remesh octree models, create weighting files, and convert octree model to non-octree model or vise-versa on both: Windows and Linux platforms. The package contains the following programs:

DCIPoctreeFwd:	Forward model conductivity/chargeability models to calculate data.
DCoctreeInv:	Invert 3D DC data to develop a conductivity model.
IPoctreeInv:	Invert 3D IP data to develop a chargeablility model.
create_octree_mesh:	Create an octree mesh file from electrode locations and optionally topography.
3DModel2Octree:	Convert from a 3D UBC-GIF model to an octree mesh/model.
octreeTo3D:	Convert from an octree model to a standard 3D UBC-GIF model.

`refine_octree`:	Make an octree mesh finer based on the values of the input model.
`remesh_octree_model`:	Convert a model from one octree mesh to another.
`surface_electrodes`:	Place the electrodes on the topographic surface.
`octree_cell_centre`:	Read in an octree mesh, and output a 3-columns file of cell centres.
`interface_weights`:	Create a weight file for the octree cell interfaces.
`create_weight_file`:	Create an octree cell weighting file.

2 Theoretical background for DCIP3D OCTREE

2.1 Introduction

This manual presents theoretical background, numerical examples, and explanation for implementing the program library DCIP3D OCTREE v1.0. This suite of algorithms, developed at UBC-GIF, is needed to efficiently invert large sets of DC potential data and IP responses over a 3D earth structure. The manual is designed so that a geophysicist who has understanding of DC resistivity and induced polarization field experiments, but who is not necessarily versed in the details of inverse theory, can use the codes to invert his or her data.

A typical DC/IP experiment involves inputting a current I to the ground and measuring the potential away from the source. In a time-domain system the current alternates in direction and has off-times between the current pulses at which the IP voltages are measured. A typical time-domain signature is shown in Figure 2. In that figure, ϕ_σ is the potential that is measured in the absence of chargeability effects. This is the instantaneous value of the potential measured when the current is turned on. In mathematical terms this potential is related to the electrical conductivity σ by:

$$\Phi_\sigma = \mathcal{F}_{dc}[\sigma] \tag{1}$$

where forward mapping operator $\mathcal{F}_{dc}$ is defined by equation (2)

$$\bigtriangledown \cdot(\sigma \bigtriangledown \phi_\sigma) = -I\delta(\mathbf{r} - \mathbf{r}_s) \tag{2}$$

and appropriate boundary conditions. In equation (2) σ is the electrical conductivity in Siemens/metre (S/m), $\bigtriangledown$ is the gradient operator, I is the strength of the input current in Amperes (A), and $\mathbf{r}_s$ is the location of the current source. For typical earth structures σ, while positive, can vary over many orders of magnitude. The potential in equation 2 is the potential due to a single current. This is the value that would be measured in a pole-pole experiment. If potentials from pole-dipole or dipole-dipole surveys are to be generated then they can be obtained by using equation 2 and the principle of superposition.

When the earth material is chargeable, the measured voltage will change with time and reach a limit value which is denoted by ϕ_η in Figure 2. There are a multitude of microscopic polarization phenomena which when combined produce this response but all of these effects can be consolidated into a single macroscopic parameter called chargeability. We denote chargeability by the symbol η. Chargeability is dimensionless, positive, and confined to the region [0,1).

To carry out forward modelling to compute ϕ_η we adopt the formulation of Siegel (1959) which states that the effect of a chargeable ground is modelled by using the DC resistivity forward mapping $\mathcal{F}_{dc}$ but with the conductivity replaced by $\sigma = \sigma(1 - \eta)$.Thus:

$$\phi_\eta = \mathcal{F}_{dc}[\sigma(1 - \eta)] \tag{3}$$

or

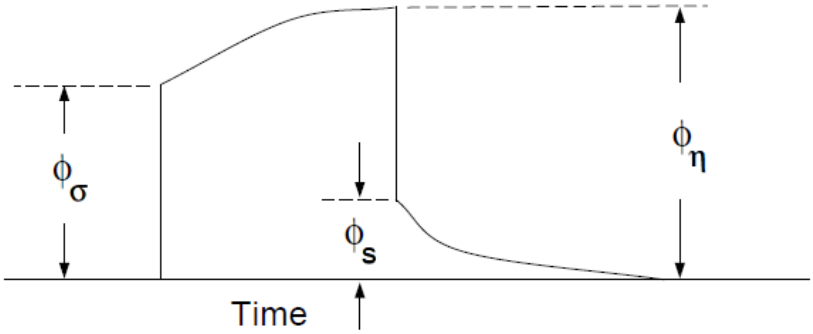

Figure 2: Definition of three potentials associated with DC/IP experiments.

$$\bigtriangledown \cdot(\sigma(1-\eta) \bigtriangledown \phi_\eta) = -I\delta(\mathbf{r}-\mathbf{r}_s) \tag{4}$$

The IP datum can be either the secondary potential (ϕ_s) or the apparent chargeability (η_a). The former is the difference of the forward modelled potentials with, and without, the IP effect:

$$\phi_s = \phi_\eta - \phi_\sigma = \mathcal{F}_{dc}[\sigma(1-\eta)] - \mathcal{F}_{dc}[\sigma] \tag{5}$$

While the apparent chargeability is then given by the ratio:

$$\eta_a = \frac{\phi_s}{\phi_\eta} = \frac{\mathcal{F}_{dc}[\sigma(1-\eta)] - \mathcal{F}_{dc}[\sigma]}{\mathcal{F}_{dc}[\sigma(1-\eta)]} \tag{6}$$

In this definition, the apparent chargeability is dimensionless and, in the case of data acquired over an earth having constant chargeability η_0, we have $\eta_a = \eta_0$. Equations 5 and 6 show that the IP data can be computed by carrying out two DC resistivity forward modellings with conductivities σ and $\sigma(1-\eta)$. The secondary potential is the more general form of IP data and the apparent chargeability is only defined when the linear (or polar) arrays are used along a line on the surface or in the same borehole. When the current and potential dipole-electrodes are arranged in 3-D space and so they are not aligned, the total potential can take on positive, zero, or negative values. The cross-line experiments on the surface and cross-hole experiment on boreholes are examples of such situations. Because of the zero-crossing in the total potentials, the commonly used apparent chargeability is undefined. In these cases, the appropriate data to measure the IP effect is the secondary potential. Therefore, we will use secondary potential as the basic IP datum except in the case of linear arrays.

The field data from a DC/IP survey are a set of N potentials (ideally ϕ_σ, but usually ϕ_η) and a set of N secondary potentials ϕ_s or a quantity that is related to ϕ_s. The goal of the inversionist is to use these data to acquire quantitative information about the distribution of the two physical parameters of interest: conductivity $\sigma(x,y,z)$ and chargeability $\eta(x,y,z)$.

The distribution of conductivity and chargeability in the earth can be extremely complicated.

Both quantities vary as functions of position in 3-D space. In addition, there is often large topographic relief. In this program library, the 3-D nature of the physical properties and surface topography are fully incorporated. The Earth model is divided into prismatic cells each having a constant value of conductivity and chargeability. The surface topography is approximated by a piecewise constant surface.

2.2 Forward Modelling

The forward modelling for the DC potentials and IP apparent chargeabilities is accomplished using a finite volume method (Dey and Morrison, 1979) and a pre-conditioned conjugate gradient technique to solve equation 2. The program that performs these calculations is DCIPoctreeFwd. The DC modelling is performed by a single solution of equation 2, and the IP modelling is performed by carrying out two DC forward modellings. The IP data are generated according to the operations indicated in equations (refeq:potentialsdiff and 6. To illustrate the DC resistivity and IP forward modelling algorithm, we generate synthetic data that would be acquired over the 3-D conductivity structure shown in Figure 3. The model consists of five rectangular blocks buried in a uniform halfspace. Three smaller blocks are placed on the surface while two larger blocks are at depth to simulate the target of the survey. The blocks S1, S2, and B2 are more conductive than the uniform halfspace; and blocks S3 and B1 are more resistive. All five blocks are chargeable. Data from ten east-west lines surface lines with a line spacing of 100m and four vertical boreholes are forward modelled. The surface experiment is carried out using pole-dipole data with a=50m and n=1, 6, while the borehole experiments use a cross-hole pole-dipole configuration with a 50m potential dipole.

Figure 4 displays the DC resistivity data from three selected lines for the surface experiment. The data are displayed in pseudo-section format. Note the strong responses to the conductivity anomalies on the surface. They appear as pant-legs extending from small n-spacing all the way to the largest n-spacing. The buried blocks are hardly identifiable since their responses have low amplitudes and broad distributions and are masked by the surface anomalies. The apparent chargeability pseudo-sections from the same lines are shown in Figure 5 (note that the apparent chargeability is well-defined in this case). The masking effect of the surface blocks are also evident in the IP data. Thus inversion is required.

Since we intend to invert these data, we have added independent Gaussian noise. The standard deviation of the noise is equal to 2% of the datum magnitude plus a small threshold to deal with near zero data. The effect of the added noise can be seen in Figures 4 and 5.

2.3 General inversion methodology

The inverse problem is formulated as an optimization problem where an objective function of the model is minimized subject to the constraints in equation 2 for DC resistivity data or equation 4 for IP data. To outline our methodology it is convenient to introduce a single notation for the data and for the model. We let $\mathbf{d} = (d_1, d_2, ..., d_N)^T$ denote the data, where N is the number of data. Using this notation d_i is either the i^{th} potential in a DC resistivity data set, or the i^{th} secondary potential/apparent chargeability in an IP survey. Let the physical property of interest be denoted

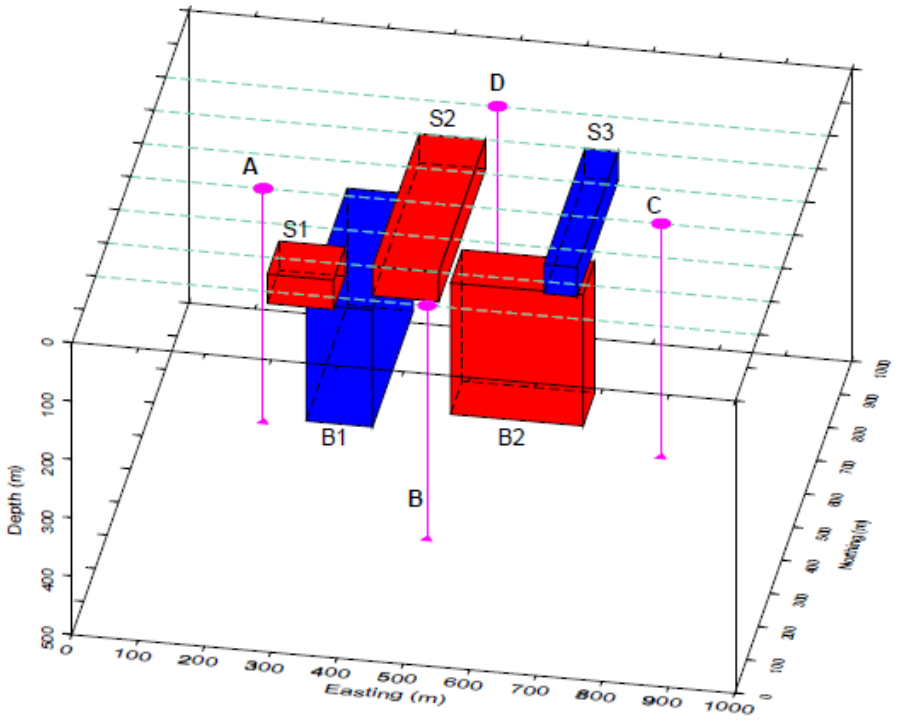

Figure 3: The synthetic model consists of five rectangular blocks in a uniform halfspace. The blocks S1, S2, and B2 are more conductive than the uniform halfspace; and blocks S3 and B1 are more resistive. All five blocks are chargeable. There are seven lines in east-west direction and they are spaced 100m apart. There are also four boreholes that extend to a depth of 400m.

by the generic symbol m for the model element. The quantity m_i denotes the conductivity or chargeability of the i^{th} model cell. For the inversion we choose $m_i = \ln(\sigma_i)$, when inverting for conductivities and $m_i = \eta_i$, when reconstructing the chargeability distribution. Having defined a "model" we next construct an objective function which, when minimized, produces a model that is geophysically interpretable and reproduces the data $\mathbf{d}$ to a justifiable level based on their associated uncertainties. The details of the objective function are problem dependent but generally we need the flexibility to be close to a reference model m_o and also require that the recovered model be relatively smooth in all three spatial directions. Here we adopt a right handed Cartesian coordinate system with y positive north and and z positive down. In defining the model objective function the reference model will generally be included in the first component of the objective function but it can be removed, if desired, from the remaining derivative terms since we are often more confident in specifying the value of the model at a particular point than in supplying an estimate of the gradient. This leads to the following two distinct formulations of the model objective function.

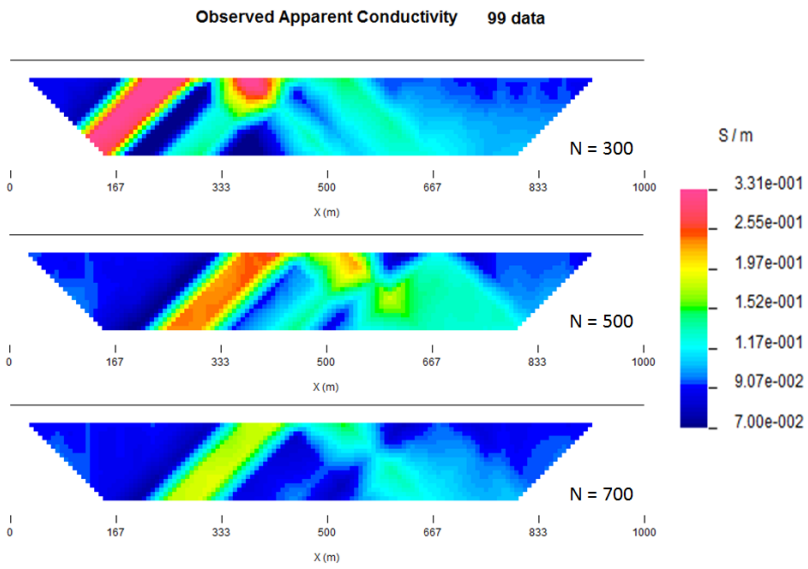

Figure 4: Examples of the apparent conductivity pseudo-sections along three east-west traverses. The data are simulated for a pole-dipole array with a=50m and n=1 to 6. The forward modelled data have been contaminated by independent Gaussian noise with a standard deviation equal to 2% of the accurate datum magnitude and mean of zero. The pseudo-sections are dominated by the surface responses, but there are some indications of the buried prisms. The colormap shows the apparent conductivity in mS/m.

$$\Phi_m = \alpha_s \int\int w_s(m-m_0)^2 dv + \alpha_x \int\int w_x \left(\frac{\partial(m-m_0)}{\partial x}\right)^2 dv + \alpha_y \int\int w_y \left(\frac{\partial(m-m_0)}{\partial y}\right)^2 dv + \alpha_z \int\int w_z \left(\frac{\partial(m-m_0)}{\partial z}\right)^2 dv, \quad (7)$$

and

$$\Phi_m = \alpha_s \int\int w_s(m-m_0)^2 dv + \alpha_x \int\int w_x \left(\frac{\partial m}{\partial x}\right)^2 dv + \alpha_y \int\int w_y \left(\frac{\partial m}{\partial y}\right)^2 dv + \alpha_z \int\int w_z \left(\frac{\partial m}{\partial z}\right)^2 dv, \quad (8)$$

where the weighting functions w_s, w_x, w_y and w_z are spatially dependent, and α_s, α_x, α_y and α_z are coefficients, which affect the relative importance of different components in the model objective function. The reference model m_o may be a general background model that is estimated from previous investigations or it could be a zero model.

The model objective function in equation 7 is used when the `SMOOTH_MOD_DIF` option is selected in the inversion input control file while equation 8 is used when the `SMOOTH_MOD` option is selected in the inversion input control file. The choice of whether or not to include m_o in the derivative terms can have significant effect on the recovered model. The relative closeness of the final model to the

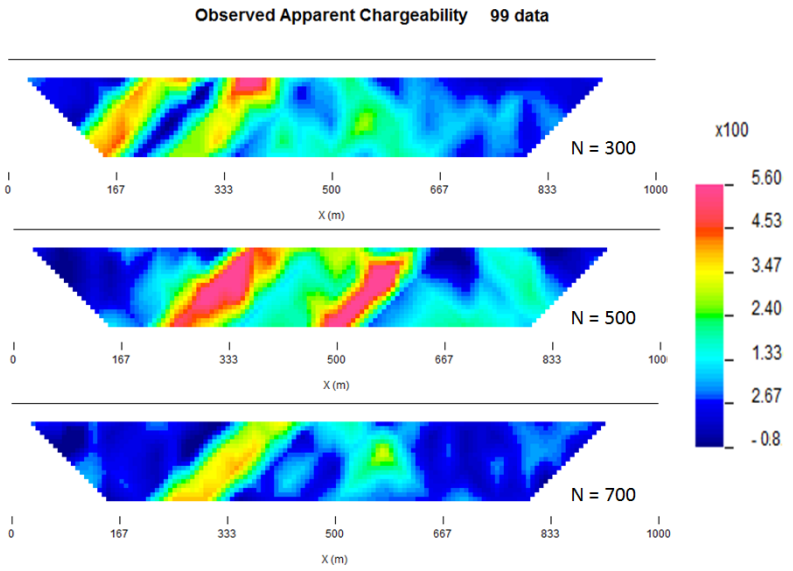

Figure 5: Examples of the apparent chargeability pseudo-sections along three east-west traverses. The data have been contaminated by independent Gaussian noise with a standard deviation equal to 2% of the accurate datum magnitude plus a minimum of 0.001. The same masking effect of near-surface prisms observed in apparent conductivity pseudo-sections is also present here. The colormap shows the apparent chargeability multiplied by 100.

reference model at any location is controlled by the function w_s. For example, if the interpreter has high confidence in the reference model at a particular region, he can specify w_s to have increased amplitude there compared to other regions of the model. The interface weighting functions w_x, w_y, and w_z can be designed to enhance or attenuate structures in various regions in the model domain. If geology suggests a rapid transition zone in the model, then a decreased weighting for flatness can be put there and the constructed model will exhibit higher gradients provided that this feature does not contradict the data.

To perform a numerical solution, we discretize the model objective functions in equations 7 and 8 using a finite difference approximation on the mesh defining the conductivity/chargeability model. This yields:

$$\begin{aligned} \phi_m(\mathbf{m}) &= (\mathbf{m}-\mathbf{m}_o)^T(\alpha_s\mathbf{W}_s^T\mathbf{W}_s + \alpha_x\mathbf{W}_x^T\mathbf{W}_x + \alpha_y\mathbf{W}_y^T\mathbf{W}_y + \alpha_z\mathbf{W}_z^T\mathbf{W}_z)(\mathbf{m}-\mathbf{m}_o), \\ &\equiv (\mathbf{m}-\mathbf{m}_o)^T(\mathbf{W}_m^T\mathbf{W}_m)(\mathbf{m}-\mathbf{m}_o), \\ &= \|\mathbf{W}_m(\mathbf{m}-\mathbf{m}_o)\|^2, \end{aligned} \tag{9}$$

for equation 7 and the following for equation 8.

$$\begin{aligned} \phi_m(\mathbf{m}) &= (\mathbf{m}-\mathbf{m}_o)^T(\alpha_s\mathbf{W}_s^T\mathbf{W}_s)(\mathbf{m}-\mathbf{m}_o) + \mathbf{m}^T(\alpha_x\mathbf{W}_x^T\mathbf{W}_x + \alpha_y\mathbf{W}_y^T\mathbf{W}_y + \alpha_z\mathbf{W}_z^T\mathbf{W}_z)\mathbf{m}, \\ &\equiv (\mathbf{m}-\mathbf{m}_o)^T(\mathbf{W}_s^T\mathbf{W}_s)(\mathbf{m}-\mathbf{m}_o) + \mathbf{m}^T\mathbf{W}^T\mathbf{W}\mathbf{m}, \end{aligned} \tag{10}$$

where $\mathbf{m}$ and $\mathbf{m}_o$ are M-length discretized model vectors which characterize the conductivity/chargeability distributions within the current model and reference model respectively. The individual matrices

$\mathbf{W}_s$, $\mathbf{W}_x$, $\mathbf{W}_y$, and $\mathbf{W}_z$ are straight-forwardly calculated once the model mesh and the weighting functions w_s , w_x, w_y, w_z are defined. The cumulative matrix $\mathbf{W}_m^T\mathbf{W}_m$ is then formed.

Having chosen an appropriate model objective function the next step in setting up the inversion is to define a data misfit measure. Here we use the l_2-norm measure

$$\Phi_d = \left\| \mathbf{W}_d(\mathbf{d} - \mathbf{d}^{obs}) \right\|_2^2 \tag{11}$$

and assume that the contaminating noise in the data is independent and Gaussian with zero mean. Specifying $\mathbf{W}_d$ to be a diagonal datum weighting matrix whose i^{th} element is $1/\sigma_i$, where σ_i is the standard deviation of the i^{th} datum, makes ϕ_d a chi-squared variable distributed with N degrees of freedom. Accordingly $E[\chi^2] = N$ provides a target misfit for the inversion.

The inverse problem is solved by finding a model m which minimizes ϕ_m and misfits the data by a pre-determined amount. Thus the solution is obtained by the following minimization problem of a global objective function ϕ,

$$\begin{aligned} \min \phi &= \phi_d + \beta\phi_m \\ \text{s. t. } \phi_d &= \phi_d^* \text{and optionally} m^l \leq m \leq m^u, \end{aligned} \tag{12}$$

where β is a trade-off parameter that controls the relative importance of the model norm and data misfit. When the standard deviations of data errors are known, the acceptable misfit is given by the expected value ϕ_d^*. In general each parameter in the recovered model ($\mathbf{m}$) lies within its respective lower ($\mathbf{m}^l$) and upper ($\mathbf{m}^u$) bound. Chargeability is positive by definition so bounds are used in all IP inversions to implement the positivity constraint.

The choice of the regularization parameter β in the DC resistivity or IP inversion ultimately depends upon the magnitude of the error associated with the data. The inversion of noisier data requires heavier regularization, thus a larger value of β is required. Since the inversion of DC resistivity data is nonlinear, it is also important need to avoid the possibility of getting trapped in a local minima. The following strategy is implemented to determine an adequate β in the program library DCIP3D OCTREE.

For known uncertainty distributions, the expected value of ϕ_d is easily calculated. For example, independent data with Gaussian noise of zero mean has an expected target misfit (ϕ_d^*) of N number of data. The value of β should be such that the expected misfit is achieved.

A line search based on the misfit curve as a function of beta is performed to approximate the optimal value of β. Due to the high computational expense associated with the inversion, we generally cannot afford to perform the line search by carrying out complete solutions for a series of β's. Starting with a sufficiently large value of β ensures that the line search will successfully find an appropriate value while avoiding the computational expense of a full line search. By reducing β by a fixed factor and performing one or two Gauss-Newton updates (which brings the recovered model close to its final solution for that β) for each value in the decreasing sequence it is possible to determine a general range for the optimal β value. Once this range is established the inversion is run to convergence for a few β values using the recovered model from a nearby β value inversion as the initial model for the next inversion. This greatly reduces the computational expense, by limiting the number of iterations required for convergence. The way optimal β value determined using the

same basic strategy in both the DC and IP inversion codes. The only difference is that which the DC inversion we need to factor the forward modeling matrix every time that the conductivity model is updated, while in the IP case, only one (initial) factorization is required. The pseudo-code for computing the optimal β is shown in Figure 6.

This inversion methodology provides a basic framework for solving a 3D geophysical inversion with arbitrary observation locations. The basic components are: the forward modelling operator, a model objective function that incorporates information about the reference model, a data misfit function, a trade-off parameter that ultimately determines how well the data will be reproduced, and an optimization algorithm that minimizes an objective function, subject to optional bound constraints. The specifics of the DC and IP data inversion are discussed in the following sections.

2.4 Inversion of DC resistivity data

The program library DCIP3D OCTREE provides a DC resistivity inversion program, `DCoctreeInv`. The inversion of DC resistivity data, formulated as the minimization of the global objective function (see equation 12), is nonlinear since the data do not depend linearly upon the conductivity model. A Gauss-Newton approach is used in which the objective function is linearized about a current model, $\mathbf{m}^{(n)}$, a model perturbation is computed, and then used to update the current model. Substituting $\mathbf{m}^{(n+1)} = \mathbf{m}^{(n)} + \delta\mathbf{m}$ into the global objective function (equation 12) give you:

$$\phi(\mathbf{m} + \delta\mathbf{m}) = \left\| \mathbf{W}_d(\mathbf{d}^{(n)} + \mathbf{J}\delta\mathbf{m} - \mathbf{d}) \right\|^2 + \beta \left\| \mathbf{W}(\mathbf{m} + \delta\mathbf{m} - \mathbf{m}_0) \right\|^2 + H.O.T \tag{13}$$

where $\mathbf{J}$ is the sensitivity matrix and the element J_{ij} quantifies the influence of the model change in j-th cell on the i-th datum,

$$J_{ij} = \frac{\partial d_i}{\partial m_j} = \frac{\partial \phi_i}{\partial ln(\sigma_i)} \tag{14}$$

Neglecting the higher order terms (H.O.T.) and setting to zero the derivative with respect to $\delta\mathbf{m}$ yields the following system to solve for the model objective function (7) used when the `SMOOTH_MOD_DIF` parameter is specified in the inversion input control file:

$$(\mathbf{J}^T\mathbf{J} + \beta\mathbf{W}_m^T\mathbf{W}_m)\delta\mathbf{m} = -\mathbf{J}^T(\mathbf{d}^{(n)} - \mathbf{d}) - \beta\mathbf{W}_m^T\mathbf{W}_m(\mathbf{m}^{(n)} - \mathbf{m}_0) \tag{15}$$

where $\mathbf{W}_m^T\mathbf{W}_m$ is defined by equation 9.

Similarly, the following system arises when the model objective function (8) is used (i.e. the `SMOOTH_MOD` parameter is specified in the inversion input control file):

$$(\mathbf{J}^T\mathbf{J} + \beta(\mathbf{W}_s^T\mathbf{W}_s + \mathbf{W}^T\mathbf{W}))\delta\mathbf{m} = -\mathbf{J}^T(\mathbf{d}^{(n)} - \mathbf{d}) - \beta(\mathbf{W}_s^T\mathbf{W}_s(\mathbf{m}^{(n)} - \mathbf{m}_0) + \mathbf{W}^T\mathbf{W}\mathbf{m}) \tag{16}$$

observed data d_{obs}
standard deviations W_d
initial model $m = m_0$
target misfit *chifact N*
Forward model (m) to get predicted data d_{pred}

initial tradeoff parameter $\beta = \beta_{max}$

start β loop

start inner loop 1 to `max_iter_beta`

gradient $g = J^T(d_{pred} - d_{obs}) + \beta W^T W(m - m_{ref})$ **(***)**

solve for model perturbation using IPCG: $(J^T J + \beta W^T W)\,\Delta m = -g$
start IPCG iterations 1 to `max_iter_ipcg`
during each iteration need to calculate $q = (J^T J)\,v$
$p = J\,v$ **(***)**
$q = J^T p$ **(***)**
quit if tolerance < `tol_ipcg`
end IPCG iterations

update the model: $m = m + \Delta m$
Forward model (m) to get predicted data d_{pred}

$\phi_d = 0.5\,\|W_d(d_{pred} - d_{obs})\|^2$
model norm $\phi_m = 0.5\,\|W(m - m_{ref})\|^2$
objective function $\phi = \phi_d + \beta\phi_m$
data misfit $2\phi_d$
quit inversion if *data misfit < target misfit*

end inner loop

update tradeoff parameter $\beta = \beta\,\beta_{factor}$
if $\beta < \beta_{min}$ quit inversion

end β loop

Forward model (m)
factor the matrices $A(m)$ **(***)**

solve $A(m, \Delta t)\,u = rhs$ **(***)**
predicted data $d_{pred} = Q\,u$
end **Forward model**

Figure 6: Pseudo-code describing the DC/IP inversion algorithm

where $\mathbf{W}^T\mathbf{W}$ and $\mathbf{W}_s^T\mathbf{W}_s$ are defined by equation 10.

In these formulations we assume that the matrix $\mathbf{W}_d$ has been absorbed into the sensitivity matrix and data vectors. By solving either of these inverse problems you obtain the model perturbation, which then allows you to generate a new model according to the following relation:

$$\mathbf{m}^{(n+1)} = \mathbf{m}^{(n)} + \alpha\delta\mathbf{m}, \tag{17}$$

where $\alpha \in (0, 1]$ limits the step size and is chosen to ensure that the total objective function is reduced.

The major computational effort in this approach includes the calculation of the sensitivity matrix, solution of the basic linearized equation (15), and the choice of regularization parameter β. The sensitivity is computed using the standard adjoint equation approach, and equation (15 or 16) is solved using a pre-conditioned conjugate gradient (CG) technique.

2.5 Inversion of IP data

To invert IP data it is necessary to linearize equation (5). Let η_i and σ_i denote the chargeability and electrical conductivity of the i^{th} model cell. Linearizing the potential ϕ_η about the conductivity model σ yields:

$$\phi_\eta = \phi(\sigma - \eta\sigma) = \phi(\sigma) - \sum_{j=1}^{M} \frac{\partial\phi}{\partial\sigma_j}\eta_j\sigma_i + H.O.T \tag{18}$$

Substituting into equation (5) yields:

$$\phi_s = \phi_\eta - \phi_\sigma = -\sum_{j=1}^{M} \frac{\partial\phi}{\partial\sigma_j}\eta_j\sigma_i + H.O.T \tag{19}$$

This can be approximately written as:

$$\phi_s = -\sum_j \sigma_j \frac{\partial\phi_i}{\partial\sigma_j}\eta_j = -\sum_j \sigma_j \frac{\partial\phi}{\partial ln(\sigma_j)}\eta_j \tag{20}$$

When apparent chargeability is used as the IP data, substituting the above equation into equation (6), yields:

$$\eta_a = -\sum_j \frac{\sigma_j}{\phi_i}\frac{\partial\phi_i}{\partial\sigma_j}\eta_j = -\sum_j \sigma_j \frac{\partial ln(\phi)}{\partial ln(\sigma_j)}\eta_j \tag{21}$$

Thus the i^{th} datum (either secondary potential or apparent chargeability) is exposed as:

$$d_i = \sum_{j=1}^{M} J_{ij}\eta_{ij} \tag{22}$$

where

$$\left\{ \begin{array}{ll} \frac{\partial \phi_i[\sigma]}{\partial ln\sigma_j}, & \mathbf{d} = \phi_s \\ \\ \frac{\partial ln\phi_i[\sigma]}{\partial ln\sigma_j}, & \mathbf{d} = \eta_a \end{array} \right\} \tag{23}$$

is the sensitivity matrix. Our inverse problem is formulated as:

$$\begin{aligned} \min \phi_m = \|\mathbf{W}_m(\eta - \eta_0)\|^2 & \\ \text{s. t. } \phi_d = \phi_d^* & \quad (24) \\ \text{and} & \quad (25) \\ \eta \geq 0 & \quad (26) \end{aligned}$$

where ϕ_d^* is a target misfit. Again, for ease of future notation we incorporate the diagonal weighting matrix ($\mathbf{W}_d$) into $\mathbf{J}$ and $\mathbf{d}$. In practice the true conductivity σ is not known and so we must use the conductivity found from the inversion of the DC resistivity data to construct the sensitivity matrix elements in equation (23).

3 Elements of the program DCIP3D OCTREE

3.1 Introduction

The DCIP3D OCTREE v1.0 program library consists of three core programs and a nine utilities.

Core Programs:

1. DCIPoctreeFwd: Forward model conductivity/chargeability models to calculate data.
2. DCoctreeInv: Invert 3D DC data to develop a conductivity model.
3. IPoctreeInv: Invert 3D IP data to develop a chargeablility model.

Utilities:

1. create_octree_mesh: Create an octree mesh file from electrode locations and optionally topography.
2. 3DModel2Octree: Convert from a 3D UBC-GIF model to an octree mesh/model.
3. octreeTo3D: Convert from an octree model to a standard 3D UBC-GIF model.
4. refine_octree: Make an octree mesh finer based on the values of the input model.
5. remesh_octree_model: Convert a model from one octree mesh to another.
6. surface_electrodes: Place the electrodes on the topographic surface.
7. octree_cell_centre: Read in an octree mesh, and output a 3-columns file of cell centres.
8. interface_weights: Create a weight file for the octree cell interfaces.
9. create_weight_file: Create an octree cell weighting file.

Each of the above programs requires an input file or files in order to run. Before detailing the procedures for running each of the above programs, we first present information about these general input/output files.

3.2 General files for DCIP3D OCTREE v1.0 programs

Input files can have any user-defined name, while **output** files have restricted file names. Generall speaking, the filename extensions are not important. While the user can provide different file extensions for each file type (i.e. *.msh for mesh files, *.con for conductivity models), some users prefer to use the *.txt filename convention so that files are more easily read and edited in the Windows environment. There are ten general file types which are used by the different codes in DCIP3D OCTREE library:

1. 3D octree mesh: 3D octree mesh defining the discretization of the 3D model region.
2. 3D standard mesh: 3D octree mesh defining the discretization of the 3D model region.
3. topography: Specifies the surface topography.
4. location: Specifies the spatial location of all current and potential electrodes.
5. observation: specifies the spatial location of all current and potential electrodes along with the observed/predicted potential differences with estimated standard deviations.
6. model: Physical property model file structure for forward, initial, reference, and recovered models.
7. cell weighting: Optional file the contains a user defined 3D cell weighting function.
8. interface weighting: Optional file the contains a user supplied interface weighting function for each spatial direction.
9. bounds: Optional file that contains values for upper and lower physical property bounds on each model cell.
10. active cell: Contains location information about active/inactive cells to be used in the inversion.

3.2.1 Octree mesh file

This file contains the 3D octree mesh, for example octree_mesh.msh, which defines the model region. Each octree mesh is defined by the underlying (base) mesh, the coordinates of the southwest top corner, the smallest cell size in each direction, and the actual number of cells in the mesh (dependent on the degree of octree refinement, always smaller or equal to the number of cells in the base mesh). Octree mesh files have the following structure:

NE	NN	NZ		!# of cells in underlying mesh
E_o	N_o	Z_o		!top corner
$\delta\mathrm{X}_{min}$	$\Delta\mathrm{Y}_{min}$	$\Delta\mathrm{Z}_{min}$		!minimum cell size
M				!size of octree mesh
$i(1)$	$j(1)$	$k(1)$	$b_{sz}(1)$	
$\vdots$	$\vdots$	$\vdots$	$\vdots$	
$i(\mathrm{M})$	$j(\mathrm{M})$	$k(\mathrm{M})$	$b_{sz}(\mathrm{M})$	

NE Maximum number of base mesh cells in the east direction if the mesh were evenly divived into cells of width, $\Delta\mathrm{X}_{min}$.

NN Maximum number of cells in the north direction if the mesh were evenly divived into cells of width, $\Delta\mathrm{Y}_{min}$.

NZ Maximum number of cells in the vertical direction if the mesh were evenly divived into cells of thickness, ΔZ_{min}.

E_o N_o Z_o Coordinates, in metres, of the southwest top corner, specified in (Easting, Northing, Elevation). The elevation can be relative to a reference elevation other than the sea level, but it needs to be consistent with the elevation used to specify the locations, observations, and topography files (see the relevant file descriptions).

ΔX_{min} Minimum cell width in the easting (X) direction.

ΔY_{min} Minimum cell width in the northing (Y) direction.

ΔZ_{min} Minimum cell thickness (minimum vertical extent).

M Actual number of discrete cells after merging of base mesh cells into octree mesh. M defines how many cells participate in modelling/inversion, and is always less than or equal to the number of cells in the base mesh.

i i^{th} Physical index of the current cell/block (ordered W to E).

j j^{th} Physical index of the current cell/block (ordered S to N).

k k^{th} Physical index of the current cell/block (ordered top to bottom).

b_{sz} size, in each direction, of the current cell/block with indecies (i,j,k). The volume of the cell/block would be $(\Delta X_{min} * b_{sz}) * (\Delta Y_{min} * b_{sz}) * (\Delta Z_{min} * b_{sz})$.

The mesh should be designed by considering it to consist of a core portion, representing the region of interest, and a padding zone, which ensures that the boundary conditions in the modelling are handled correctly. In the core portion, the size of the smallest cell in the mesh is controlled by the location of current/potential electrodes, the locations of the boreholes, and topography. The selection of the smallest cell for the underlying (base) mesh and the padding distance in each direction is set by the user in the input file for the ultility `create_octree_mesh`, which will be discussed in detail further in the document.

In the presence of surface topography, the top of the octree mesh corresponds to the highest point on the surface (see topography file description).

Example of an octree mesh file

This example shows an octree mesh that is based off of an underlying mesh with 128 cells in both horizontal directions and 64 cells in the vertical direction. The smallest cells in the core portion of the mesh are 25m by 25m by 15m. Following the octree selective mesh refinement process the resulting mesh contains a total of 46,533 cells. The top south-west corner of the mesh has an elevation of 200m and has (x,y) coordinates of (-1064.5m,-1089.5m).

```
128          128      64       !# of cells in underlying mesh
-1064.5   -1089.5   200       !top corner
25            25       15       !minimum cell size
46533                           !size of octree mesh
1             1        1    8
9             1        1    8
17            1        1    8
⋮             ⋮        ⋮    ⋮
113           1        1    8
121           1        1    8
1             9        1    8
9             9        1    8
⋮             ⋮        ⋮    ⋮
121         121      58    8
```

3.2.2 Standard 3D Mesh file

This file contains the 3D mesh, for example mesh.msh, which defines the model region. Standard 3D mesh file have the following structure:

NE	NN	NZ	
E_o	N_o	Z_o	
ΔE_1	ΔE_2	$\cdots$	ΔE_{NE}
ΔN_1	ΔN_2	$\cdots$	ΔN_{NN}
ΔZ_1	ΔZ_2	$\cdots$	ΔZ_{NZ}

NE Number of cells in the East direction.

NN Number of cells in the North direction

NZ Number of cells in the vertical direction

E_o N_o Z_o Coordinates, in meters, of the southwest top corner, specified in (Easting, Northing, Elevation). The elevation can be relative to a reference elevation other than the sea level, but it needs to be consistent with the elevation used to specify the locations, observations, and topography files (see the relevant file descriptions).

ΔE_n n^{th} cell width in the easting direction (ordered W to E).

ΔN_n n^{th} cell width in the northing direction (ordered S to N).

ΔZ_n n^{th} cell thickness (ordered top to bottom).

The mesh can be designed in accordance with the area of interest and the spacing of the data available in the area. In general, the mesh consists of a core region which is directly beneath the area of available data, and a padding zone surrounding this core mesh. Within the core mesh, the size of the cells should be comparable with the spacing of the data. There is no restriction on the relative position of data locations and nodal points in the horizontal direction. The cell width in this area is usually uniform (this becomes important for converting from a standard mesh to and octree mesh).

The maximum depth of the mesh used for inversion should be large enough so that conductive/chargeable material below that depth does not produce a noticeable anomaly with the length scale covered by the data area. A rule of thumb is that the maximum depth should be at least half of the longest side of the data area. Based upon the user's knowledge of the survey area, one may adjust the maximum depth as necessary. The cell thickness in vertical direction usually increases slightly with depth. In the shallow region, the ratio of thickness to width of about half is good, especially when surface topography is present. At depth, a cell thickness close to the cell width is advisable. Once this core mesh is designed, it can be extended laterally by padding with a few cells, possibly of variable width. This padding is necessary when the extracted anomalies are close to the boundary of the core mesh or if there are influences from anomalies outside the area which cannot be easily removed. Problems with more than 1,000,000 model cells, and/or more than a few thousand data points would be considered large, and can be expected to require a considerable amount of computing memory and time.

The vertical position of the mesh is specified in elevation. This is to accommodate the inversion of a data set acquired over a topographic surface. When there is strong topographic relief, which the user wishes to incorporate it into the inversion, special care should be taken to design the mesh. A conceptually simple approach is first to design a rectangular mesh whose top (specified by Z_o) is just below the highest elevation point, and then to strip off cells that are above the topographic surface. This is the approach taken in DCIP3D OCTREE v1.0. The number of cells to be stripped off in each column is determined by the user-supplied topography file. Only the remaining cells will be used in the forward modelling or included in the inversion as model parameters.

Example of mesh file

This example shows a mesh that consists of 26 cells in easting, 27 cells in the northing, and 23 cells in the vertical directions. The top of the mesh is located at 0 m of elevation and the southwest corner is at -350 m easting and -400 m northing. The cells in the core portion of the mesh are all 50 m $\times$ 50 m $\times$ 25 m. There are three cells in the padding zone in every direction.

```
26          27     23
-350       -400    0
200         100    50    21*50.0   50   100   200
200         100    50    20*50.0   50   100   200
20*25.0     50     100     200
```

3.2.3 Topography file

This optional file is used to define the surface topography of the 3D model by the elevation at different locations. The topography file has the following structure:

!	comment	
npt		
E_1	N_1	$ELEV_1$
E_2	N_2	$ELEV_2$
⋮	⋮	⋮
E_n	N_n	$ELEV_n$

Parameter definitions:

- `!` Top lines starting with ! are comments.
- `npt` Number of points defining the topographic surface.
- E_n Easting of the i^{th} point on the surface.
- N_n Northing of the j^{th} point on the surface.
- $ELEV_n$ Elevation of the n^{th} point on the profile.

The lines in this file can be in any order as long as the total number is equal to `npt`. The topographic data need not be supplied on a regular grid. DCIP3D OCTREE assumes a set of scattered points for generality and uses triangulation-based interpolation to determine the surface elevation above each column of cells. To ensure the accurate discretization of the topography, it is important that the topographic data be supplied over the entire area above the model and that the supplied elevation data points are not too sparse.

Example of topography file

The following is an example of a topography file:

```
2007
12300.00   9000.00   0.109411E+04
12300.00   9025.00   0.109545E+04
12300.00   9050.00   0.109805E+04
12300.00   9075.00   0.110147E+04
12300.00   9100.00   0.110555E+04
12300.00   9125.00   0.111011E+04
12300.00   9150.00   0.111490E+04
12300.00   9175.00   0.111971E+04
```

NOTE: Although the cells above the topographic surface are made inactive, they must still be included in the model file as if they are a part of the model. For input model files these cells can be assigned any value. For conductivity models these air cells are typically assigned a value of $10^{-}8$ S/m. The recovered model produced by inversion program also includes the cells that are excluded from the model, but these cells will have unrealistic values as an identifier (e.g. `-100`).

3.2.4 Locations file

This file is used to specify current and potential electrode locations required for the forward modelling of DC/IP data. The locations file has the following structure:

```
!  Comments
[IPTYPE=int]
XA(1)              YA(1)          [ZA(1)]          XB(1)          YB(1)          [ZB(1)]       n(1)
XM(1,1)            YM(1,1)        [ZM(1,1)]        XN(1,1)        YN(1,1)        [ZN(1,1)]
XM(1,2)            YM(1,2)        [ZM(1,2)]        XN(1,2)        YN(1,2)        [ZN(1,2)]
XM(v3)             YM(1,3)        [ZM(1,3)]        XN(1,3)        YN(1,3)        [ZN(1,3)]
⋮                     ⋮              ⋮                ⋮              ⋮
XM(1,n(1))       YM(1,n(1))     [ZM(1,n(1))]     XN(1,n(1))     YN(1,n(1))     [ZN(1,n(1))]

XA(2)              YA(2)          [ZA(2)]          XB(2)          YB(2)          [ZB(2)]       n(2)
XM(2,1)            YM(2,1)        [ZM(2,1)]        XN(2,1)        YN(2,1)        [ZN(2,1)]
⋮                     ⋮              ⋮                ⋮              ⋮
XM(2,n(2))       YM(2,n(2))     [ZM(2,n(2))]     XN(2,n(2))     YN(2,n(2))     [ZN(2,n(2))]

⋮                     ⋮              ⋮                ⋮              ⋮

XM(NC,n(NC))     YM(NC,n(NC))   [ZM(NC,n(NC))]   XN(NC,n(NC))   YN(NC,n(NC))   [ZN(NC,n(NC))]
```

Parameter definitions:

`!` — Lines starting with ! are comments.

`IPTYPE` — A special directive that indicates the IP data type. This directive is only required in IP data files. The `IPTYPE` enables the IP inversion programs to distinguish the apparent chargeability and other similar IP measurements from the basic secondary potentials. `IPTYPE = 1` is commonly used for IP data in which apparent chargeability is well defined (i.e. using dimensionless apparent chargeability, integrated chargeability, PFE, or phase data acquired using electrode configurations that do not produce zero crossings in the measured total potential). The following are some examples of this type of geometry: any pole-pole array (surface or borehole), surface pole-dipole or dipole-dipole array along the same traverse, gradient arrays where the potential electrodes are parallel to the current electrodes, or borehole pole-dipole or dipole-dipole array with all active electrodes in the

same borehole. IPTYPE = 2 is used for secondary potential IP data measured using any electrode geometry. This is typically used when cross-line surface data or cross-hole borehole data are inverted. For these array geometries, the apparent chargeability cannot be defined since the total potential can be zero. The dimensionless apparent chargeabilities (IPTYPE = 1) and the secondary potentials (IPTYPE = 2) can be mixed in the same file. Thus an IP data file can have several occurrences of IPTYPE. All the data are treated as the same type following an IPTYPE directive until a new line changes the type.

XA(i),YA(i),ZA(i)	Location (X,Y,Z) of the i^{th}, current electrode A (measured in metres).
XB(i),YB(i),ZB(i)	Location (X,Y,Z) of the i^{th}, current electrode B (measured in metres).
XM(i,j),YM(i,j),ZM(i,j)	Location (X,Y,Z) of the j^{th} potential electrode M, corresponding with the i^{th} current electrode or electrode pair (measured in metres).
XN(i,j),YN(i,j),ZN(i,j)	Location of the j^{th}, potential electrode N corresponding with the i^{th} current electrode or electrode pair (measured in metres).
NC	The total number of current electrodes or electrode pairs.

NOTE: The brackets $[\cdots]$ indicate that the enclosed parameter is optional. The Z loaction of the electrodes is optional if you as working only with surface data (i.e. your electrodes are dramped to topography) and the IPTYPE only needs to be specified if you are working with IP data.

Examples of a locations file

We provide two example files below. The first file is for a simple surface dataset while the second file shows how borehole data can be incorporated.

Example of surface data locations:

```
!        surface    data
0.00     100.00     0.00    100.00   6
50.00    100.00   100.00    100.00
100.00   100.00   150.00    100.00
150.00   100.00   200.00    100.00
200.00   100.00   250.00    100.00
250.00   100.00   300.00    100.00
300.00   100.00   350.00    100.00

50.00    100.00    50.00    100.00   6
100.00   100.00   150.00    100.00
150.00   100.00   200.00    100.00
200.00   100.00   250.00    100.00
250.00   100.00   300.00    100.00
300.00   100.00   350.00    100.00
350.00   100.00   400.00    100.00

⋮          ⋮        ⋮         ⋮      ⋮
```

Example with borehole data locations:

```
!       borehole    data
500.0    200.0      0.0    500.0   200.0    0.0     45
500.0    800.0      0.0    500.0   800.0   -50.0
500.0    800.0    -25.0    500.0   800.0   -75.0
500.0    800.0    -50.0    500.0   800.0  -100.0
500.0    800.0    -75.0    500.0   800.0  -125.0
500.0    800.0   -100.0    500.0   800.0  -150.0
500.0    800.0   -125.0    500.0   800.0  -175.0
500.0    800.0   -150.0    500.0   800.0  -200.0
500.0    800.0   -175.0    500.0   800.0  -225.0
500.0    800.0   -200.0    500.0   800.0  -250.0
⋮          ⋮        ⋮       ⋮       ⋮       ⋮
```

3.2.5 Observations file

This file is used to specify the current/potential electrode locations along with the observed potential differences (voltages) and their estimated standard deviation. The general format of the observations file is identical to that of the locations file, except for the addition of the voltage and standard deviation columns to the lines specifying the location of potential electrodes M and N. It is important to note that the output of the forward modelling program `DCIPoctreeFwd` does not quite have the correct format to be considered an observation file since the final column which is supposed to contain standard deviations for the error is instead replaced with computed apparent

conductivities/chargeabilities. To convert the DCIPoctreeFwd output into an observation file to be used as the input for the inversion code the column of apparent conductivities/chargeabilities needs to be deleted and proper standard deviations need to be assigned. The following is the file structure of an observation file:

```
!  Comment
[IPTYPE=int]
XA(1)              YA(1)          [ZA(1)]          XB(1)          YB(1)          [ZB(1)]          n(1)
XM(1,1)            YM(1,1)        [ZM(1,1)]        XN(1,1)        YN(1,1)        [ZN(1,1)]        V(1,1)        SD(1,1)
XM(1,2)            YM(1,2)        [ZM(1,2)]        XN(1,2)        YN(1,2)        [ZN(1,2)]        V(1,2)        SD(1,2)
XM(1,3)            YM(1,3)        [ZM(1,3)]        XN(1,3)        YN(1,3)        [ZN(1,3)]        V(1,3)        SD(1,3)
⋮                  ⋮              ⋮                ⋮              ⋮
XM(1,n(1))         YM(1,n(1))     [ZM(1,n(1))]     XN(1,n(1))     YN(1,n(1))     [ZN(1,n(1))]     V(1,n(1))     SD(1,n(1))

XA(2)              YA(2)          [ZA(2)]          XB(2)          YB(2)          [ZB(2)]          n(2)
XM(2,1)            YM(2,1)        [ZM(2,1)]        XN(2,1)        YN(2,1)        [ZN(2,1)]        V(2,1)        SD(2,1)
⋮                  ⋮              ⋮                ⋮              ⋮
XM(2,n(2))         YM(2,n(2))     [ZM(2,n(2))]     XN(2,n(2))     YN(2,n(2))     [ZN(2,n(2))]     V(2,n(2))     SD(2,n(2))


⋮                  ⋮              ⋮                ⋮              ⋮
XM(NC,n(NC))       YM(NC,n(NC))   [ZM(NC,n(NC))]   XN(NC,n(NC))   YN(NC,n(NC))   [ZN(NC,n(NC))]   V(NC,n(NC))   SD(NC,n(NC))
```

Parameter definitions: Parameter definitions:

`!`	Lines starting with ! are comments.
`IPTYPE`	A special directive that indicates the IP data type. This directive is only required in IP data files. The `IPTYPE` enables the IP inversion programs to distinguish the apparent chargeability and other similar IP measurements from the basic secondary potentials. `IPTYPE = 1` is commonly used for IP data in which apparent chargeability is well defined (i.e. using dimensionless apparent chargeability, integrated chargeability, PFE, or phase data acquired using electrode configurations that do not produce zero crossings in the measured total potential). The following are some examples of this type of geometry: any pole-pole array (surface or borehole), surface pole-dipole or dipole-dipole array along the same traverse, gradient arrays where the potential electrodes are parallel to the current electrodes, or borehole pole-dipole or dipole-dipole array with all active electrodes in the same borehole. `IPTYPE = 2` is used for secondary potential IP data measured using any electrode geometry. This is typically used when cross-line surface data or cross-hole borehole data are inverted. For these array geometries, the apparent chargeability cannot be defined since the total potential can be zero. The dimensionless apparent chargeabilities (`IPTYPE = 1`) and the secondary potentials (`IPTYPE = 2`) can be mixed in the same file. Thus an IP data file can have several occurrences of IPTYPE. All the data are treated as the same type following an IPTYPE directive until a new line changes the type.
`XA(i),YA(i),ZA(i)`	Location (X,Y,Z) of the ith, current electrode A (measured in metres).

`XB(i),YB(i),ZB(i)`	Location (X,Y,Z) of the i^{th}, current electrode B (measured in metres).
`XM(i,j),YM(i,j),ZM(i,j)`	Location (X,Y,Z) of the j^{th} potential electrode M, corresponding with the i^{th} current electrode or electrode pair (measured in metres).
`XN(i,j),YN(i,j),ZN(i,j)`	Location of the j^{th}, potential electrode N corresponding with the i^{th} current electrode or electrode pair (measured in metres).
`V(i,j)`	Data value. The DC data should be the potential difference normalized by the current strength and has the units of V/A. While the IP data can have a variety of different units depending on the `IPTYPE`. When aparent chargeability is specified using `IPTYPE=1` the data can have a variety of units, but is most commonly dimensionless. When the secondary potential is specified by using `IPTYPE = 2`, the data must also be in V/A.
`SD(i,j)`	Standard deviation of the datum `V(i,j)`. This is an absolute value and should not be specified as a percentage.
`NC`	The total number of current electrodes or electrode pairs.

NOTE: The brackets $[\cdots]$ indicate that the Z loaction of the electrodes is optional if you as working only with surface data (i.e. your electrodes are dramped to topography).

NOTE: Special care needs to be taken when mixed IP data are present. Only the dimensionless apparent chargeability can be mixed with the secondary potential data. In this case, the recovered chargeability will be the dimensionless quantity. Any other chargeability data (e.g., PFE or phase) must be first converted to dimensionless apparent chargeability. If no conversion is possible, then the data must be inverted as a single data type (`IPTYPE`). In that case, the recovered chargeability model has the same units as the data.

Examples of an observation file

We provide two example files below. The first file is for a simple surface dataset while the second file shows how borehole data can be incorporated.

Example of surface data observations:

```
  !     surface   data
 0.00    100.00    0.00   100.00        6
 50.00   100.00   100.00  100.00  -5.72998E-04  5.00012E-03
100.00   100.00   150.00  100.00  -9.99526E-03  5.00041E-03
150.00   100.00   200.00  100.00  -1.23266E-03  5.00083E-03
200.00   100.00   250.00  100.00   3.53100E-03  5.00116E-03
250.00   100.00   300.00  100.00  -1.91704E-03  5.00129E-03
300.00   100.00   350.00  100.00  -6.31114E-03  5.00038E-03

 50.00   100.00    50.00  100.00        6
100.00   100.00   150.00  100.00   5.75265E-03  5.00014E-03
150.00   100.00   200.00  100.00   2.28981E-04  5.00052E-03
200.00   100.00   250.00  100.00   2.14355E-03  5.00075E-03
250.00   100.00   300.00  100.00  -4.12922E-03  5.00129E-03
300.00   100.00   350.00  100.00  -3.93082E-03  5.00057E-03
350.00   100.00   400.00  100.00   8.30425E-03  5.00117E-03

   ⋮        ⋮        ⋮       ⋮          ⋮         ⋮
```

Example with borehole observations:

```
  !    borehole   data
500.0    200.0     0.0    500.0  200.0    0.0         45
500.0    800.0     0.0    500.0  800.0   -50.0   -3.18948E-05  1.113E-05
500.0    800.0   -25.0    500.0  800.0   -75.0    -6.511E-05   1.215E-05
500.0    800.0   -50.0    500.0  800.0  -100.0    -6.198E-05   1.266E-05
500.0    800.0   -75.0    500.0  800.0  -125.0    -3.357E-05   1.248E-05
500.0    800.0  -100.0    500.0  800.0  -150.0    -1.701E-05   1.195E-05
500.0    800.0  -125.0    500.0  800.0  -175.0    -2.955E-05   1.133E-05
500.0    800.0  -150.0    500.0  800.0  -200.0    -2.266E-05   1.082E-05
500.0    800.0  -175.0    500.0  800.0  -225.0     3.665E-06   1.049E-05
500.0    800.0  -200.0    500.0  800.0  -250.0    -2.053E-06   1.036E-05
  ⋮        ⋮       ⋮       ⋮      ⋮       ⋮          ⋮          ⋮
```

3.2.6 Model file

This file contains the cell values of the conductivity or chargeability model. The conductivity must have values in S/m, while chargeabilityis often unitless. The initial, reference, and forward models must be in this format. Likewise, the recovered model files will be in this format. Model files have the following structure:

$$\begin{array}{lll} \sigma_1 & \texttt{or} & \eta_1 \\ \sigma_2 & \texttt{or} & \eta_2 \\ \vdots & & \vdots \\ \sigma_i & \texttt{or} & \eta_i \\ \vdots & & \vdots \\ \sigma_M & \texttt{or} & \eta_M \end{array}$$

σ_i Conductivity of the i^th cell. The conductivity is always in S/m. There are no a priori bound constraints set on the recovered conductivitymodel unless specific bound constraints are applied by the user.

η_i Chargeability of the i^th cell. The chargeability is typically unitless. Since chargeability ranges between [0,1) a positivity constraint should be applied.

M The total number of cells in the octree model and its associated octree mesh.

3.2.7 Weights file

This file supplies the user-based weights that act upon the model objective function. The following is the file structure is for the weights file:

$$\begin{array}{l} \texttt{W.S}_1 \\ \vdots \\ \texttt{W.S}_M \end{array}$$

Parameter definitions:

$\texttt{W.S}_i$ Cell weight for the i^th cell of the model. This weight is used within the smallest model component of the model objective function.

Within each part, the values are ordered in the same way as in model file, however, they can be all on one line, or broken up over several lines.

If the surface topography file is supplied, the cell weights above the surface will be ignored. It is recommended that these weights be assigned a value of -1.0 to avoid confusion. If `NO_WEIGHTS` is entered instead of a weights file, then all of the cell weights will be set equal (1.0).

3.2.8 Interface weights file

This file supplies the user-based interface weights that act upon the model objective function. The following is the file structure is for the weights file:

$$
\begin{array}{|c|}
\hline
\texttt{W.E}_1 \\
\vdots \\
\texttt{W.E}_i \\
\vdots \\
\texttt{W.E}_{NIE} \\
\texttt{W.N}_1 \\
\vdots \\
\texttt{W.N}_i \\
\vdots \\
\texttt{W.N}_{NIN} \\
\texttt{W.Z}_1 \\
\vdots \\
\texttt{W.Z}_i \\
\vdots \\
\texttt{W.Z}_{NIZ} \\
\hline
\end{array}
$$

Parameter definitions:

$\texttt{W.E}_i$	Cell weight for the i^th interface perpendicular to the easting direction.
$\texttt{W.N}_i$	Cell weight for the i^th interface perpendicular to the northing direction.
$\texttt{W.Z}_i$	Cell weight for the i^th interface perpendicular to the vertical direction.
NIE	The number if cell interfaces perpendicular to the easting direction.
NIN	The number if cell interfaces perpendicular to the northing direction.
NIZ	The number if cell interfaces perpendicular to the vertical direction.

Within each part, the values are ordered in the same way as in model file. Since these weights for the derivative terms of the model objective function are applied to the interface/boundary between cells, each weighting vector has a length equal to the number of interfaces in that direction(NIE,NIN, or NIZ).

If the surface topography file is supplied, the cell weights above the surface will be ignored. It is recommended that these weights be assigned a value of -1.0 to avoid confusion. If NO_FACE_WEIGHT is entered instead of the weights file, then all of the cell weights will be set equal (1.0).

3.2.9 Bounds file

This file contains an upper and lower physical property bound for each cell in the model. It is optional for the inversion program. Bounds files have the following structure:

$$\begin{array}{cc} \mathtt{b}_1^l & \mathtt{b}_1^u \\ \mathtt{b}_2^l & \mathtt{b}_2^u \\ \vdots & \vdots \\ \mathtt{b}_i^l & \mathtt{b}_i^u \\ \vdots & \vdots \\ \mathtt{b}_M^l & \mathtt{b}_M^u \end{array}$$

Parameter definitions:

$\mathtt{b}_i^l$	Is the lower bound on the i^th model cell.
$\mathtt{b}_i^u$	Is the upper bound on the i^th model cell.

The ordering of the cells is the same as that for model files. The total number of lines in this file should be equal to M, where M is the total number of cells in the octree mesh and model. If a surface topography file is supplied, the bounds for cells above the surface will be ignored. It is recommended that these values be assigned a negative value (e.g. `-1.0`) to avoid confusion.

3.2.10 Active cells file

This file is optional. It has exact same format as the model file, and thus must be the same size. The active cells file contains information about the cells that will be incorporated into the inversion. There are 2 basic types of active cell files: topography active cell and model active cell files. As the name suggests, the topography active cell file defines which cells within the model fall above the topographic surface. By default all cells below the earth's surface are active (set to 1) and incorporated into the inversion while the air cells will be marked as inactive (set to 0) and excluded from the inversion. The model active cell file can be used to make additional cells, which lie beneath the topographic surface, inactive. In doing this the inactive cells are fixed to their corresponding value in the reference model. As in the topography active cell file a 0 marks inactive cells while a 1 marks active cells. Any inactive cells will not influence the minimization of the model objective function. The following is an example of an active cells file:

```
0
0  !  inactive cell
⋮
0
1  !  active cell
⋮
0
⋮
1
```

4 Running the programs

The software package DCIP3D OCTREE includes three three core programs and nine utilities.

Core Programs:

1. DCIPoctreeFwd: Forward model conductivity/chargeability models to calculate data.
2. DCoctreeInv: Invert 3D DC data to develop a conductivity model.
3. IPoctreeInv: Invert 3D IP data to develop a chargeablility model.

Utilities:

1. create_octree_mesh: Create an octree mesh file from electrode locations and optionally topography.
2. 3DModel2Octree: Convert from a 3D UBC-GIF model to an octree mesh/model.
3. octreeTo3D: Convert from an octree model to a standard 3D UBC-GIF model.
4. refine_octree: Make an octree mesh finer based on the values of the input model.
5. remesh_octree_model: Convert a model from one octree mesh to another.
6. surface_electrodes: Place the electrodes on the topographic surface.
7. octree_cell_centre: Read in an octree mesh, and output a 3-columns file of cell centres.
8. interface_weights: Create a weight file for the octree cell interfaces.
9. create_weight_file: Create an octree cell weighting file.

This section discusses the use of these codes individually.

4.1 Introduction

All programs in the package can be executed under Windows or Linux environments. They can be run by either typing the program name by itself, or followed by a control file in the "command prompt" (Windows) or "Terminal" (Linux). They can be executed directly on the command line or in a shell script or batch file. When a program is executed without any arguments, it will either print a simple message describing the usage or otherwise search for a proper control file name in the working directory (this is the case, when the control file name is hard coded). If this is the case, then the name of the corresponding control file if changed by user will result in termination of the executable, followed by an error message. Some executables require more than one input argument.

4.1.1 Execution on a single computer

The command format for use on a single processsor are described below. Within the command prompt or terminal, any of the programs can be called using:

$$\texttt{program arg}_1 \ [\texttt{arg}_2 \cdots \texttt{arg}_i]$$

where:

`program`	is the name of the executable
$\texttt{arg}_i$	is a command line argument, which can be a name of corresponding required or optional file. Typing `-inp` as the control file, serves as a help function and returns an example input file. Some executables do not require control files and should be followed by multiple arguments instead. This will be discussed in more detail later in this section.

Each input control file contains a formatted list of arguments, parameters and filenames specific to the executable. All input control file formats are explained in detail within this section.

For many large data sets running one of the codes may require a prohibitively long time, so it is often useful parallelize the job and send it to multiple processors (cores) on the same computer. The DCIP3D OCTREE v1.0 program library's main programs have been parallelized with Message Pass Interface (MPI).The MPI installation package can be downloaded from `http://www.mcs.anl.gov/research/projects/mpich2/`. The following is an example of a command line executing an MPI process to run `DCIPoctreeFwd` on 4 processors of the local machine:

```
"C:\Program Files\MPICH2\bin\mpiexec.exe" -localonly 4 -priority 1 DCIPoctreeFwd
```

Here, the input arguements are:

`PATH`	Properly defined path to the `mpiexec.exe`.
`-localonly`	Tells the machine that the job is only going to be run on the local machine, and not on a local network or cluster. The number which follows `-localonly` specifies the total number of processors (cores) to be used.
`-priority #`	Sets the priority of the process. Integer grades from -1 (lowest) to 4 (highest) follow. Higher priority means that RAM and processing resources will be primarily allocated for this process, at expense of lower priority processes. Generally, a large job should be assigned a lower priority, as selective resource allocation may slow down other important processes on the computer, including those needed for stable functioning of the operating system.

`DCIPoctreeFwd` The name of the executable. In our case it is assumed that there is an existing path to the executable directory, otherwise proper path should be provided.

4.1.2 Execution on a local network or cluster

MPI can also be used to run the DCIP3D OCTREE core programs on a local nestwork or cluster. The requirements for running an MPI job on a local network or cluster are as follows:

- An identical version of MPI must be installed on all participating machines
- The user must create an identical network account with matching `username` and `password` on every machine.
- Both the executable folder and the working directory need to be shared and visible on every participating computer.
- Before the MPI job is executed, the firewall on all participating computers should be turned off.
- The path should be defined to the executable directory

```
‘‘C:\Program Files\MPICH2\bin\mpiexec.exe’’ -machinefile machine.txt nproc
                        -priority 0 DCIPoctreeFwd
```

Where the input arguements are:

`PATH` Properly defined path to the `mpiexec.exe`.

`-machinefile` The list of participating machines will be read from a “machine file.”

`machine.txt` Name of the machine file. This file lists the network names of the participating machines and number of processors to be allocated for the MPI job for each machine. The following is an example of a machine file:

```
machine01  16
machine02  16
```

In this simple example, there are two participating machines (named `machine01` and `machine02` and each is required to allocate 16 processors for the MPI job.

`nproc` The total number of allocated processors. This number should be equal to the sum of all processors listed for all machines in the machine file.

`-priority 0` — Sets the priority of the process. Integer grades from -1 (lowest) to 4 (highest) follow. Higher priority means that RAM and processing resources will be primarily allocated for this process, at expense of lower priority processes. Generally, a large job should be assigned a lower priority, as selective resource allocation may slow down other important processes on the computer, including those needed for stable functioning of the operating system.

`DCIPoctreeFwd` — The name of the executable. In our case it is assumed that there is an existing path to the executable directory, otherwise proper path should be provided.

4.2 DCIPoctreeFwd

This program performs 3D forward modelling of DC resistivity and IP data over octree meshes.

4.2.1 Control parameters and input files

As a command line argument, `DCIPoctreeFwd` requires an input file containing all parameters and files needed to carry out the forward modelling calculations. This input control file must be named DCIP_octree_fwd.inp and needs to be located in the working directory, from which `DCIPoctreeFwd` is executed. The following is the input control file format:

```
DC |IP |IPL
octree mesh file
LOC_XY |LOC_XYZ locations file
conductivity model
chargeability model
topography active cell file    |ALL_ACTIVE
```

NOTE: Formats of the files listed in this control file are explained in section 3 of this document.

`DC|IP|IPL` — The `DC` option to performs only DC forward modelling, while the `IP` option performs both DC and IP forward modelling. The `IPL` option calculates the IP data by multiplying the sensitivity matrix by the chargeability model. When the `DC` is chosen, the chargeability model line is ignored.

octree mesh — Name of the octree mesh file.

LOC_XY(Z) — `LOC_XY` specifies that the electrode location file only has surface electrodes (no Z coordinate is provided), while `LOC_XYZ` indicates there may be a mix of surface and subsurface electrodes requiring Z locations to assigned for each current and potential electrode in the file.

This is followed by the user-defined name of the file, which contains electrode location coordinates.

conductivity model File containing the cell values of a conductivity model in S/m.

chargeability model File containing the cell values of a chargeability model. Required only if the `IP` or `IPL` option is selected in the first line. This model must be provided in dimensionless units, ranging from $[0, 1)$.

topography active cells If there is a topography file involved in creation of the octree mesh, then the utility `create_octree_mesh` will generate a file named active_cells.txt along with the mesh file. This file has exactly the same format as the model file. It is a single column with number of elements equal to number of cells in the octree mesh. The column is populated by 0's (inactive "air" cells) and 1's (active cells). Inactive cells do not participate in the forward modeling or inversion. For DC inversions inactive cells are assigned an air conductivity value of 10^-8 S/m, while in IP inversions air cells have a chargeability of 0.

4.2.2 Output files

data_dc.txt The DC potential data

data_ip.txt The IP data (only if the `IP` or `IPL` option was selected in line 1 of the control file.

mumps.txt A diagnostic log file output by the MUMPS (a **MU**ltifrontal **M**assively **P**arallel sparse direct **S**olver) package.

model0.(con/chg) The conductivity/chargeability model that was used for forward modelling with air cells removed.

DCIP_octree_fwd.txt Log file which provides details about the parameters used in the forward modelling and diagnostic imformation about the results.

Example of DCIP_octree_fwd.inp control file

```
DC                  !  Output data type
octree_mesh.txt     !  Octree mesh file
LOC_XYZ obs_3d.loc  !  3D (XYZ) electrode location file
model.con           !  Conductivity model
model.chg           !  Chargeability model
ALL_ACTIVE          !  No topography
```

4.3 DCoctreeInv

DCoctreeInv performs the inversion of the DC resistivity data, using the parameters defined in the control file. The program does not require entry of any additional arguments in the command line, however it will be looking for a control file with the specific name (dc_octree_inv.inp), which should not be renamed by user. This input control file contains the parameters and input file names required for the inversion in the following format:

```
octree mesh file
LOC_XY |LOC_XYZ data file
initial model file              |VALUE
reference model file            |VALUE
topography active cell file     |ALL_ACTIVE
model active cell file          |ALL_ACTIVE
cell weighting file             |NO_WEIGHT
interface weighting file        |NO_FACE_WEIGHT
DEFAULT                         |beta_max beta_min beta_factor
alpha_s alpha_x alpha_y alpha_z
chifact
tol_nl mindm iter_per_beta
tol_ipcg max_iter_ipcg
CHANGE_MREF                     |NOT_CHANGE_MREF
SMOOTH_MOD                      |SMOOTH_MOD_DIF
BOUNDS_NONE                     |BOUNDS_CONST bl bu |BOUNDS_FILE file
```

4.3.1 Control parameters and input files

octree mesh — Name of octree mesh file.

LOC_XY(Z) — LOC_XY specifies that the observation file only has surface electrodes (no Z coordinate is provided), while LOC_XYZ indicates there may be a mix of surface and subsurface electrodes requiring Z locations to assigned for each current and potential electrode in the file. This is followed by the user-defined name of the observation file.

initial model — The starting conductivity model can be defined as VALUE, followed by a constant or as a model file for a non-uniform starting model. The latter is especially useful when a previously terminated inversion has to be restarted.

reference model — The reference conductivity model can be defined as VALUE, followed by a constant or as a model file for non-uniform reference models.

topography active cell — This active cell file is used to simulate topography, it has the same format as the model file and should be compatible with the octree

mesh. In this file active cells are denoted by a 1, while inactive cells are denoted by a 0. Inactive cells are assigned a conductivity of 10e-8 S/m in the recovered model. If no topography is considered for the inversion then `ALL_ACTIVE` should be selected.

model active cell

An active cell file which controls which model cells are included in the inversion. Like the topography active cell file, it has the same format as the model file and should be compatible with the octree mesh. In this file active cells are denoted by a 1, while inactive cells are denoted by a 0. Inactive cells in the recovered model are set to the corresponding physical property value in the refernece model. If you wish to solve for all model cells then `ALL_ACTIVE` should be selected.

cell weighting

File containing the cell weighting vector (one weight for each cell in the octree model). If `NO_WEIGHT` is entered, default values of unity are used.

interface weighting

File containing information for cell interface weighting (i.e one weighting value for each cell interface). The utility `interface_weights` is used to create this interface weighting file. If `NO_FACE_WEIGHT` is entered, default values of unity are used.

`beta`

This line controls the selection of the initial regularization parameter (`beta_max`), as well as its cooling step (`beta_factor`) and the minimum β value (`beta_min`). These values are computed automatically if `DEFAULT` option is selected. However if a previously terminated inversion has to be restarted it is convenient to quickly resume the job at its last step by assigning these parameters manually.

$\alpha_s, \alpha_x, \alpha_y, \alpha_z$

Coefficients for each model component: α_s is the smallest model component, α_x is the coefficient for the derivative in the easting direction, α_y is the coefficient for the derivative in the northing direction, and α_z is the coefficient for the derivative in the vertical direction.

Some reasonable starting values might be: $\alpha_s = 0.0001, \alpha_x = \alpha_y = \alpha_z = 1.0$. None of the alpha's can be negative and they cannot be all set equal to zero.

NOTE: The four coefficients α_s, α_x, α_y and α_z in line 10 of the control file can be though of in terms of three corresponding length scales L_x, L_y and L_z. To understand the meaning of the length scales, consider the ratios α_x/α_s, α_y/α_s and α_z/α_s. They generally define the smoothness of the recovered model in each direction. Larger ratios result in smoother models, while smaller ratios result in blockier models. The conversion from α's to length scales can be done by:

$$L_x = \sqrt{\frac{\alpha_x}{\alpha_s}};\ L_y = \sqrt{\frac{\alpha_y}{\alpha_s}};\ L_z = \sqrt{\frac{\alpha_z}{\alpha_s}} \tag{27}$$

where length scales are defined in metres. When user-defined, it is preferable to have length scales exceed the corresponding cell dimensions.

`chifact` Chi-factor can be used to scale the data misfit tolerance. By default a `chifact`=1 should be used. Increasing or decreasing the `chifact` is equivalent to scaling the assigned standard deviations, an increased `chifact` corresponds to increased error values, which allows for a larger data misfit at convergence.

`tol_nl, mindm, iter_per_beta` The first parameter `tol_nl` defines a tolerance for the relative gradient at each β step. $tol\_nl = \frac{\|g\|}{\|g_o\|}$, where g is the current gradient and g_o is the gradient at the start of the current β step iteration. If the relative gradient is less than `tol_nl` then the code exits the current β iteration and dereases β by `beta_factor`. `mindm` defines the smallest allowable model perturbation (if the Δm recovered as a result of IPCG iteration is smaller than `mindm`, then the current β iteration is terminated and the regularization parameter (β) is reduced by `beta_factor` before the next β step. `iter_per_beta` sets the maximum number of times that the model can be updated within a given β iteration.

`tol_ipcg, max_iter_ipcg` `tol_ipcg` is the fit tolerance to the IPCG iteration needs to solve the model perturbation (Δm) (defines how well the system $(\mathbf{J}^T\mathbf{J} + \beta\mathbf{W}_m^T\mathbf{W}_m)\Delta m = -g$ is solved); while `max_iter_ipcg` defines the maximum number of IPCG iterations allowed per β step to solve for the model perturbation (Δm).

`CHANGE_MREF` |`NOT_CHANGE_MREF` This parameter provides the optional capability to change the reference model at each β step. If the `CHANGE_MREF` option is selected, then reference model is updated every time the regularization parameter changes and is set to the last recovered model from previous iteration. This may result in quicker convergence. If the `NOT_CHANGE_MREF` option is used, then the same reference model, as originally defined in line 4 is used throughout the inversion.

`SMOOTH_MOD` |`SMOOTH_MOD_DIF` This option is used to define the reference model in and out of the derivative terms of the objective function. The options are: SMOOTH_MOD_DIF (reference model is defined in the derivative terms of the objective function) and SMOOTH_MOD (reference model is defined only the smallest model term of the objective function).

`BOUNDS` There are three options regarding the bound selection. `BOUNDS_NONE` lifts any boundary constraints and releases the sought parameter range to infinity. `BOUNDS_CONST` followed by a lower bound (`bl`) and an upper bound (`bu`) is used in cases, where there are some generalized restrictions on the recovered model properties (as is the case with chargeability, which must be fall within the range $[0, 1)$).

BOUNDS_FILE is a more advanced option, which is followed by the name of your bounds file. This option allows the user to enforce individual bound constraints on each model cell, which can be very useful when there is reliable a-priori physical property information available. This can be used as a technique to incorporate borehole measurements into the inversion or to impose more generalized estimates regarding the physical property values of known geological formations.

Example of DCOctreeInv control File:

```
octree_mesh.txt          !  mesh file
LOC_XYZ data.dat         !  data file
VALUE 0.001              !  initial conductivity model
VALUE 0.001              !  reference conductivity model
active_cells.txt         !  topography active cells file
ALL_ACTIVE               !  model active cells file
w.dat                    !  weighting file
NO_FACE_WEIGHT           !  no interface weighting applied
DEFAULT                  !  beta_max beta_min beta_factor
1.e-5 1.  1.  1.         !  alpha_s alpha_x alpha_y alpha_z
1.                       !  chifactor
1.e-2 1.e-3 2            !  tol_nl mindm iter_per_beta
1.e-2 15                 !  tol_ipcg max_iter_ipcg
NOT_CHANGE_MREF          !  does not change reference model
SMOOTH_MOD_DIF           !  reference model used in derivative
BOUNDS_CONST 0.0001 1    !  bounds
```

NOTE: A sample input file can be obtained by executing: DCoctreeInv -inp in the command prompt.

NOTE: DCoctreeInv will terminate before the specified maximum number of iterations is reached if the expected data misfit is achieved or if the model norm has plateaued. However, if the program is terminated by the maximum iteration limit, the file DC_octree_inv.log and DC_octree_inv.out should be checked to see if the desired misfit (equal to chifact times the number of data) has been reached and if the model norm is no longer changing. If neither of these conditions have been met then the inversion should be restarted.

4.3.2 Output files:

DCctreeInv saves a model after each iteration. The models are ordered: inv_01.con, inv_02.con, inv_03.con, etc. Similarly, the predicted data is output at each iteration into a predicted data file: dpred_01.txt, dpred_02.txt, dpred_03.txt, etc. The following is a list of all output files created by the program DCctreeInv.

inv.con Conductivity model from the latest iteration. The model is stored in model format and is overwritten at the end of each iteration.

DC_octree_inv.txt A log file in which all of the important information regarding the flow of the inversion is stored, including the starting inversion parameters, mesh information, details regarding the computation (CPU time, number of processors, etc), and information about each iteration of the inversion (i.e. data misfit, model norm components (S,X,Y,Z), model norm, total objective function, norm gradient and relative residuals at each β iteration).

dpred.txt Predicted data from the inverted model in the latest iteration. The predicted data is in the observation file format, with the final column corresponding to data error/standard deviation replaced by apparent conductivity.

DC_octree_inv.out This file is appended at the end of each iteration and has 7 columns, which are as follows: `beta` (value of regularization parameter); `iter` (number of IPCG iteration in a beta loop); `misfit` (data misfit * 2); `phi_d` (data misfit); `phi_m` (model norm); `phi` (total objective function); `norm g`(gradient equal to -RHS when solving Gauss-Newton) and `g rel`(relative gradient equal to $\|g\| \,/\, \|g_o\|$).

mumps.txt A diagnostic log file output by the MUMPS (a **MU**ltifrontal **M**assively **P**arallel sparse direct **S**olver) package.

4.4 IPoctreeInv

`IPoctreeInv` performs the inversion of the IP data, using the parameters defined in the control file. The program does not require entry of any additional arguments in the command line, however it does look for an input control file with the specific name (ip_octree_inv.inp), which must be in the working directory. This control file contains the parameters and input file names required for the inversion, and has the following format:

```
octree mesh file
LOC_XY |LOC_XYZ data file
initial model file              |VALUE
reference model file            |VALUE
conductivity model file
topography active cell file     |ALL_ACTIVE
model active cell file          |ALL_ACTIVE
cell weighting file             |NO_WEIGHT
interface weighting file        |NO_FACE_WEIGHT
DEFAULT                         |beta_max beta_min beta_factor
alpha_s alpha_x alpha_y alpha_z
chifact
tol_nl mindm iter_per_beta
tol_ipcg max_iter_ipcg
CHANGE_MREF                     |NOT_CHANGE_MREF
SMOOTH_MOD                      |SMOOTH_MOD_DIF
BOUNDS_NONE                     |BOUNDS_CONST bl bu |BOUNDS_FILE file
```

4.4.1 Control parameters and input files

octree mesh Name of octree mesh file.

LOC_XY(Z) LOC_XY specifies that the observation file only has surface electrodes (no Z coordinate is provided), while LOC_XYZ indicates there may be a mix of surface and subsurface electrodes requiring Z locations to assigned for each current and potential electrode in the file. This is followed by the user-defined name of the observation file.

initial model The starting chargeability model can be defined as VALUE, followed by a constant or as a model file for a non-uniform starting model. The latter is especially useful when a previously terminated inversion has to be restarted.

reference model The reference chargeability model can be defined as VALUE, followed by a constant or as a model file for non-uniform reference models.

conductivity model The conductivity model is required for the IP inversions since it is needed to compute sensitivites. In most circumstances DC data is collected along with IP data, allowing the user to first invert the DC data and then use the recovered conductivity model as input for the IP inversion.

topography active cell This active cell file is used to simulate topography, it has the same format as the model file and should be compatible with the octree mesh. In this file active cells are denoted by a 1, while inactive cells are denoted by a 0. Inactive cells are assigned a chargeability of 0 in

the recovered model. If no topography is considered for the inversion then ALL_ACTIVE should be selected.

model active cell

An active cell file which controls which model cells are included in the inversion. Like the topography active cell file, it has the same format as the model file and should be compatible with the octree mesh. In this file active cells are denoted by a 1, while inactive cells are denoted by a 0. Inactive cells in the recovered model are set to the corresponding physical property value in the refernece model. If you wish to solve for all model cells then ALL_ACTIVE should be selected.

cell weighting

File containing the cell weighting vector (one weight for each cell in the octree model). If NO_WEIGHT is entered, default values of unity are used.

interface weighting

File containing information for cell interface weighting (i.e one weighting value for each cell interface). The utility interface_weights is used to create this interface weighting file. If NO_FACE_WEIGHT is entered, default values of unity are used.

beta

This line controls the selection of the initial regularization parameter (beta_max), as well as its cooling step (beta_factor) and the minimum β value (beta_min). These values are computed automatically if DEFAULT option is selected. However if a previously terminated inversion has to be restarted it is convenient to quickly resume the job at its last step by assigning these parameters manually.

$\alpha_s, \alpha_x, \alpha_y, \alpha_z$

Coefficients for each model component: α_s is the smallest model component, α_x is the coefficient for the derivative in the easting direction, α_y is the coefficient for the derivative in the northing direction, and α_z is the coefficient for the derivative in the vertical direction.

Some reasonable starting values might be: $\alpha_s = 0.0001, \alpha_x = \alpha_y = \alpha_z = 1.0$. None of the alpha's can be negative and they cannot be all set equal to zero.

NOTE: The four coefficients α_s, α_x, α_y and α_z in line 10 of the control file can be though of in terms of three corresponding length scales L_x, L_y and L_z. To understand the meaning of the length scales, consider the ratios α_x/α_s, α_y/α_s and α_z/α_s. They generally define the smoothness of the recovered model in each direction. Larger ratios result in smoother models, while smaller ratios result in blockier models. The conversion from α's to length scales can be done by:

$$L_x = \sqrt{\frac{\alpha_x}{\alpha_s}};\ L_y = \sqrt{\frac{\alpha_y}{\alpha_s}};\ L_z = \sqrt{\frac{\alpha_z}{\alpha_s}} \tag{28}$$

where length scales are defined in metres. When user-defined, it is preferable to have length scales exceed the corresponding cell dimensions.

`chifact` Chi-factor can be used to scale the data misfit tolerance. By default a `chifact`=1 should be used. Increasing or decreasing the `chifact` is equivalent to scaling the assigned standard deviations, an increased `chifact` corresponds to increased error values, which allows for a larger data misfit at convergence.

`tol_nl, mindm, iter_per_beta` The first parameter `tol_nl` defines a tolerance for the relative gradient at each β step. $tol\_nl = \frac{\|g\|}{\|g_o\|}$, where g is the current gradient and g_o is the gradient at the start of the current β step iteration. If the relative gradient is less than `tol_nl` then the code exits the current β iteration and dereases β by `beta_factor`. `mindm` defines the smallest allowable model perturbation (if the Δm recovered as a result of IPCG iteration is smaller than `mindm`, then the current β iteration is terminated and the regularization parameter (β) is reduced by `beta_factor` before the next β step. `iter_per_beta` sets the maximum number of times that the model can be updated within a given β iteration.

`tol_ipcg, max_iter_ipcg` `tol_ipcg` is the fit tolerance to the IPCG iteration needs to solve the model perturbation (Δm) (defines how well the system $(\mathbf{J}^T\mathbf{J} + \beta\mathbf{W}_m^T\mathbf{W}_m)\Delta m = -g$ is solved); while `max_iter_ipcg` defines the maximum number of IPCG iterations allowed per β step to solve for the model perturbation (Δm).

`CHANGE_MREF` |`NOT_CHANGE_MREF` This parameter provides the optional capability to change the reference model at each β step. If the `CHANGE_MREF` option is selected, then reference model is updated every time the regularization parameter changes and is set to the last recovered model from previous iteration. This may result in quicker convergence. If the `NOT_CHANGE_MREF` option is used, then the same reference model, as originally defined in line 4 is used throughout the inversion.

`SMOOTH_MOD` |`SMOOTH_MOD_DIF` This option is used to define the reference model in and out of the derivative terms of the objective function. The options are: SMOOTH_MOD_DIF (reference model is defined in the derivative terms of the objective function) and SMOOTH_MOD (reference model is defined only the smallest model term of the objective function).

`BOUNDS` There are three options regarding the bound selection. `BOUNDS_NONE` lifts any boundary constraints and releases the sought parameter range to infinity. `BOUNDS_CONST` followed by a lower bound (`bl`) and an upper bound (`bu`) is used in cases, where there are some generalized restrictions on the recovered model properties (as is the case with chargeability, which must be fall within the range $[0, 1)$). `BOUNDS_FILE` is a more advanced option, which is followed by the name of your bounds file. This option allows the user to enforce individual bound constraints on each model cell, which can be very

useful when there is reliable a-priori physical property information available. This can be used as a technique to incorporate borehole measurements into the inversion or to impose more generalized estimates regarding the physical property values of known geological formations.

Example of IPOctreeInv control File:

```
octree_mesh.txt    !  mesh file
LOC_XYZ data.dat   !  data file
VALUE 0            !  initial chargeability model
VALUE 0            !  reference chargeability model
inv.con            !  conductivity file
active_cells.txt   !  topography active cells file
ALL_ACTIVE         !  model active cells file
w.dat              !  weighting file
NO_FACE_WEIGHT     !  no interface weighting applied
DEFAULT            !  beta_max beta_min beta_factor
1.e-5 1.  1.  1.   !  alpha_s alpha_x alpha_y alpha_z
1.                 !  chifactor
1.e-2 1.e-3 2      !  tol_nl mindm iter_per_beta
1.e-2 15           !  tol_ipcg max_iter_ipcg
NOT_CHANGE_MREF    !  does not change reference model
SMOOTH_MOD_DIF     !  reference model used in derivative
BOUNDS_CONST 0 1   !  bounds
```

4.4.2 Output files

IPctreeInv saves a model after each iteration. The models are ordered: inv_01.chg, inv_02.chg, inv_03.chg, etc. Similarly, the predicted data is being written at each iteration into predicted data files: dpred_01.txt, dpred_02.txt, dpred_03.txt, etc. Following is the list of all files created by the program IPctreeInv.

inv.chg Chargeability model from the latest iteration. The model is stored in the standard model format and the file is overwritten at the end of each iteration.

IP_octree_inv.txt A log file in which all of the important information about the flow of the inversion is stored, including the starting inversion parameters, mesh information, details regarding the computation (CPU time, number of processors, etc), and information about each iteration of the inversion (i.e. data misfit, model norm components (S,X,Y,Z), model norm, total objective function, norm gradient and relative residuals at each β iteration).

dpred.txt Predicted data from the inverted model in the latest iteration. The predicted data is in the observation file format, with the final column corresponding to data error/standard deviation replaced by apparent chargeability.

IP_octree_inv.out This file is appended at the end of each iteration and has 7 columns, which are as following: `beta` (value of regularization parameter); `iter` (number of IPCG iteration in a beta loop); `misfit` (data misfit * 2); `phi_d` (data misfit); `phi_m` (model norm); `phi` (total objective function); `norm g`(gradient equal to -RHS when solving Gauss-Newton) and `g rel`(relative gradient equal to g/g_0).

mumps.txt A diagnostic log file output by the MUMPS (a **MU**ltifrontal **M**assively **P**arallel sparse direct **S**olver) package.

4.5 create_octree_mesh

This utility creates an octree mesh from electrode locations and optionally a topography file.

4.5.1 Command line usage

```
create_octree_mesh
```

This utility requires an input control file "create_mesh.inp" to exist in the working directory. The input control file should not be changed by the user.

4.5.2 Input files

The following is the control file format:

```
min_dx                          min_dy                  min_dz
total_expansion_x          total_expansion_y      total_expansion_z
LOC_XY |LOC_XYZ          electrode location file
topography file                  |NO_TOPO
APPROXTOPO |GOODTOPO
```

The input parameters for the control file are:

`min_dx(dy,dz)` The size of base mesh cell (smallest possible cell) in metres.

`expansion` Defines the padding distance in metres outside of the survey area in each direction.

LOC_XY(Z) Electrode location file which is needed for assigning the lateral extent and the depth of the core mesh region based on the electrode geometry. The lateral extent is consistent with the lateral extent of the survey and the depth is assigned as 1/2 of the maximum Tx - Rx distance.

topography Topography file. If no topography is used then the NO_TOPO option should be selected.

APPROXTOPO |GOODTOPO This option allows the user to control the number of cells that are used to define topography in the padding cell region. GOODTOPO will define the topography in the padding region very accurately using a large number of fine cells, while APPROXTOPO will appoximate the topo in the padding region using a smaller number of coarse cells. Since it is typically not crucial to have well defined topography in the padding region APPROXTOPO minimizes the number of padding cells in the octree mesh, which helps improve computational efficiency.

4.5.3 Output files

octree_mesh.log Log file specifying the parameters used by the create_octree_mesh utility.

octree_mesh_#.txt Output octree mesh. During the mesh creation process the user is given 40 different size options which allows them to control the total number of cells in the output octree mesh (cell sizes will range from relatively coarse to nearly as fine as the underlaying mesh). The selected number will be reflected in the name of the produced octree mesh where the "#" now appears.

active_cells.txt The active cell file which defines the inactive air cells as those which lie above the topographic surface. (This file is only is output if a topography file is provided.)

3D_mesh_core.txt Standard 3D mesh for only the core region of the survey.

3D_mesh.txt Standard 3D mesh for the entire volume.

data_z.txt Contains the data file with electrodes placed on the surface. If LOC_XYZ is specified, electrodes above the surface will be moved to the surface, and electrode locations below the surface will be unchanged. This file is only output when there is topography specified.

4.6 refine_octree

This utility is designed to refine a previously created octree mesh at intermediate iteration steps to make it finer, given the corresponding octree conductivity or chargeability model. The idea is that a

balance needs to be maintained between the accuracy of the forward model and the computational speed. It is therefore expected that on the first run, the data will be fit to an increased `chifact` on a rather coarse mesh. The mesh should be refined again so that at each refinement step the data can be fit to a progressively decreasing chifact, which will eventually become 1.

Unlike the initial mesh, the discretization process (which was only dependent on electrode locations), the post refinement discretization will also be based on the curvature of the model. Regions with more abrupt property variations will be discretized to greater degree (although not smaller than the initial underlaying base mesh).

4.6.1 Command line usage

`refine_octree`

This utility requires a control file refine_mesh.inp to exist in the working directory. The control file name is not to be changed by the user.

4.6.2 Input files

The following is the control file format:

```
LOG_MODEL |LIN_MODEL
input octree mesh file
input octree model file
output octree mesh file
output model file
```

The input parameters for the control file are:

`LOG_MODEL` |`LIN_MODEL`	Linear versus logrithmically scaled model file. `LOG_MODEL` is typically used for DC conductivity models while `LIN_MODEL` is usually used for IP chargeability models.
input octree mesh	Defines the input (initial) octree mesh.
input octree model	Defines the input octree model.

4.6.3 Output files

refine_mesh.log	Log file specifying the parameters used by the `refine_octree` utility.
output octree mesh	Defines the output (refined) octree mesh.
output octree model	Defines the output (refined) octree model.

4.7 remesh_octree_model

This utility is used to convert a previously created 3D octree model from one existing octree mesh to another existing octree mesh.

4.7.1 Command line usage

```
remesh_octree_model.exe mesh1_in model1_in mesh2_in model2_out
```

4.7.2 Input files

mesh1_in Input octree mesh.

model1_in Input octree model.

mesh2_in Octree mesh to be used for remeshing.

4.7.3 Output files

model_out New remeshed octree model, defined on mesh2_in.

4.8 octreeTo3D

This utility is designed to convert an existing 3D octree model defined over an octree mesh into a standard model defined over an existing standard 3D mesh.

4.8.1 Command line usage

```
octreeto3D octreeMesh_in octreeModel_in 3Dmesh_in 3Dmodel_out
```

4.8.2 Input files

octreeMesh_in Input octree mesh.

octreeModel_in Input model based on the octree mesh.

3Dmesh_in Input standard 3D mesh that you wish to convert your model to.

4.8.3 Output files

3Dmodel_out Output standard 3D model, defined on the input standard 3D mesh (3Dmesh_in).

4.9 3Dmodel2Octree

This utility is designed to convert an existing standard 3D model, defined over an standard mesh, into a new octree model defined over an existing 3D octree mesh. Inorder for this utility to work the standard 3D mesh and model must have a uniform cell size (i.e. all cells need to have the same dimensions, padding cells need to be removed) and this cell size should be the same as the minimum octree mesh cell size.

4.9.1 Command line usage

```
3Dmodel2octree control_file.inp
```

This utility works with an arbitrary (user-defined) input control file name.

4.9.2 Input files

The input control file format is as follows:

```
LOG_MODEL |LIN_MODEL
input octree mesh file
input standard 3D mesh file
input standard 3D model file
output octree mesh file        |USE_INPUT_MESH
output octree model file
```

LOG_MODEL |LIN_MODEL	Linear versus logrithmically scaled model file. LOG_MODEL is typically used for DC conductivity models while LIN_MODEL is usually used for IP chargeability models.
input octree mesh	Input 3D octree mesh on which to define the new 3D octree model.
input standard mesh	Input standard 3D mesh on which the standard 3D model is based.
input standard model	Input standard 3D model that you wish to convert to an 3D octree model.

4.9.3 Output files

output octree mesh	Output octree mesh file on which the new octree model is defined. The USE_INPUT_MESH option can be specified if you want the output model to be defined on the input octree mesh.

output octree model Output 3D octree model which hopefully contains the structures from the input standard model.

NOTE: Unless the USE_INPUT_MESH option is selected, your input and output octree mesh will not be the same. This is because the input octree mesh, which was dependent on the electrode locations and optionally topography, is refined and cells are subdivided to to improve model resolution in regions of the model where sturctures exist. For this reason the output octree mesh will always have more cells than the input octree mesh unless the input standard model is a uniform half/wholespace. The more structure that your input standard model has the larger the size of your output octree mesh and model.

4.10 surface_electrodes

This utility is designed to drape the existing surface electrode survey geometry onto a user-provided 3D topographic surface. This essentially takes a LOC_XY location file and interpolates the defined topographic surface to determine the Z location of each electrode on the tomographic surface. The electrode locations are then output in a LOC_XYZ location file.

4.10.1 Command line usage

```
surface_electrodes
```

This utility requires a control file surface_electrodes.inp to exist in the working directory. The control file name is not to be changed by the user.

4.10.2 Input files

The following is the control file format:

```
input octree mesh file
topography active cell file
[ONLY_LOC] LOC_XY |LOC_XYZ     electrode location file
output data file
```

input octree mesh Input octree mesh on which the topography active cell file is defined.

topography active cell Input active cell file which defines topography.

LOC_XY(Z) Input observation file. If the ONLY_LOC option is specified a locations file may be used in place of an observation file

NOTE: If an LOC_XYZ observation or location file is specified in the input control file, electrodes above the surface will be draped to the surface, while electrode locations below the surface will remain unchanged.

4.10.3 Output files

LOC_XYZ Output observation/location file, in which the electrodes have been draped onto the topographic surface.

4.11 create_weight_file

This utility is designed to build an octree cell weighting file. Since these are cell weights they are assigned to cell centers. While the primary use of this weighting file is to help control the variability of physical properties in the near surface layers of your recovered model, it can also be manually edited to place more/less wieght on particular model cells where you might have a-priori information.

4.11.1 Command line usage

```
create_weight_file weight.inp
```

4.11.2 Input files

```
input octree mesh file
topography active cell file  |ALL_ACTIVE
3                                          !  # of surface layers
10 5 2.5                                   !  cell weight values
output weight file
```

input octree mesh Input octree mesh on which the topography active cell file is defined.

topography active cell Input active cell file which defines topography. `ALL_ACTIVE` can be selected if there is no topography and all model cells are active.

`# of surface layers` An integer that defines the number of surface layers that you would like to apply weights to. Each layer is a single cell in thickness. Since cell thickness will vary throughout the octree model the layers are defined on the core region of the model where you have the smallest cells. The cell weights are then assigned based on where the top SW corner of the cell falls (i.e. for a large padding cell near the edge of your octree model the topmost cell might be ten times thicker than your smallest surface cell in the core region. In this case this entire cell would be assigned the weight of your surface layer. All of the cells beneath this edge cell would remain unweighted though, because the top SW corners lie below the depth of the second and third layers, as defined by the smaller surface cells in the core region of the model).

`surface weight values` — One surface weight value is required for each of the surface layers. All weight values must be greater than or equal to 1, with 1 denoting no weight (identity) and high numbers heavily weighting the cell towards the refernece model.

4.11.3 Output files

output weight file — Output cell surface weight file. This file has the same general structure as the model files, except with the physical property values are replaced by cell weights.

4.12 interface_weights

This utility is designed to build an interface weighting file which can be particularly useful if you know the location a sharp boundary within your model with a large physical property contrast across it. This utility looks at the physical property gradient across all cell faces within the input model and assigns a small interface weight (less than 1) if the gradient is above a specified tolerance. Assigning the small interface weight (less than 1) forces a sharp contact. This utility is also used to smooth surface variations laterally by placing large weights (greater than 1) on the topographic surface.

4.12.1 Command line usage

```
interface_weights weight.inp
```

This utility works with an arbitrary (user-defined) input control file name.

4.12.2 Input files

The following is the control file format:

```
input octree mesh file
topography active cell file  |ALL_ACTIVE
input octree model file       |NO_MODEL
LOG_MODEL |LIN_MODEL
1.e-3 0.01                                  !  gradtol weightedge
3                                           !  # of surface layers
200.  100.  50.                             !  surface weight values for X and Y faces
output face weight file
```

input octree mesh Input octree mesh on which the topography active cell file and the input octree model are defined.

topography active cell Input active cell file which defines topography. `ALL_ACTIVE` can be selected if there is no topography and all model cells are active.

input octree model Input octree model upon which to compute the physical property gradient across all cell faces. The `NO_MODEL` option is used if you only with to apply surface interface weights.

`LOG_MODEL` |`LIN_MODEL` Linear versus logrithmically scaled model file. `LOG_MODEL` is typically used for DC conductivity models while `LIN_MODEL` is usually used for IP chargeability models.

`gradtol` Gradient tolerance above which to assign to assign an interface weight of `weightedge` to the cell interface.

`weightedge` Interface weight to assign to cell interfaces with a large physical property gradient, which exceeds `gradtol`. Small weight values (less than 1) will force a sharp contact.

`# of surface layers` An integer that defines the number of surface layers that you would like to apply weights to. Each layer is a single cell in thickness. Since cell thickness will vary throughout the octree model the layers are defined on the core region of the model where you have the smallest cells. The interface weights are then assigned based on where the top SW corner of the cell falls (i.e. for a large padding cell near the edge of your octree model the topmost cell might be ten times thicker than your smallest surface cell in the core region. In this case this entire cell would be assigned the weight of your surface layer. All of the cells beneath this edge cell would remain unweighted though, because the top SW corners lie below the depth of the second and third layers, as defined by the smaller surface cells in the core region of the model).

`interface weight values` One interface weight value is required for each of the surface layers. This defines a lateral interface weight for the near surface cells. As these lateral interface weights are increased so does the degree of lateral smoothing.

4.12.3 Output files

output face weight Output interface weighting file which contains three interface weighting vectors: $mathbfw_x$, $mathbfw_y$, and $mathbfw_z$. These vectors are listed in a single column with $mathbfw_x$ followed by $mathbfw_y$ and $mathbfw_z$.

5 Example

5.1 5 Prism example

The example model is comprised of five anomalous rectangular prisms embedded in a uniform halfspace. There are three surface prisms simulating near-surface distortions, and two buried prisms simulating deeper targets (see Figure 3).

The five blocks from Figure 3 are assigned conductivity and chargeability values in accordance with Table 1.

ID	Conductivity (mS/m)	Chargeability (%)
S_1	10	5
S_2	5	5
S_3	0.5	5
B_1	0.5	15
B_2	10	15

Table 1: Electrical conductivity and chargeability, assigned to the 5 blocks contained within the synthetic model.

DC resistivity and IP data are forward modelled for both surface and cross-borehole arrays using the `DCIPoctreeFwd` code. Three different electrode configurations were were used for both DC and IP data types to show the benfits of a joint inversion using both surface and borehole data. Details of the three survey types are as follows:

1. Surface dataset: Pole-dipole arrays with $a = 50$m (smallest potential electrode spacing) and n (potential electrode position) ranging from 1 to 6. Eleven east-west lines with a line spacing of 100m were used to cover the core region of the model which is 1km square. In total 1,089 observations were forward modelled using 209 current electrodes.

2. Borehole dataset: Pole-Dipole arrays located within 4 boreholes, whose locations are specified in table 2. The data were simulated using a borehole array configuration in which the current electrode is moved down each of the 4 boreholes with 25m steps to the depth of 350m. This results in a total of 51 current electrode locations. For each of the current electrode locations, the potential electrode array, with a 50m spread was placed in each of the remaining three boreholes and moved down to a depth of 350m at 25m intervals resulting in 1,530 forward modelled observations.

3. Combined surface and borehole dataset: A combination of the the above pole-dipole surface and borehole arrays, resulting in 260 current electrodes and 2,619 total forward modelled observations.

Prior to inversion, 5% white Gaussian noise was added to the forward modelled data, and uncertainties were assigned to be 5% of the data value plus a small floor. For each of the experiments conducted, most of the inversion parameters were held constant for consistency. Some of these parameters include:

ID	Easting (m)	Northing (m)
A	200	500
B	500	200
C	800	500
D	500	800

Table 2: Locations of the synthetic boreholes.

octree_mesh: 159,772 cells on an underlaying base mesh which is 128x128x64 cells and has a smallest cell size of 30x30x15 m.

ref. model: A uniform halfspace with a conductivity of 0.001 S/m and a chargeability of 0.0001.

int. model: Same as the reference model.

active cells: No topography is used and all model cells are active in this example so, the topography and model active cells are both set to `ALL_ACTIVE`.

cell weights: No cell weights are used (`NO_WEIGHTS`).

interface weights: No interface weights are used (`NO_FACE_WEIGHTS`).

β: β values are set to `DEFAULT`.

α: α coefficients ($\alpha_s = 1.0e-5$, $\alpha_x = 1$, $\alpha_y = 1$, $\alpha_z = 1$). This corresponds to a length scale of approximately 316.23 m in all three directions.

`chifact`: 1.

`tol_nl`: 1.e-2.

`mindm`: 1.e-3.

`iter_per_beta`: 2.

`tol_ipcg`: 1.e-2.

`max_iter_ipcg`: 15.

m_{ref} `change`: The reference model is not changed at each `beta_step` (`NO_CHANGE_MREF`).

`smoothing`: The `SMOOTH_MOD_DIF` option was used in all inversions, so the model objective function is defined by equation(7) (i.e. the reference model is used in the derivative terms).

bounds: For the DC data `NO_BOUNDS` were applied, but for the IP data a positivity constraint was enforced using constant bounds (`BOUNDS_CONST 0 1`) this forces all recovered chargeability values to be between $[0, 1)$.

5.1.1 DC resistivity inversion of surface data over an octree mesh

The first inversion result, which uses only the DC surface data, was carried out using the following input control file.

```
octree_mesh_11.txt      !  octree mesh file
LOC_XYZ 5prism_dc.dat   !  data file
VALUE 0.001             !  initial model
VALUE 0.001             !  reference model
ALL_ACTIVE              !  topography active cell file
ALL_ACTIVE              !  model active cell file
NO_WEIGHT               !  cell weighting file
NO_FACE_WEIGHT          !  interface weighting file
DEFAULT                 !  |beta_max; beta_min; beta_factor
1.0e-5 1.  1.  1.       !  alpha_s; alpha_x; alpha_y; alpha_z
1.                      !  chifact
1.e-2 1.e-3 2           !  tol_nl mindm; iter_per_beta
1.e-2 15                !  tol_ipcg; max_iter_ipcg
NO_CHANGE_MREF          !  change mref
SMOOTH_MOD_DIF          !  smoothing
BOUNDS_NONE             !  bounds
```

The inversion converged after 17 beta iterations to a final data misfit of 1.65615E+03. The recovered model is shown in Figure 7. While the recovered model is quite similar to the true model, especially in the near surface regions its ability to resolve the deeper blocks is clearly limited. Within each of the sections presented the black outlines show the location of the blocks in the true model.

The top panel of Figure 7 shows a cross section through the recovered model at Y = 480m. In this view, the itersected conductive surface block is well resolved, but the thinner resistive surface block is slightly more difficult to pick out among the near surface artifacts (more refined inversion models could be devised to remove or smooth out many of these near surface anomalies using cell and interface weighting). While the presence of the deeper blocks is clearly visible in the top panel the recovered anomalies are smeared out and lack definition.

The second panel from the top shows a depth slice through the model at a depth of 15m. In this view all 3 of the surface blocks are well fairly well resolved. As should be expected the conductive blocks are slightly better resolved than the resistive block. The boundaries of the resistive surface block are somewhat blurred by the presence near surface artifacts (most of which appear to be more resistive than the background in this particular section).

The bottom panel of Figure 7 shows another depth slice through the recovered model. This section cross-cuts the 2 deeper blocks at a depth of Z = 165m. As was observed in the top panel, the deeper blocks are clearly visible but somewhat diffuse in that they are spread over a region larger than that of the true block and lack sharp boundaries. In this section the deep conductive block is much better resolved than the deep resistive block. Although the conductive anomaly is slightly larger than the true block it is centered about the true location. In contrast, the deep

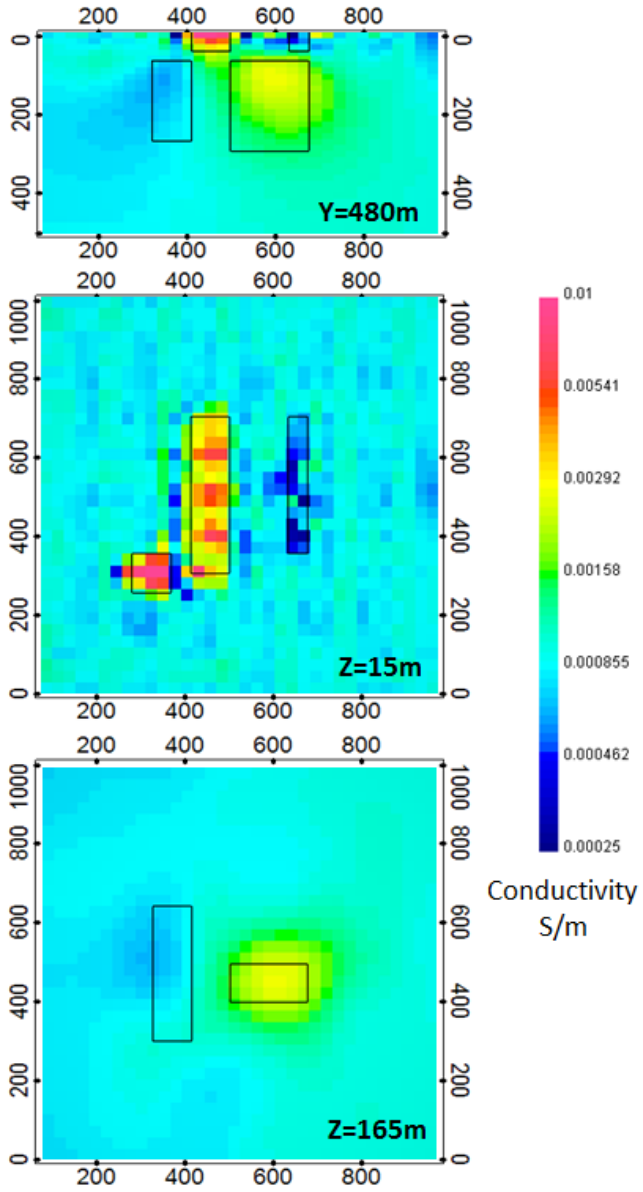

Figure 7: The conductivity model recovered from inversion of surface data. Each of the panels shows a different section through the reocovered model. The top panel shows a cross section along Y = 480m, the middle panel is a depth section from a depth of Z = 15m, and the bottom panel shows a second depth section from Z = 165m. The positions of the true prisms are indicated by the black outlines within each model section. While the surface blocks are nicely resolved by the inversion, the deeper blocks only show up as diffuse anomalies whose shape, spatial location, and physical property contrast with the background are not very well defined.

resistive anomaly is shifted slightly to the west and north of the true block location and is smeared out extensively towards the western edge of the model. As a result of the resistive anomaly's larger size there is less of a physical property contrast between the anomaly and the background. For

this type of surface data the observed decrease in model resolution at depth is anticipated, since we have a limited separation between current and potential electrodes.

5.1.2 IP inversion of surface data over an octree mesh

The following IP inversion result was derived using the same surface electrode array as the previous DC inversion to recover a chargeability model of the subsurface. The input control file for this inversion has the following form:

```
octree_mesh_11.txt      !  octree mesh file
LOC_XYZ 5prism_ip.dat   !  data file
VALUE 0.001             !  initial model
VALUE 0.001             !  reference model
inv.con                 !  refernce conductivity model
ALL_ACTIVE              !  topography active cell file
ALL_ACTIVE              !  model active cell file
NO_WEIGHT               !  cell weighting file
NO_FACE_WEIGHT          !  interface weighting file
DEFAULT                 !  beta_max; beta_min; beta_factor
1.0e-5 1.  1.  1.       !  alpha_s; alpha_x; alpha_y; alpha_z
1.                      !  chifact
1.e-2 1.e-3 2           !  tol_nl; mindm; iter_per_beta
1.e-2 15                !  tol_ipcg; max_iter_ipcg
NO_CHANGE_MREF          !  change mref
SMOOTH_MOD_DIF          !  smoothing
BOUNDS_CONST 0 1        !  bounds
```

Primary differences between this inversion and the previous surface data inversion that was preformed using DC data are restircted to the reference model and bound constraints. For the IP surface data inversion a uniform halfspace with a chargeability of 0.0001 (near-zero) is used for the reference model and the sensitivity calculation was done using the recovered conductivity model from the DC surface data inversion (see Figure 7). While no bound ocnstraints were applied in the DC surface data inversion, a positivity constraint is applied for all of the IP inversions presented here. Constant bounds were set in the input file (`BOUNDS_CONST 0 1`, setting the lower bound to zero and upper bound to 1) to prevent the recovered chargeability values from being negative.

The IP surface data inversion converged after 6 beta iterations to a a final data misfit of 1.30090E+03. The recovered model is shown in Figure 8. While the recovered model offers a good representation of the large scale chargeability distribution, many of the details are lost. Within each of the sections presented the red outlines show the location of the blocks in the true model.

The top panel of Figure 8 shows a cross section through the recovered model at Y = 480m. In this view, the wider of the 2 itersected surface blocks is well resolved, while the thinner chargeable surface block is slightly more difficult to resolve. Although the presence of the deeper blocks is clearly visible in the top panel the recovered anomalies are smeared together into a single anomaly at depth which appears to connect with the chargeable surface anomalies. Despite the fact that

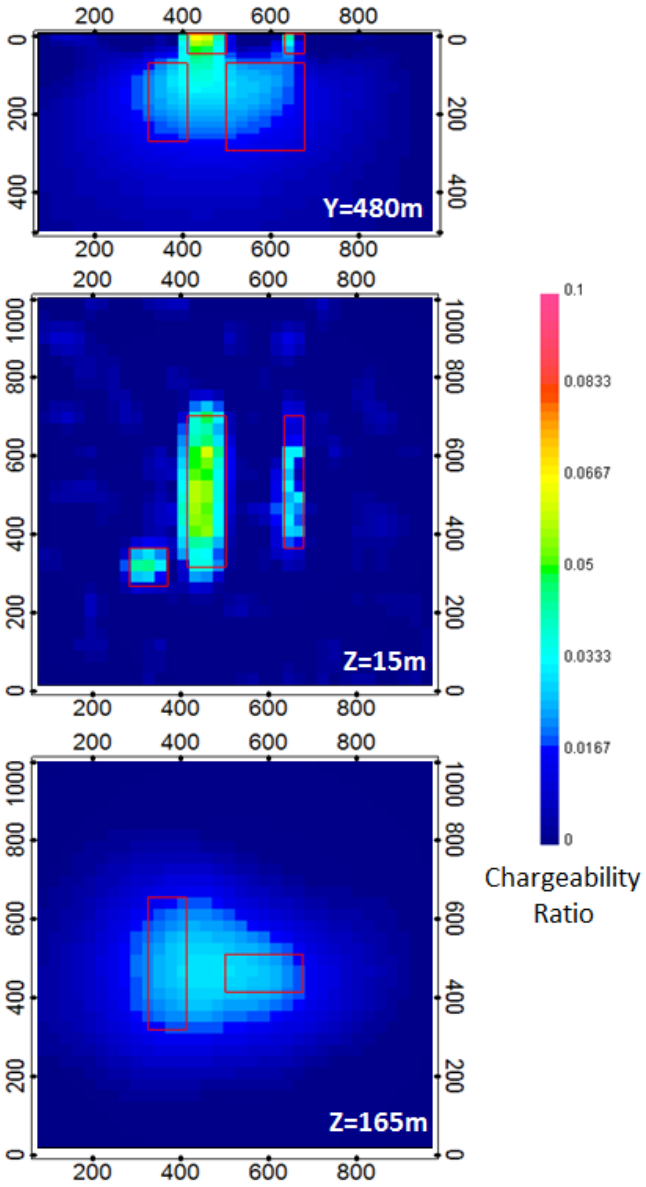

Figure 8: The chargeability model recovered from inversion of surface data shown using 3 different section views which transect the 5 chargeable blocks in the true model. The top panel shows a cross section along Y = 480m, while the middle panel shows a depth section at of Z = 15m, and the bottom panel shows a second depth section from Z = 165m. The positions of the true blocks are indicated by the red outlines within each model section. While the surface blocks are nicely resolved by the inversion, the deeper blocks only show up as a single diffuse anomaly. In addition to the lateral smearing of the chargeable blocks at depth, vertical smearing has also connected the shallow chargeable blocks with those at depth indicating that the resolution of the model decays rapidly with depth.

the deeper blocks are more chargeable than the surface blocks, the inversion result indicates the opposite as a result of the larger near surface sensitivities.

The middle panel shows a depth slice through the model at a depth of 15m. In this view all 3 of the surface blocks are well resolved. The response from the eastern most chargeable surface block is fainter than the other surface blocks because it is thinner. When compared with the DC inversion of surface data (see Figure 7) the near surface artifacts in the IP inversion are lower in amplitude, making it easier to resolve all three surface blocks.

The bottom panel of Figure 8 shows another depth slice through the recovered model which cuts the 2 deeper blocks at a depth of Z = 165m. As was observed in the top panel, the deeper blocks are have been smeared together to form a single diffuse anomaly. From this inversion result it is impossible to tell that the true model contained 2 separate chargeable blocks at depth. As this result clearly illustrates the surface data alone is not capable of accurately resolving the chargeable bodies at depth.

5.1.3 DC inversion of borehole data over an octree mesh

Since the DC inversion based on surface data (see Figure 7) did not resolve the anomalous blocks at depth very precisely, here we preform another inversion using data from 4 separate boreholes. The input control file for this inversion has the following form:

```
octree_mesh_11.txt                ! octree mesh file
LOC_XYZ 5prism_dc_borehole.dat    ! data file
VALUE 0.001                       ! initial model
VALUE 0.001                       ! reference model
ALL_ACTIVE                        ! topography active cell file
ALL_ACTIVE                        ! model active cell file
NO_WEIGHT                         ! cell weighting file
NO_FACE_WEIGHT                    ! interface weighting file
DEFAULT                           ! |beta_max; beta_min; beta_factor
1.0e-5 1.  1.  1.                 ! alpha_s; alpha_x; alpha_y; alpha_z
1.                                ! chifact
1.e-2 1.e-3 2                     ! tol_nl; mindm; iter_per_beta
1.e-2 15                          ! tol_ipcg; max_iter_ipcg
NO_CHANGE_MREF                    ! change mref
SMOOTH_MOD_DIF                    ! smoothing
BOUNDS_NONE                       ! bounds
```

This inversion converged after 15 beta iterations to a a final data misfit of 1.49948E+03. The recovered model is shown in Figure 9. Within each of the sections presented the black outlines show the location of the blocks within the true model. When compared with the inversion of DC surface data in Figure (7) there is a significant decrease in resolution throughout the model and the amplitude of the recovered anomalies significantly under estimates the conductivity contrast between the blocks and the background.

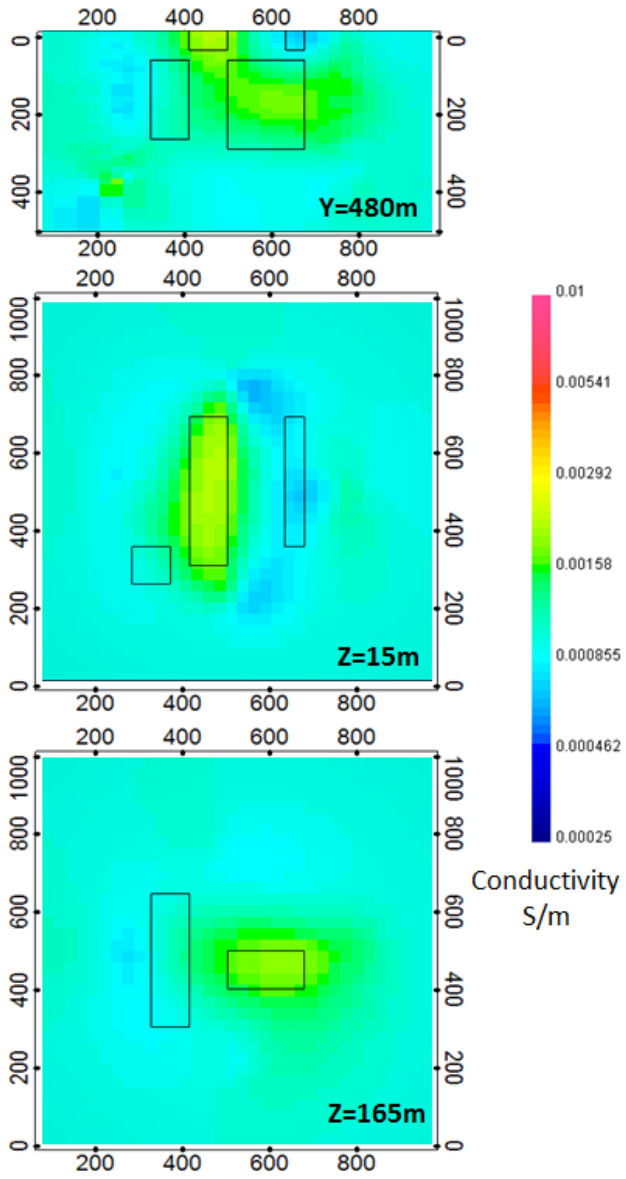

Figure 9: The electrical conductivity model recovered from the inversion of DC borehole data. The position of the true blocks are indicated by the black outlines. When compared with the inversion result using the DC surface data (Figure 7) there is a general decrease in model resolution as a result of decreased sensitivites in region surrounding the blocks due to the borehole survey geometry. While the most significant decreases in the model resoultion are seen in the near surface, this inversion result also does a very poor job of resolving the deep resistive block.

The top panel of Figure 9 shows a cross section through the recovered model at Y = 480m. Here, both of the surface blocks are visible but they lack sharp boundaries and the large conductive surface block has been smeared downwards to connect with the conductive block at depth. While there is a resistive anomaly in the vacinity of the deep resistive block, the recovered anomaly is

shifted to the west.

The second panel from the top shows a depth slice through the model at a depth of 15m. In this section only the large surface conductive block is resolved. The small western most conductive surface block is not recovered at all, and the thin resistive surface block to the east has been highly distorted to form a chevron like shape which points to the east. While the large surface conductive block is the best resolved surface block it has still bled slighly into the background region surrounding the block and lacks sharp boundaries.

The bottom panel of Figure 9 shows another depth slice through the recovered model. This section cross-cuts the 2 deeper blocks at a depth of Z = 165m. At this depth the deep conductive block is recovered, but only a faint trace of resistive material is present to the west of the true location of the deep resistive block. Although the conductive anomaly is slightly larger than the true block it is centered about the true location. As in the above panels the recovered anomalies have very diffuse boundaries.

5.1.4 IP inversion of borehole data over an octree mesh

Using the same survey geometry as in DC borehole data inversion above (Figure 9) an IP inversion was also done to see how well we could resolve the 5 chargeable blocks. Below is the input control file that was used for this IP inversion.

```
octree_mesh_11.txt                !  octree mesh file
LOC_XYZ 5prism_ip_borehole.dat    !  data file
VALUE 0.001                       !  initial model
VALUE 0.001                       !  reference model
inv.con                           !  refernce conductivity model
ALL_ACTIVE                        !  topography active cell file
ALL_ACTIVE                        !  model active cell file
NO_WEIGHT                         !  cell weighting file
NO_FACE_WEIGHT                    !  interface weighting file
DEFAULT                           !  |beta_max; beta_min; beta_factor
1.0e-5 1.  1.  1.                 !  alpha_s; alpha_x; alpha_y; alpha_z
1.                                !  chifact
1.e-2 1.e-3 2                     !  tol_nl; mindm; iter_per_beta
1.e-2 15                          !  tol_ipcg; max_iter_ipcg
NO_CHANGE_MREF                    !  change mref
SMOOTH_MOD_DIF                    !  smoothing
BOUNDS_CONST 0 1                  !  bounds
```

As in the previous IP inversion, the sensitivity was calculated using the conductivity model recovered from the corresponing DC inversion (i.e. DC borehole inversion, see Figure 9) and upper and lower bounds were set to 0 and 1 respectively to enforce a positivity constraint on the recovered chargeability.

This inversion converged after 27 beta iterations to a a final data misfit of 1.52016E+03. The recovered model is shown in Figure 10. Within each of the sections presented the red outlines

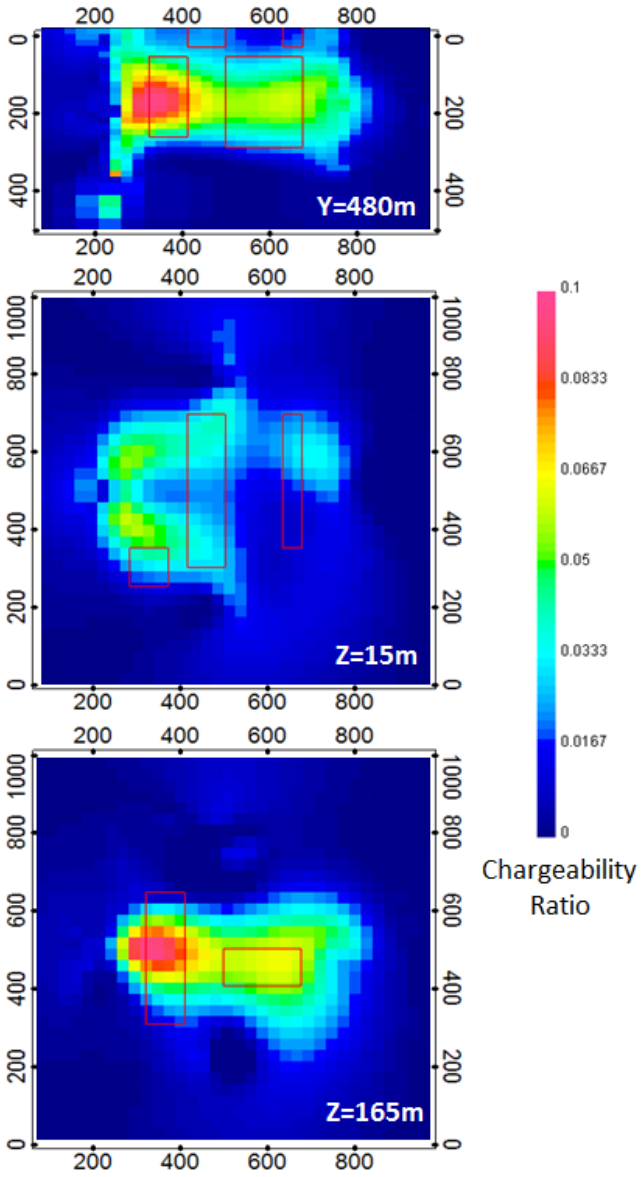

Figure 10: The chargeability model recovered from inversion of surface data shown using 3 different section views which transect the 5 chargeable blocks in the true model. The top panel shows a cross section along Y = 480m, while the middle panel shows a depth section at of Z = 15m, and the bottom panel shows a second depth section from Z = 165m. The positions of the true prisms are indicated by the red outlines within each model section. The conductivity from the DC borehole data inversion (see Figure 9) was used to calculate sensitivities. The depth resolution of this model has significantly increased compared to surface data inversion (Figure 8), however this was done at the expense of the near surface resolution.

show the location of the blocks in the true model. Although the recovered model does a fair job of resolving the deep chargeable blocks it is unable to recover any of the surface blocks.

The top panel of Figure 10 shows a cross section through the recovered model at Y = 480m. In this view, a high chargeability anomaly is centred at depth around the 2 deep blocks, but extensive smearing is visible in both lateral and vertical directions. The region of highest chargeability is centred about the eastern deep chargeable block. This anomaly has been smeared laterally to the east so that it connects with the other deep chargeable block and vertically up to the surface. It does not appear as though any of the surface blocks have been recovered.

The middle panel shows a depth slice through the model at a depth of 15m. This section shows to inability of this inversion result to resolve any of the surface blocks. Based on the location and shape of the high chargeability anomaly visible in this section it appears as though this anomaly is an artifact of the inversion produced by the vertical smearing of the deep chargeable blocks up to the surface. As one would expect with the borehole data the near surface sensitivites are very small when contrasted with the surface data IP inversion (see Figure 8).

The bottom panel of Figure 10 shows another depth slice through the recovered model which cuts the 2 deeper blocks at a depth of Z = 165m. As was observed in the top panel, the deeper blocks are have been smeared together to form a single diffuse anomaly with a region of higher chargeability centred around the western block. From this inversion result it is very difficult to tell that the true model contained 2 separate chargeable blocks at depth. The fact that the deep western chargeable block in the recovered model has a higher charageability than the eastern block must be a result of the north-south orientation of the western block and its proxity to the boreholes since both deep blocks have the same chargeability.

5.1.5 DC resistivity inversion of joint surface and borehole data sets over an octree mesh

Since neither the surface or borehole DC data were able to adequately resolve all 5 blocks, a joint inverion was done using both data sets. The input control file for this inversion has the following form:

```
octree_mesh_11.txt            !  octree mesh file
LOC_XYZ 5prism_dc_joint.dat   !  data file
VALUE 0.001                   !  initial model
VALUE 0.001                   !  reference model
ALL_ACTIVE                    !  topography active cell file
ALL_ACTIVE                    !  model active cell file
NO_WEIGHT                     !  cell weighting file
NO_FACE_WEIGHT                !  interface weighting file
DEFAULT                       !  |beta_max; beta_min; beta_factor
1.0e-5 1.  1.  1.             !  alpha_s; alpha_x; alpha_y; alpha_z
1.                            !  chifact
1.e-2 1.e-3 2                 !  tol_nl; mindm; iter_per_beta
1.e-2 15                      !  tol_ipcg; max_iter_ipcg
NO_CHANGE_MREF                !  change mref
SMOOTH_MOD_DIF                !  smoothing
BOUNDS_NONE                   !  bounds
```

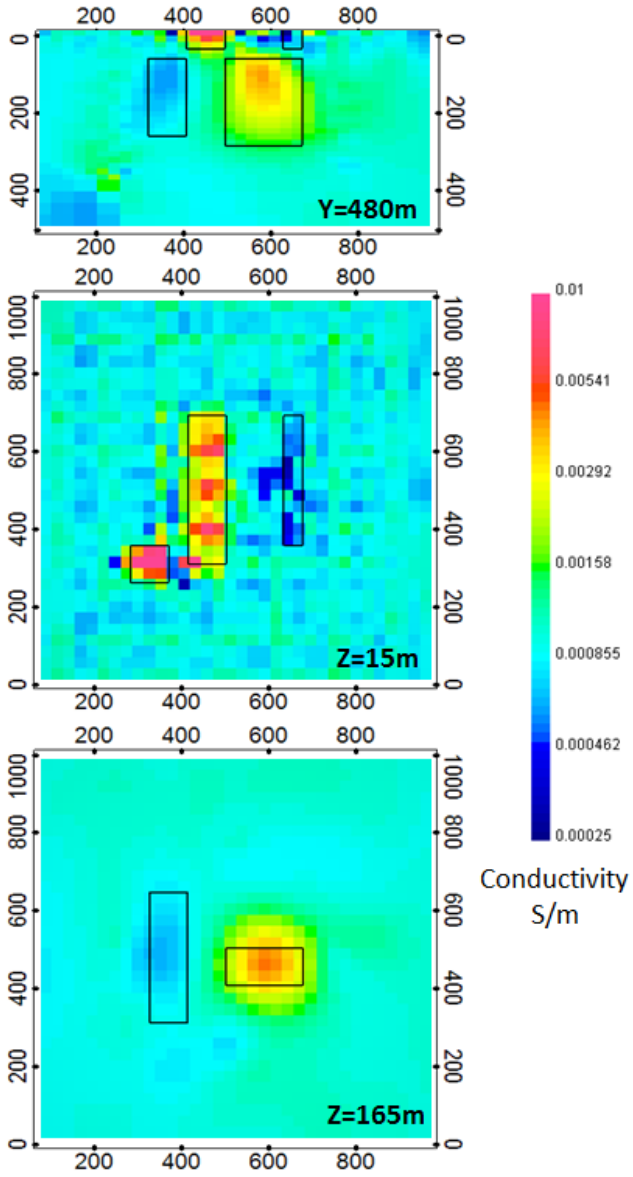

Figure 11: As anticipated the recovered model from the joint inversion of surface and borehole DC data does the best job of resolving the location of all 5 of the blocks in the true synthetic model. The position of the true blocks are indicated by the black outlines. When compared with the inversion result of the DC surface data (see Figure 7) there is a significant increase in model resolution at depth due to the incorporation of the borehole data. The near surface model resolution does not appear to have changed significantly as a result of the joint inversion.

The inversion converged after 23 beta iterations to a final data misfit of 2.45176E+03. The recovered model is shown in Figure 11. Within each of the sections presented the black outlines show the location of the blocks in the true model. While there is not a substantial improvement in the near surface model resolution when compared to the DC surface data inversion (see Figure 7), the deeper conductive and resistive blocks are much better resolved.

The top panel of Figure 11 shows a cross section through the recovered model at Y = 480m. In this view, the itersected conductive surface block is very well resolved, but the thinner resistive surface block is slightly obscured by near surface artifacts (more refined inversion models could be devised to remove or smooth out many of these near surface anomalies using cell and interface weighting). Although the general size and location of the deeper blocks in well constrained they still lack sharp boundaries.

The second panel from the top shows a depth slice through the model at Z = 15m. In this view all 3 of the surface blocks are well fairly well resolved. As should be expected the conductive blocks are slightly better resolved than the resistive block. The boundaries of the resistive surface block are somewhat difficult to discern as a result of near surface artifacts, most of which appear to be more resistive than the background.

The bottom panel of Figure 11 shows another depth slice through the recovered model which cuts through the 2 deeper blocks at a depth of Z = 165m. While the recovered anomalies lack sharp outlines and have been slightly smeared to create circular and oval shaped anomalies. This joint DC inversion result does a far better job of resolving the deep blocks than either the DC surface or DC borehole data inversions (see Figures 7 and 9). The deep resistive block is still shifted slightly to the west in the recovered model, but its north-south location is better defined and the extent of the smearing is dramatically reduced.

5.1.6 IP inversion of joint surface and borehole data sets over an octree mesh

To see if a joint inversion of the surface and borehole IP data would help to better resolve the chargeable blocks in the true model this final inversion was run using the following input control file:

```
octree_mesh_11.txt            !  octree mesh file
LOC_XYZ 5prism_ip_joint.dat   !  data file
VALUE 0.001                   !  initial model
VALUE 0.001                   !  reference model
inv.con                       !  refernce conductivity model
ALL_ACTIVE                    !  topography active cell file
ALL_ACTIVE                    !  model active cell file
NO_WEIGHT                     !  cell weighting file
NO_FACE_WEIGHT                !  interface weighting file
DEFAULT                       !  |beta_max; beta_min; beta_factor
1.0e-5 1.  1.  1.             !  alpha_s; alpha_x; alpha_y; alpha_z
1.                            !  chifact
1.e-2 1.e-3 2                 !  tol_nl; mindm; iter_per_beta
1.e-2 15                      !  tol_ipcg; max_iter_ipcg
NO_CHANGE_MREF                !  change mref
SMOOTH_MOD_DIF                !  smoothing
BOUNDS_CONST 0 1              !  bounds
```

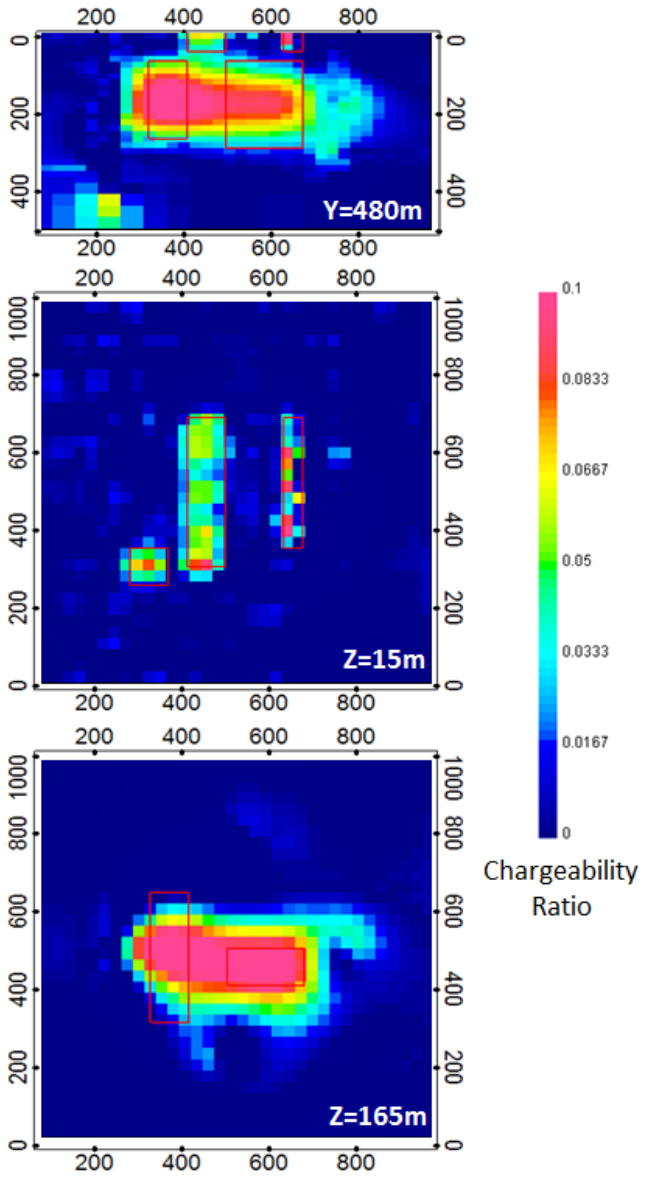

Figure 12: The chargeability model recovered from the joint inversion of surface and borehole IP data is shown using 3 different section views which transect the 5 chargeable blocks in the true model. The top panel shows a cross section along Y = 480m, while the middle panel shows a depth section at of Z = 15m, and the bottom panel shows a second depth section from Z = 165m. The positions of the true prisms are indicated by the red outlines within each model section. The conductivity from the joint DC data inversion (see Figure 11) was used to calculate sensitivities. Significant improvements in the model resolution (both in the near surface and at depth) are apparent when you compare the joint IP data inversion result to that of the surface or borehole IP data inversions.

As in the previous IP inversions, the sensitivity was calculated using the conductivity model recovered from the corresponing DC inversion (i.e. DC joint inversion, see Figure 11) and upper and lower bounds were set to 0 and 1 respectively to enforce a positivity constraint on the recovered chargeability.

This IP inversion converged after 27 beta iterations to a a final data misfit of 2.56821E+03. The recovered model is shown in Figure 12. Within each of the sections presented the red outlines show the location of the blocks in the true model. The recovered model offers a good representation of the overall chargeability distribution, and does the best job of resolving the 5 chargeable blocks contained within the true synthetic model. Despite the significant improvements in model resolution at depth, the recovered model from the joint IP inversion is still incapable of distinguishing the two deep blocks.

The top panel of Figure 12 shows a cross section through the recovered model at Y = 480m. Here, both of the surface blocks are reasonably well resolved. While the vertical extent of the deeper blocks is clearly defined the recovered anomalies are smeared together into a single anomaly at depth. While some vertical smearing is also present between the large surface block and the conductive anomaly at depth the vertical smearing is not nearly as pervasive as it was in the surface or borehole IP inversions (see Figures 8 and 10). In this inversionresult it is also possible to descern that the deeper blocks are more chargeable than the surface blocks, while the surface IP inversion indicated the opposite.

The middle panel shows a depth slice through the model at a depth of Z = 15m. In this view all 3 of the surface blocks are well resolved. The response from the eastern most chargeable surface block is smaller than the other surface blocks because it is thinner. When compared with the DC inversion of surface data (see Figure 7) the near surface artifacts in the IP inversion are lower in amplitude, making it easier to resolve all three surface blocks. Surface interface weighting could be easily applied to remove some of the near surface anomalies and potentially sharpen the recovered surface block boundaries.

The bottom panel of Figure 12 shows another depth slice through the recovered model. This section cross-cuts the 2 deeper blocks at a depth of Z = 165m. As was observed in the top panel, the deeper blocks are have been smeared together to form a single anomaly whose vertical extent is fairly well defined. Although the joint surface and borehole IP data inversion was still unable to resolve the 2 chargeable blocks at depth, it still produces the chargeabilty model which is the most similar to the true model.

6 References

Dey, A., and Morrison, H. F., 1979, 3-D resistivity modelling for arbitrarily shaped three-dimensional structures: Geophysics, **44**, 753–780.

Siegel, H. O., 1959, 3-D mathematical formulation and type curves for induced polarization: Geophysics, **24**, 547–565.