Three-dimensional massively parallel electromagnetic inversion—I. Theory

G. A. Newman and D. L. Alumbaugh

Geophysics Department, Sandia National Laboratories, PO Box 5800, *Albuquerque* NM 87185–0750, *USA*

Accepted 1996 September 23. Received 1996 August 13; in original form 1996 February 13

SUMMARY

An iterative solution to the non-linear 3-D electromagnetic inverse problem is obtained by successive linearized model updates using the method of conjugate gradients. Full wave equation modelling for controlled sources is employed to compute model sensitivities and predicted data in the frequency domain with an efficient 3-D finite-difference algorithm. Necessity dictates that the inverse be underdetermined, since realistic reconstructions require the solution for tens of thousands of parameters. In addition, large-scale 3-D forward modelling is required and this can easily involve the solution of over several million electric field unknowns per solve. A massively parallel computing platform has therefore been utilized to obtain reasonable execution times, and results are given for the 1840-node Intel Paragon. The solution is demonstrated with a synthetic example with added Gaussian noise, where the data were produced from an integral equation forward-modelling code, and is different from the finite difference code embedded in the inversion algorithm

Key words: electromagnetic modelling, inversion, tomography.

INTRODUCTION

The solution of the non-linear 3-D electromagnetic inverse problem has been a goal of geophysicists for many years. The search for this solution has been motivated by its potential applications in mapping electrical conductivity and dielectric permittivity. Knowledge of these electrical properties is critical, since they can be utilized in hydrological modelling, chemical and nuclear waste-site evaluations, mineral, oil, and gas exploration, and more recently reservoir and aquifer characterization.

Up until now, a complete solution to this problem has been hindered by insufficient computing resources. Realistic 3-D reconstructions require the estimation of tens of thousands of unknown electrical parameters. This demand, coupled with the forward-modelling overhead, where up to several million field unknowns may need to be calculated to determine model sensitivities and predicted data, make the solution of the 3-D inverse problem non-trivial. Attempts to circumvent this difficulty have included the use of quasi-linear approximations in both forward and inverse modelling (*cf.* Torres-Verdin & Habashy 1995, 1994; Habashy *et al.* 1995; Zhdanov & Fang 1995) and the use of approximate model sensitivities (Farquharson & Oldenburg 1995). Unfortunately, even these approaches suffer when the number of parameters being estimated exceeds several thousand. Only with the advent of massively parallel (MP) computers could a realistic attack on the problem be proposed.

Even with an MP platform one must be careful when implementing a solution to the inverse problem. Primarily, it is important to avoid directly inverting large matrix systems that are either sparse or full. Rigorous modelling of 3-D EM fields can be carried out efficiently using staggered finite differences, which produces a sparse linear system. On an MP platform this system, if properly pre-conditioned, can be quickly solved using iterative Krylov subspace methods (Alumbaugh *et al.* 1996). On the other hand, the solution of the least-squares inverse problem requires dealing with a full linear system. However, since this system satisfies the normal equations it can also be efficiently solved iteratively with conjugate-gradient (CG) methods. Mackie & Madden (1993) and Zhang *et al.* (1995) used this approach to attack the 3-D magnetotelluric (MT) and direct current (DC) inverse problems, respectively, on scalar platforms. Here we will apply the approach to the 3-D EM inverse problem for frequency-domain dipolar source fields, where the source strengths and locations are known. Because the controlled-source EM problem is far more computationally demanding than both the DC problem, due to its vector nature, and the MT problem, due to the sheer number of source fields to be considered (upwards of several hundred), an MP platform is a necessity. As will be demonstrated below, such a platform allows large models to be reconstructed, which are not underparametrized, in a reasonable amount of time.

A key consideration in developing any inverse solution is the efficient computation of model sensitivities. Because we will solve the inverse problem from an underdetermined point of view, we can efficiently carry out calculations involving model sensitivities using reciprocity, which is known as the adjoint solution to the problem. The use of reciprocity, where the receivers act as sources, can be used to limit forward modelling to the number of transmitter and receiver positions at a given frequency. The traditional approach requires the number of forward solves to be equal to the number of parameters used in the inverse. When the number of parameters far exceeds the number of transmitters and receivers, the adjoint approach is obviously most efficient (*cf.* McGillivray & Oldenburg 1990). In fact, by using the adjoint approach coupled with the CG solution of the normal equations one can even avoid forming individual components of the model sensitivity matrix, resulting in a significant saving of computational memory.

In this paper we first present the theory behind the 3-D inversion scheme, including details of how the scheme must be modified to run on a parallel computer. Next, synthetic data generated by an integral equation code will be inverted. This provides an independent check on the solution, as the data are produced by a code that is very different in nature from the finite-difference code used in the inversion routine, and the two methods are thus prone to different numerical errors. In a companion paper (Alumbaugh & Newman 1996), the inversion scheme will be employed to design a 3-D crosswell survey and invert a crosswell data set collected at the Richmond field station north of Berkeley California.

THE INVERSE SOLUTION

Regularized least squares

As already mentioned, the parametrization used in the 3-D inverse solution will be kept fine because we are interested in reconstructions that do not underparametrize the Earth. This forces the 3-D inverse problem to be underdetermined, which makes it unstable and ill-posed. Reliable estimates of the model parameters (**m**) may be possible if the least-squares inversion is stabilized with regularization (Tikhonov & Arsenin 1977). Regularization removes solutions that are too rough by imposing an additional constraint on the data fit. Reconstructions are required to be smoothed versions of the Earth's electrical properties at the expense of an increase in the error between the measured and predicted data.

Linearizing about a given earth model, $\mathbf{m}^{(i)}$, at a given iteration i, the following functional can provide smooth reconstructions if it is minimized with respect to the model parameters, **m**, which can include both the electrical conductivity and dielectric permittivity:

$$S=[\{\mathbf{D}[(\mathbf{d}-\mathbf{d}^{\mathrm{p}(i)})-\mathbf{A}^{\mathrm{p}(i)}(\mathbf{m}-\mathbf{m}^{(i)})]\}^{\mathrm{T}}$$
$$\{\mathbf{D}[(\mathbf{d}-\mathbf{d}^{\mathrm{p}(i)})-\mathbf{A}^{\mathrm{p}(i)}(\mathbf{m}-\mathbf{m}^{(i)})]\}-\chi^2]+\lambda(\mathbf{Wm})^{\mathrm{T}}(\mathbf{Wm}). \quad (1)$$

The terms in eq. (1) that control how well the data are fitted by the model are as follows: (1) the observed data, represented by the vector **d**; (2) the predicted data arising from the reference model $\mathbf{m}^{(i)}$ denoted by $\mathbf{d}^{\mathrm{p}(i)}$; (3) a data-weighting matrix **D**, which is diagonal and consists of the reciprocal of the data standard deviations, the reciprocal of the data amplitude or in some instances an identity matrix if data weighting is unwarranted; (4) the Jacobian or model sensitivities matrix given by $\mathbf{A}^{\mathrm{p}(i)}$; and (5) χ^2 the estimated composite noise for all the observed data. This noise can be determined by first estimating the error (standard deviation) of each data point through a series of repeated measurements, and then scaling it according to the weighting scheme employed within the inversion. To form χ^2, each individual weighted error is then squared and summed. If the measured standard deviations of the data are used as weights, then the expected value of χ^2 is the number of data employed, assuming the data errors are Gaussian-distributed with zero mean.

In addition to the above-mentioned quantities in eq. (1), T represents the transpose operator instead of the Hermitian operator, because the data, predicted data, data-weighting matrix and the Jacobian matrix have been split into real and imaginary parts, where we assume the model parameters, **m**, to be always real-valued. The parameters that control model smoothness are (1) the regularization matrix **W**, which consists of a finite-difference approximation to the Laplacian (∇^2) operator and is sparse, and (2) the trade-off parameter λ, which is used to control the amount of model smoothness in the reconstruction. Its selection requires special care if the inverse solution is to provide acceptable results. Selecting trade-off parameters that are too small can produce models that are physically unreasonable; although the models produce superior data fits they are unreasonably rough. Selecting trade-off parameters that are too large produces highly smoothed models; however, these models show poor dependence on the data. We defer further discussion of this parameter until we discuss the iterative nature of eq. (1).

Minimization of eq. (1) with respect to **m** yields the model update

$$\mathbf{m}=[(\mathbf{DA}^{\mathrm{p}(i)})^{\mathrm{T}}(\mathbf{DA}^{\mathrm{p}(i)})+\lambda(\mathbf{W})^{\mathrm{T}}(\mathbf{W})]^{-1}(\mathbf{DA}^{\mathrm{p}(i)})^{\mathrm{T}}(\mathbf{D}\boldsymbol{\delta}\mathbf{d}^{(i)}) \quad (2)$$

with

$$\boldsymbol{\delta}\mathbf{d}^{(i)}=(\mathbf{d}-\mathbf{d}^{\mathrm{p}(i)}+\mathbf{A}^{\mathrm{p}(i)}\mathbf{m}^{(i)}). \quad (3)$$

Because negative values of **m** are an admissible solution arising from eq. (2), it is advisable that before minimizing eq. (1) it should be reformulated so that one can invert for the natural logarithm of the parameters instead of the parameters themselves (Appendix A). This causes the imaged properties to be always positive, which is a physical requirement. By using a log parametrization, it is also possible to incorporate a lower-bound positivity constraint in the inverse solution.

Derivation of the Jacobian matrix elements

Deriving a computationally efficient form of the Jacobian matrix elements is critical for a robust inverse solution, since calculation and manipulation of these elements is the bottleneck within the inversion. To derive these elements, consider a single predicted data point, d_j, defined for a given frequency and transmitter–receiver pair as

$$d_j=d_j^{\mathrm{b}}+\mathbf{g}_j^{\mathrm{T}}\mathbf{E}_{\mathrm{s}}. \quad (4)$$

In this equation, d_j^{b} is a field arising from some specified uniform-space or layered-space background model at location j and $\mathbf{E}_{\mathrm{s}}$ is the scattered electric field vector arising due to 3-D changes within this background. $\mathbf{E}_{\mathrm{s}}$ has dimension $NT\times 1$, where NT represents the number of electric field unknowns that are determined from the finite-difference forward solution (Alumbaugh *et al.* 1996). The vector $\mathbf{g}_j^{\mathrm{T}}$ is an interpolator vector for the jth measurement point and is of dimension $1 \times NT$. This vector will interpolate the sampled fields on the forward-modelling grid to the measurement point, and can also be used to numerically approximate magnetic field measurements through the curl of the electric field. With this definition, an element of the Jacobian matrix is written as

$$\partial d_j/\partial m_k = \mathbf{g}_j^{\mathrm{T}} \partial \mathbf{E}_s/\partial m_k . \qquad (5)$$

From the forward problem (Alumbaugh *et al.* 1996), the scattered electric fields are determined from the linear system,

$$\mathbf{K}\mathbf{E}_s = \mathbf{s}, \qquad (6)$$

where $\mathbf{K}$ is the sparse finite-difference stiffness matrix with 13 non-zero entries per row and depends linearly on the electrical parameters we desire to estimate. Because the forward problem is formulated for the scattered fields, the source vector, $\mathbf{s}$, for a given transmitter also depends linearly on the model parameters. It is related to the difference between the model parameters and the background model, weighted by the background electric field, $\mathbf{E}^b$; refer to Alumbaugh *et al.* (1996) for the details. Thus differentiating eq. (6) with respect to m_k yields

$$\partial \mathbf{E}_s/\partial m_k = \mathbf{K}^{-1}(\partial \mathbf{s}/\partial m_k - \partial \mathbf{K}/\partial m_k \mathbf{E}_s), \qquad (7)$$

and an element of the Jacobian matrix in complex form can be written as

$$\partial d_j/\partial m_k = \mathbf{g}_j^{\mathrm{T}} \mathbf{K}^{-1}(\partial \mathbf{s}/\partial m_k - \partial \mathbf{K}/\partial m_k \mathbf{E}_s). \qquad (8)$$

Model step via conjugate gradients

As the number of unknowns increases beyond several thousand, using direct matrix inversion to compute the updated model, $\mathbf{m}$, in eq. (2) is not feasible, even with an MP platform. Instead we opt for an iterative solution. Since eq. (1) satisfies the normal equations, the conjugate-gradient method of Hestenes & Stiefel (1952) can be used to obtain the solution. This method offers a benefit over direct inversion in two ways: (1) following Mackie & Madden (1993) and Zhang *et al.* (1995) it is possible to avoid explicitly forming the Jacobian matrix, $\mathbf{A}^{\mathrm{p}(i)}$ and its transpose altogether, thus saving considerable computer memory; and (2) as the number of unknowns, n, increases, the solution for the direct inverse goes as n^3, compared with n^2 for the iterative approach. Finally, it is much easier to implement a CG routine on a parallel platform when compared to a full matrix inversion.

In the CG method all that is needed is one matrix–vector multiplication per relaxation step. However, because the matrix given by this operation is $[(\mathbf{D}\mathbf{A}^{\mathrm{p}(i)})^{\mathrm{T}}(\mathbf{D}\mathbf{A}^{\mathrm{p}(i)}) + \lambda(\mathbf{W})^{\mathrm{T}}(\mathbf{W})]$, there are several other matrix–vector multiplications to be considered. First, the matrix product of $(\mathbf{D}\mathbf{A}^{\mathrm{p}(i)})^{\mathrm{T}}$ with $\mathbf{D}\mathbf{A}^{\mathrm{p}(i)}$ requires two matrix–vector multiplications. In addition, the regularization-matrix product with its transpose requires two more matrix–vector multiplications. Since the latter matrix–vector multiplications are easy to implement and compute, no further elaboration will be given until the MP implementation of the 3-D inverse is discussed.

For the Jacobian matrix–vector multiplications, $\mathbf{D}\mathbf{A}^{\mathrm{p}(i)}$ and $(\mathbf{D}\mathbf{A}^{\mathrm{p}(i)})^{\mathrm{T}}$, we have

$$\mathbf{y} = \mathbf{D}\mathbf{A}^{\mathrm{p}(i)}\mathbf{u} \qquad (9)$$

and

$$\mathbf{z} = (\mathbf{D}\mathbf{A}^{\mathrm{p}(i)})^{\mathrm{T}}\mathbf{y}, \qquad (10)$$

where $\mathbf{u}$ is an arbitrary real vector, known as a CG search-direction vector. Because the data weighting and Jacobian matrices are real (recall that we treat real and imaginary components of the data separately), the vector $\mathbf{y}$ is real with dimension $2N$, where N is the number of complex data points used in the inversion. The vector $\mathbf{z}$ is real since the model parameters are assumed to be real-valued. We now determine compact and computationally efficient forms for the two matrix–vector multiplications. These forms will also be used to treat the matrix–vector multiplications given in eqs (2) and (3), i.e $\mathbf{A}^{\mathrm{p}(i)}\mathbf{m}^{(i)}$ and $(\mathbf{D}\mathbf{A}^{\mathrm{p}(i)})^{\mathrm{T}}(\mathbf{D}\boldsymbol{\delta}\mathbf{d}^{(i)})$, which are needed to initialize the CG solver at each iteration of the inversion. For compact programmable expressions, we let the vector $\mathbf{y}$ in eq. (9), the observed and predicted data, as well as the data weighting matrix be redefined as complex so that they can be conveniently stored in the computer memory. Using the results from Appendix B and eq. (8), we have for the jth element of the first matrix–vector multiplication

$$y_j = \mathrm{Complex}\left\{\mathscr{Re}\left[\mathbf{g}_j^{\mathrm{T}}\mathbf{K}^{-1}\sum_{k=1}^{M} u_k(\partial \mathbf{s}/\partial m_k - \partial \mathbf{K}/\partial m_k \mathbf{E}_s)\right]\mathscr{Re}(D_{jj}),\right.$$
$$\left.\mathscr{Im}\left[\mathbf{g}_j^{\mathrm{T}}\mathbf{K}^{-1}\sum_{k=1}^{M} u_k(\partial \mathbf{s}/\partial m_k - \partial \mathbf{K}/\partial m_k \mathbf{E}_s)\right]\mathscr{Im}(D_{jj})\right\}, \qquad (11)$$

where M is the total number of parameters to be estimated and D_{jj} is the jth diagonal entry of the matrix $\mathbf{D}$. $\mathbf{E}_s$ here denotes the scattered electric field arising from a given transmitter at a specific frequency used to determine the model sensitivities and predicted data at location j. Using the same approach, one can also show that, for the second matrix–vector multiplication,

$$z_k = \mathscr{Re}\left\{\sum_{j=1}^{N} \mathrm{Complex}\,[\mathscr{Re}(D_{jj})\mathscr{Re}(y_j), \mathscr{Im}(D_{jj})\mathscr{Im}(y_j)]^*\right.$$
$$\left.\times\, \mathbf{g}_j^{\mathrm{T}}\mathbf{K}^{-1}(\partial \mathbf{s}/\partial m_k - \partial \mathbf{K}/\partial m_k \mathbf{E}_s)\right\}, \qquad (12)$$

where N is the number of complex data points used in the inversion and the symbol '*' stands for complex conjugation. Note that even though the summation in eq. (12) is over all the data points, parts of the sum could be over different transmitters and/or frequencies, and hence $\mathbf{E}_s$ will change. Finally, the derivatives $\partial \mathbf{s}/\partial m_k$ and $\partial \mathbf{K}/\partial m_k$ in eqs (11) and (12) are rapid to compute analytically; it is shown in Appendix C that the vector $\partial \mathbf{s}/\partial m_k$ and matrix $\partial \mathbf{K}/\partial m_k$ each have 12 non-zero entries when m_k represents either the conductivity or permittivity.

In addition to the forward solutions necessary to determine $\mathbf{E}_s$ for each source and frequency, the matrix–vector multiplications in eqs (11) and (12) require solving a series of forward problems corresponding to the total number of unique data measurement locations, where

$$\mathbf{v}_j^{\mathrm{T}} = \mathbf{g}_j^{\mathrm{T}}\mathbf{K}^{-1}, \qquad (13)$$

or since $\mathbf{K}^{\mathrm{T}} = \mathbf{K}$ (Alumbaugh *et al.* 1996),

$$\mathbf{K}\mathbf{v}_j = \mathbf{g}_j \qquad (14)$$

(note: the fact that $\mathbf{K}$ is symmetric is simply a statement of reciprocity). A unique measurement location consists of the measurement of a specific field component made at a site. Thus the total number of forward solutions needed for each model update is given by $N_{\rm tx}+N_{\rm rx}$, where $N_{\rm tx}$ and $N_{\rm rx}$ are the total number of transmitters and unique receiver positions used in the inversion at a given frequency; note that multiple-frequency data will require additional forward solutions for both the source and unique receiver positions.

Handling the Jacobian matrix–vector multiplications in this manner is much more efficient then attempting to explicitly solve eq. (7) and using the results to form the matrix–vector multiplications. For example, if we are estimating over 30 000 parameters, this would require 30 000 separate forward solutions, which is impractical. On the other hand, because the amount of data used in the inversion is limited, we anticipate no more than several thousand forward solutions per model update. Limiting the number of forward solutions has also been recommended by McGillivray & Oldenburg (1990) and Oldenburg (1990), because of its efficiency, and has been used by Park (1983), Mackie & Madden (1993) and Zhang *et al.* (1995) in their constructions of the inverse solution.

An iterative solution and selection of the trade-off parameter

Because of the computational cost of using an exact forward solution in the inversion, we do not have the luxury of slowly reducing the trade-off parameter or determining an optimal trade-off parameter at a given iteration to ensure against arbitrarily rough models. However, experience indicates that smooth models can be produced with the strategy we are now going to discuss.

We initiate an inversion assuming an initial background model, where we compute the predicted data for all transmitter locations. At the first iteration we use our scheme to determine the matrix–vector multiplications efficiently in the CG algorithm and to determine the model update via eq. (2). This model is determined once the trade-off parameter, λ, is selected. Through extensive numerical experiments we have found that a smoothed solution can be obtained when the trade-off parameter is selected as the maximum row sum of the matrix product $[(\mathbf{DA}^{\mathrm{p}(i)})^{\mathrm{T}}\mathbf{DA}^{\mathrm{p}(i)}]$, where

$$\lambda = \max_{1\leq m\leq M}\left|\sum_{j=1}^{M} a_{mj}\right| \Big/ 2^{(i-1)}. \tag{15}$$

Here a_{mj} is an indicated element of $[(\mathbf{DA}^{\mathrm{p}(i)})^{\mathrm{T}}\mathbf{DA}^{\mathrm{p}(i)}]$ with $i=1$ for the first iteration. The above expression is easy to compute following from eqs (9) and (10) with $\mathbf{u}$ selected to be a vector with unit entries. For models we have tested, eq. (15) delivers smooth reconstructions, since weighting $(\mathbf{W})^{\mathrm{T}}(\mathbf{W})$ by λ allows only the larger eigenvalues of the non-regularized least-squares system matrix to influence the solution. This is particularly so at the early iterations.

To digress for a moment, we note that the CG method is designed for linear systems that are symmetric positive-definite. While the normal equations in eq. (2) are symmetric, both $(\mathbf{DA}^{\mathrm{p}(i)})^{\mathrm{T}}(\mathbf{DA}^{\mathrm{p}(i)})$ and $(\mathbf{W})^{\mathrm{T}}(\mathbf{W})$ possess a zero eigenvalue. Thus it appears that the matrix describing the normal equations may be semi-definite. However, when $(\mathbf{DA}^{\mathrm{p}(i)})^{\mathrm{T}}(\mathbf{DA}^{\mathrm{p}(i)})$ and $(\mathbf{W})^{\mathrm{T}}(\mathbf{W})$ are summed as $(\mathbf{DA}^{\mathrm{p}(i)})^{\mathrm{T}}(\mathbf{DA}^{\mathrm{p}(i)})+\lambda(\mathbf{W})^{\mathrm{T}}(\mathbf{W})$, experience shows the CG algorithm converges provided the trade-off parameter is reasonably selected. One must avoid selecting λ too large such that non-zero elements of $(\mathbf{W})^{\mathrm{T}}(\mathbf{W})$ are much greater than the corresponding elements of $(\mathbf{DA}^{\mathrm{p}(i)})^{\mathrm{T}}(\mathbf{DA}^{\mathrm{p}(i)})$ as this will cause a degradation of the convergence rate within the CG algorithm. We have found for the examples presented here that this problem can be avoided when eq. (15) is used to determine λ.

We proceed to the next iteration if the data error (sum of square errors) is above χ^2. If this is true, the model is linearized again about the new model $\mathbf{m}$, new predicted data and electric fields are computed from the updated background model, and the new model update determined with the trade-off parameter specified with eq. (15). In general, we have found that for the first few iterations this method of selecting the trade-off parameter reduces the error by about a factor of 2. The iterative procedure, just outlined, is continued until the data error matches χ^2, convergence of the data error occurs, or a pre-specified number of iterations has taken place.

Even with this procedure, it is possible to drive the trade-off parameter down too quickly, especially when one attempts to fit the data to an unrealistic noise level or uses an excessive number of iterations. However, it has been our experience that if the trade-off parameter is not relaxed sufficiently the inversion can stall out far above the estimated noise level. Our solution to this difficulty is to have a good estimate of the data noise, and monitor the trade-off parameter and squared error in the inversion. If excessive model structure is being incorporated into the image, or if the inversion is over-fitting the data, we stop the inversion and relaunch it using a different noise model; if the data are weighted by the noise this changes the data weighting scheme such that bad data are given less weight and good data more. While this strategy is somewhat subjective, it has yielded acceptable results.

At each iteration we restrict the number of relaxation steps in the CG routine, since only a modest number of steps are sufficient to produce an accurate model update, especially during the early stages of the scheme (Zhang *et al.* 1995). For the first and second iterations, 20 and 40 relaxation steps are used, respectively. Subsequent iterations use 60 steps.

MASSIVELY PARALLEL IMPLEMENTATION

EM inversion in 3-D can easily require the solution of at least several hundred forward solutions per iteration. Alumbaugh *et al.* (1996) demonstrate how these forward solutions can be efficiently computed on an MP machine, where each solution could constitute over five million field unknowns. A significant portion of the storage required to perform the inversion is taken up by the electric field solution vectors, which are produced by these forward solutions and are needed to complete the matrix–vector multiplications in the CG routine. Fortunately, on the 1840-node Intel Paragon at Sandia National Laboratories it is possible to execute and store all forward solutions without writing to disk; the Paragon has approximately 30 Gbytes of accessible memory.

As determined by Alumbaugh *et al.* (1996), the most efficient use of the processors is to divide the problem into, as close as possible, an equal number of unknowns for which to solve on each processor. Because each processor needs only to make calculations for a subset of the forward and inverse problems, and because the processors are making their calculations in parallel, the solution time is reduced by a factor which is approximately equal to the number of processors employed.

The parallelization of the inverse problem is achieved by assigning a given number of processors in each direction of the forward-modelling domain (*nx* in *x*, *ny* in *y* and *nz* in *z*). Hence the number of processors dedicated to the problem is determined by *nx***ny***nz*. The actual estimation of the Earth's electrical properties is carried out on the same sets of processors as dedicated to the forward problem, with all the processors sharing the same data, but storing different parts of the inversion and forward-modelling domains. However, it is possible that some of the processors may not contain portions of the inversion domain, and thus will be idle during the CG solve. The reason for this is that cells outside the inversion domain are necessary to keep the boundary of the forward-modelling domain at distance (Fig. 1). We desire parameter estimates that are not adversely affected by grid truncation errors in the forward modelling.

We now need to address the manner in which the model is input into the parallel machine. The input could constitute a starting model needed to launch the inverse or a restart model in the event of a system crash or if excessive model structure was being incorporated in the inversion. To accomplish this input, we have decomposed the data into two different sets, a global data set and a local data set (Alumbaugh *et al.* 1996). Global data are those variables that each processor needs to know, such as the source and receiver positions, the frequencies and the mesh coordinates. These form a fairly small data set which can easily be read in by a 'lead' processor and then 'broadcast' to all other processors. The second type of input is the local data, or local model parameters (electrical conductivity and dielectric permittivity) that are assigned to each cell within the model. Because each processor needs only a small subset of this data and contains only a small amount of local memory, the local data is broken up into multiple files, one for each processor, which are then read in individually from a parallel disk system which allows multiple files to be accessed simultaneously.

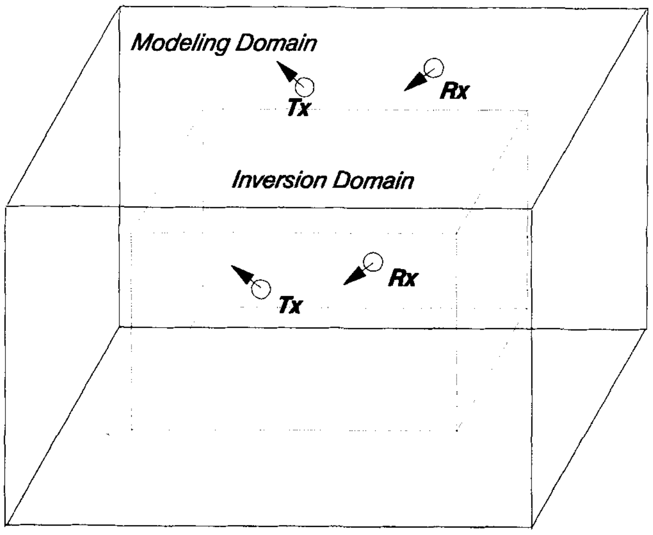


Figure 1. The inversion domain is a subset of the forward-modelling domain because of forward-modelling errors near grid boundaries. Transmitters and receivers can be placed either inside or outside the inversion domain. External transmitters and receivers could correspond to surface or airborne configurations, while internal sources and receivers could correspond to crosswell configurations.

Communication or message passing amongst the processors will be needed to complete calculations in both the forward and inverse problems. Communication amongst processors consists of both the global and local variety. Global communication is easy to implement and involves all the processors working on a given calculation, such as a dot product. In this type of calculation, each processor independently computes and sends its portion of the dot product to a lead processor, where it is then summed and broadcast across the machine since all the processors need the result. This type of communication will be required to treat the five dot products within a generic CG routine and an additional one in eq. (11). On the other hand, calculations involving the matrix–vector multiplications require local communication. Here a given processor needs to communicate with some of its neighbours to complete its local version of the computation.

Within the inversion, three types of local communication will be needed; an additional local communication is needed within the forward solve, which is discussed in Alumbaugh *et al.* (1996). The first will involve communication of electric field values on processor boundaries such that the matrix–vector products in eqs (11) and (12) can be completed. This communication will occur before the CG routine is called, for efficiency. The second type of communication will involve the CG search-direction vectors needed for the matrix–vector products involving the regularization matrix and its transpose. This occurs within the CG routine at every relaxation step, because (1) we have explicitly formulated the regularization matrix and (2) the CG vectors are constantly updated. The final type of communication occurs after exiting the CG routine. Electrical properties of cells along processor boundaries must be communicated with neighbouring processors for proper averaging of electrical properties at cell edges; these averages are needed in subsequent forward-modelling calculations. After this message passing, calculations with the forward solution can proceed with the next iteration, given the convergence criteria outlined above.

To deduce the communication pattern of the first type, consider eight nodes located at the corners of a cell whose properties we wish to estimate (Fig. 2). Consider the simplest case where each processor is in charge of only one node and cell. For example, node (i, j, k) has the cell in Fig. 2 assigned to it as well as the three components of the electric field at $(i+1/2, j, k)$ $(i, j+1/2, k)$ and $(i, j, k+1/2)$. To complete its calculations, the processor that owns this node and cell also needs the electric fields on the cell edges assigned to other nodes on different processors. These processors will thus need to supply the field components. Furthermore, the processor that owns the node (i, j, k) may also have to send its electric field components to nodes on other processors. For example, node $(i-1, j, k)$ will require the y-component of electric field assigned to node (i, j, k).

The pattern for the second type of communication can be obtained from Fig. 3. The stencil shows the required coupling between the centre cell and its neighbours arising from the Laplacian operator, as applied in the regularization matrix–vector multiplications. Again consider the case where each processor contains only a single cell. To complete its local version of the matrix–vector multiplication, the centre processor needs components of the CG search-direction vector which are assigned to the other cells and hence processors. In addition to this, the processor holding the centre cell will also

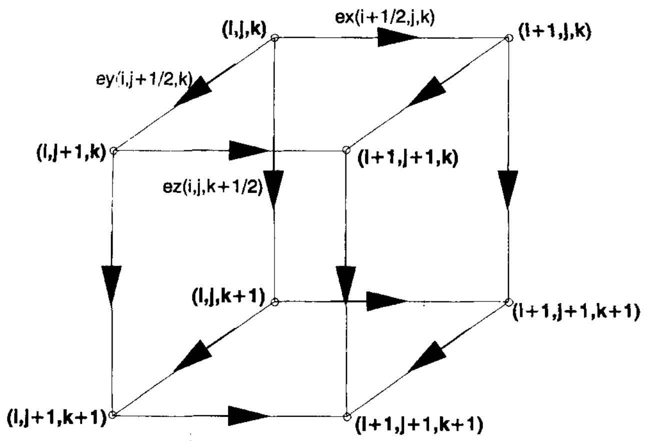


Figure 2. The electric field stencil needed to complete the Jacobian matrix–vector multiplications in the inversion algorithm for a single cell. Node (i, j, k) has the cell and the x, y and z electric field components assigned at $(i+\frac{1}{2}, j, k)$, $(i, j+\frac{1}{2}, k)$ and $(i, j, k+\frac{1}{2})$, respectively. Assignment of other electric field components to other nodes as shown in the figure follows analogously. Using results for the single cell, a processor map can be developed to carry out the required local communication amongst the processors.

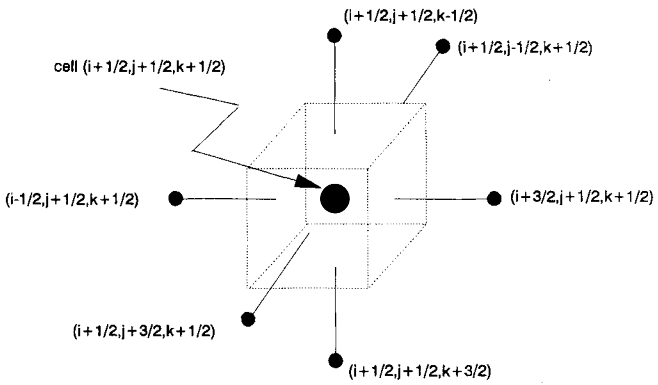


Figure 3. The stencil needed to complete local regularization matrix–vector multiplications in the CG routine. Using results for the single cell assigned to a single processor, a processor map can be developed to carry out the required local communication amongst processors.

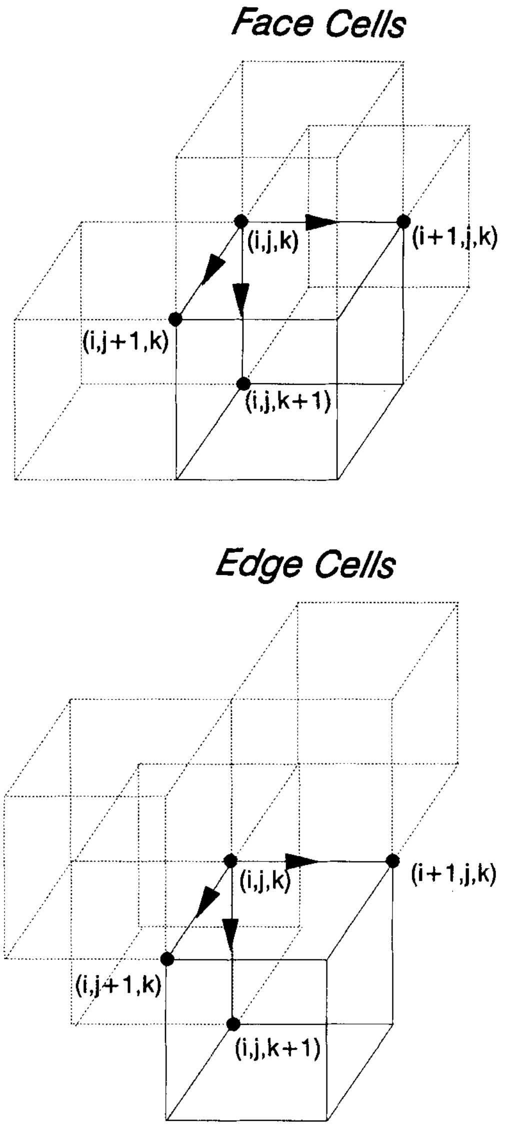
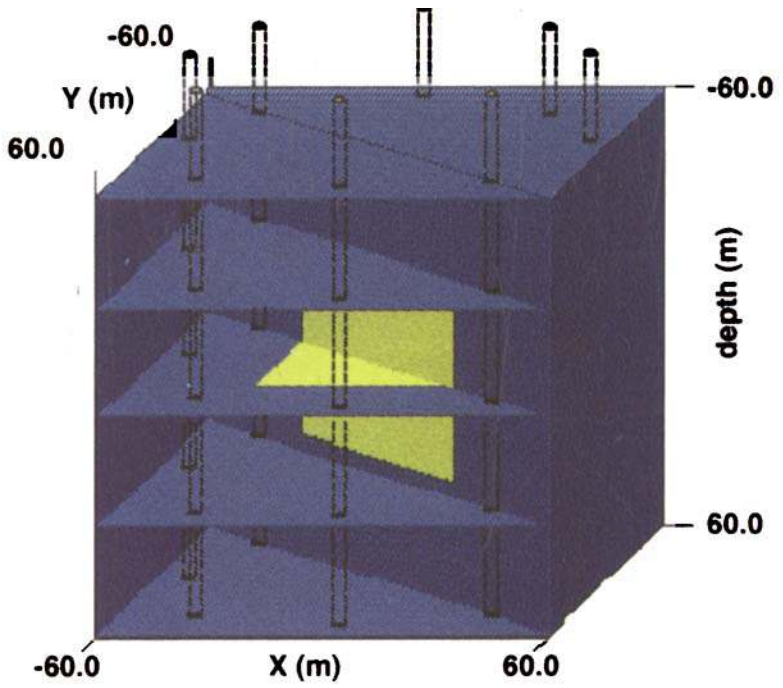


Figure 4. The different cells needed to compute average electrical properties at $(i+\frac{1}{2}, j, k)$ $(i, j+\frac{1}{2}, k)$ and $(i, j, k+\frac{1}{2})$. These edges, as well as the solid cell are assigned to node (i, j, k). The additional face and edge cells needed to compute average electrical properties are indicated by the dashed outlines. Using results for the single processor and cell, a processor map can be developed to carry out the required local communication amongst processors, necessary for subsequent forward-modelling calculations.

be required to send components to the neighbouring processors so that they can complete their corresponding computations.

From Fig. 4, the final communication pattern can be inferred. Consider the computation of the average electrical properties at cell edges $(i+1/2, j, k)$ $(i, j + 1/2, k)$ and $(i, j, k+1/2)$, which are assigned to node (i, j, k). The electrical properties of the four cells that form each edge will be needed, and the computation at these positions will be carried out on the processor that holds the solid cell also assigned to node (i, j, k); additional cells that are required are indicated by the dashed outlines. Let us now consider that each node, cell, and its associated electrical properties belong to a different processor. Since the dashed cells belong to different processors, their electrical properties need to be passed to the processor (indicated by the solid cell) that will compute the averages. In addition, this processor will be required to send its electrical properties. Consider computing average electrical properties at location $(i+1/2, j+1, k)$. Since this computation is carried out on a different processor, the electrical properties assigned to the solid cell in Fig. 4 will be needed.

The local communication pattern for the inverse problem can now be summarized in Fig. 5, where each cube represents a different processor with subsets of nodes and cells assigned to it. For the matrix–vector multiplications involving the Jacobian matrix and its transpose, communication is via the faces of processors as well as their edges. Specifically, information is passed from the central processor (marked by the heavy outline) to those neighbours that are dashed in Fig. 5. Likewise, those neighbouring processors with solid boundaries pass information to the central processor. Local communication for multiplications with the regularization matrix and its transpose involve only communication along processor



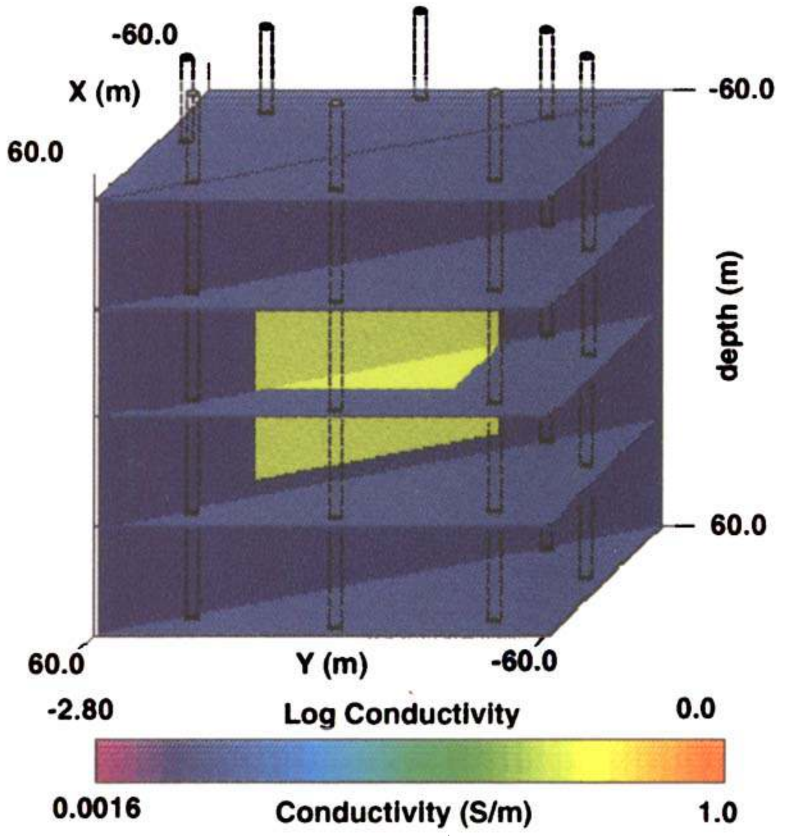
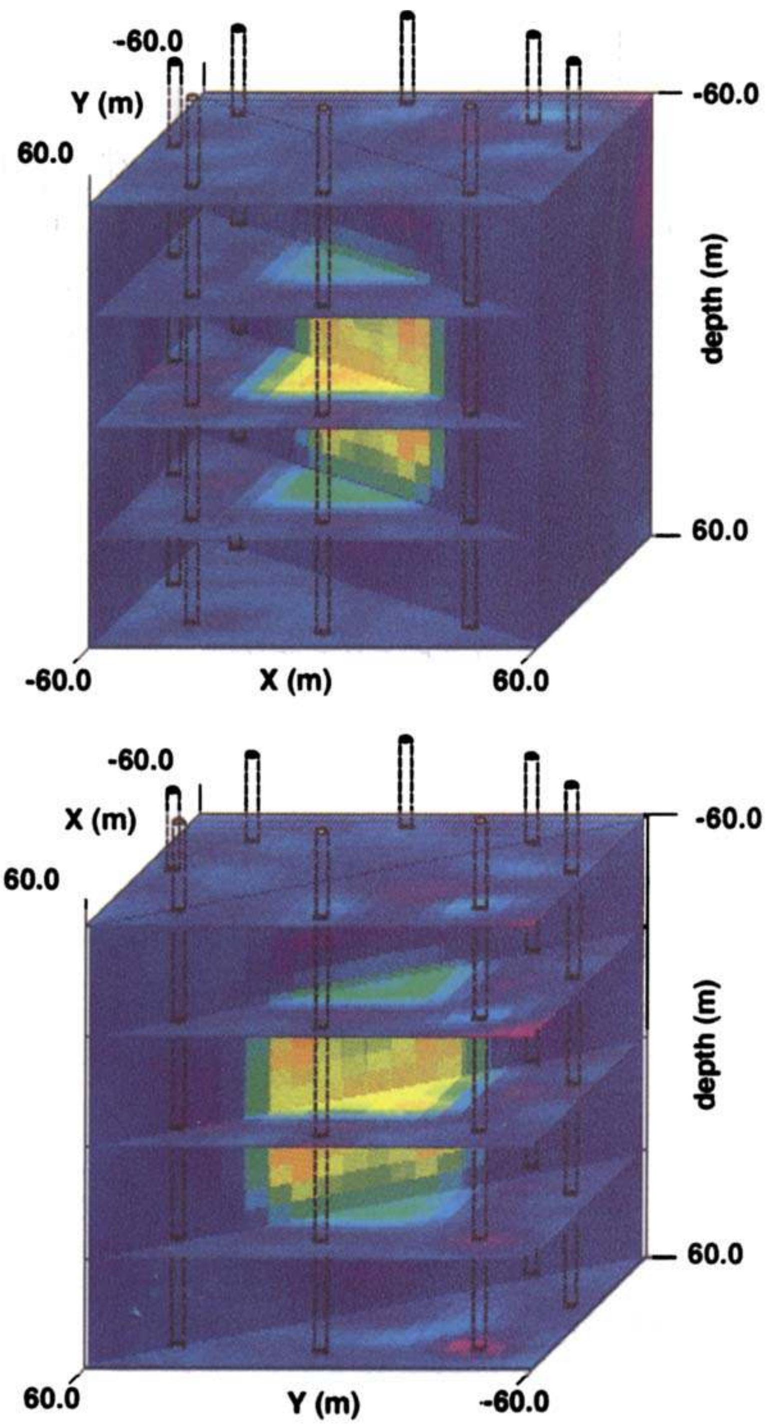


Figure 7. Synthetic example, with wellbores, used to test the inversion algorithm. The data were calculated from this model using an integral equation solution. The model is shown at two different vertical perspectives in the earth for different horizontal depth slices with the wellbores indicated. In this figure, the two shades of yellow represents the 0.2 $\mathrm{S\,m^{-1}}$ target, while two shades of dark blue represents the 0.005 $\mathrm{S\,m^{-1}}$ background. The shading is needed to render the 3-D viewing perspective.



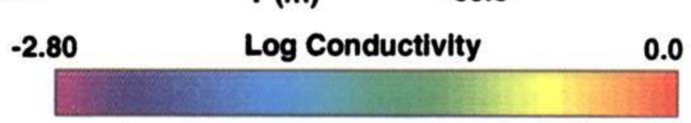
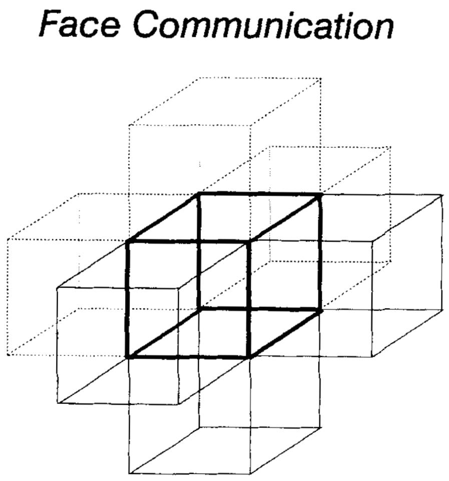


Figure 8. Reconstructed log conductivity and conductivity for the synthetic example illustrated in Fig. 7 for different perspectives and slices. The wellbores used in the simulation are again indicated. Because of the large number of conductivity estimates, a continuous colour scale is used to describe the range of conductivity and log conductivity values.

faces in Fig. 5, where all the processors send the required elements of the CG vectors to the central processor as well as receiving from it. Finally, the communication needed for averaging electrical properties of the cells at adjacent processor boundaries is in the opposite direction to that needed for the communication of the Jacobian matrix–vector multiplications. Those face and edge processors marked with a dashed outline send to the central processor, while those that are solid receive information from it.

To provide for the required message passing, we have chosen to employ 'message passing interface' (MPI; Gropp, Lusk & Skijellum 1995) instead of using machine-specific commands. MPI provides portability to the code as it will be able to run on any parallel machine and/or distributed network of machines on which this public domain library is available.

As previously mentioned, the solution time will decrease with the number of processors employed. This is demonstrated in Fig. 6 for an example described in Paper II (Alumbaugh & Newman 1996). A significant increase in speed is observed starting from 80 processors for a single iteration of the inverse algorithm. However, as the number of processors continues to increase, inter-processor communication becomes more of a factor, resulting in an asymptotic behaviour in the solution time with increasing number of processors. Here the amount of message passing will eventually limit the speed at which the computation can proceed. Put simply, increased message passing implies more time communicating and less time computing. Thus optimal use of the machine may entail running the example in Fig. 6 using 200 processors and launching several such jobs simultaneously. On the other hand, if turn-around time is an issue, one would want to operate near the far right end of the curve.



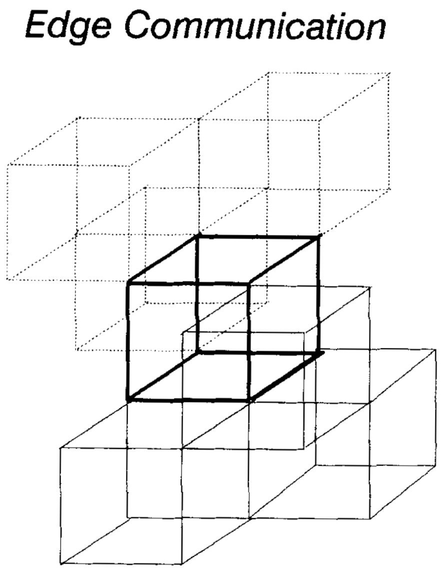


Figure 5. Local processor communication scheme used in the 3-D MP inverse. The solid cube depicts the central processor that is sending and receiving information to and from its neighbours. Both face and edge communication patterns are indicated.

SYNTHETIC EXAMPLE

Fig. 7 shows two different perspectives of a model used to test the 3-D inverse. The data from this model were generated from the integral equation solution of Newman, Hohmann & Anderson (1986), and provide a stronger check on the inversion scheme than using data generated by the staggered finite-difference code; use of data generated with the same forward code as embedded in the inverse will be prone to the same numerical errors and thus will not be fully independent. The test model consists of a 0.2 S m^{-1} cube, with sides of 50 m, residing in a 0.005 S m^{-1} background. Eight wells surround the target, with 15 vertical magnetic dipole (VMD) transmitters at 10 m intervals straddling the target. The vertical magnetic fields were calculated in all other wells at 10 m intervals, excluding the transmitter well. Because the frequency of excitation used in this test is only 20 kHz, the dielectric properties of the target and host are not important in the simulation,

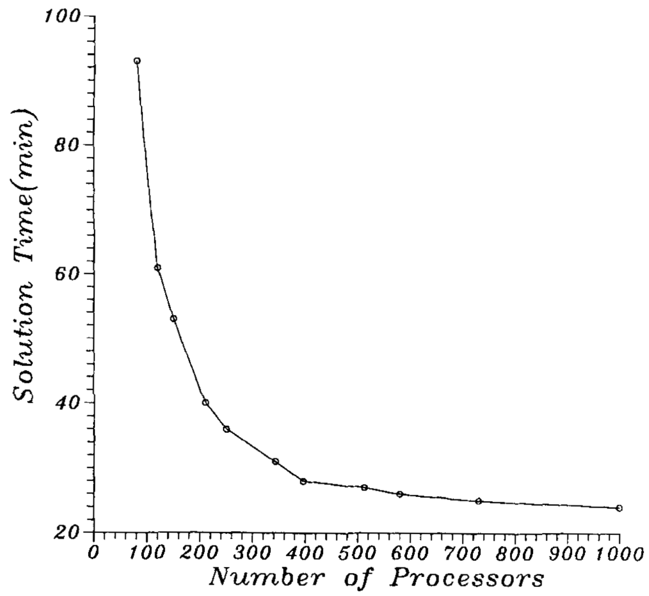


Figure 6. Solution time for one iteration of the inversion versus the number of processors employed. Results are for the eight-well Richmond model used in the design experiment discussed in Paper II. The model is discretized at 114 000 cells for the forward model calculations, with the inverse parametrization using 88 200 cells. The total number of transmitter–receiver pairs used in the inversion is 1848.

and only the conductivity properties need be estimated; the magnetic permeability is assumed constant throughout the model and set to that of free space. Gaussian noise equal to two per cent of the data amplitude was added to the data set. The data were then weighted by this percentage before inversion. In total, the data consist of 12 600 transmitter–receiver pairs.

The inversion domain consists of 29 791 cells, but only 13 824 cells are shown in the inter-well region in Figs 7 and 8; cells outside this region are used to keep the boundary of the inversion domain at distance, so as to not affect the conductivity estimates within the inter-well region. We assume in this synthetic example that the structure is sufficiently deep for the frequency employed, such that the inversion can be launched assuming a 0.005 $\mathrm{S\,m^{-1}}$ whole space. Note that if the structure is sufficiently close to the Earth's surface then the effect of the air–earth interface has to be included in the inversion. Incorporation of this interface is possible for the formulation presented here, but this could cause the inversion to run more slowly and is not critical for this example. Nevertheless, the synthetic example is realistic since it is often possible to neglect the air–earth interface with borehole data, as demonstrated in Paper II.

The image in Fig. 8 has recovered the location and geometry of the cube fairly well, but a smeared version of its conductivity within the cube boundary; the estimates vary from 0.1 to 0.75 $\mathrm{S\,m^{-1}}$. The conductivity estimates of the background range as low as 0.0016 $\mathrm{S\,m^{-1}}$. It has been our experience that improved resolution of the background and cube can be obtained by tightening the lower-bound positivity constraint. In this example, the conductivity estimates were restricted to be greater than 0.001 $\mathrm{S\,m^{-1}}$.

Fifteen iterations were needed to obtain this reconstruction, where the reduction in the normalized squared error against iteration count is illustrated in Fig. 9. For our purposes, the normalized squared error is defined by

$$e_n^2 = [\mathbf{D}(\mathbf{d}-\mathbf{d}^p)]^T\mathbf{D}(\mathbf{d}-\mathbf{d}^p)/2N\,, \tag{16}$$

where $\mathbf{d}$ and $\mathbf{d}^p$ are the observed and predicted data and N is the total number of complex data used in the inversion; in this expression we treat the real and imaginary components of the data separately. Notice that after the 15th iteration the error begins to increase; in a non-linear inverse problem there is no guarantee that an iterative technique will always reduce the error (see Menke 1984, pp. 154–156).

Assuming Gaussian noise with zero mean, the inversion is assumed to have converged when the normalized squared error approaches the value of unity, since we have weighted the data by the noise. Because the error level is still above one in Fig. 9, this might suggest that more information could be extracted from the data. However, we assume that the error level originates from bias in the data. These data were produced from a forward-modelling algorithm that is different from the one used in the inverse. Finally, the processing time needed to produce the image in Fig. 8 was approximately 31 hr on the Paragon, with 512 processors utilized.

DISCUSSION

The MP inversion scheme we have presented has been demonstrated on data sets that are impossible to invert on scalar workstation platforms, due to the limitations in memory and processor speeds (refer to Paper II for additional examples). An important question to ask is what the largest model the MP inversion can handle is. Certainly the maximum model size (both forward and inverse) will be related to the number of transmitters and receivers specified in the data set, because this will determine the number of electric field vectors, $\mathbf{E}_s$, that need to be computed and stored. Given a maximum memory on the Intel Paragon of 16 Mbytes per processor, and considering a problem divided amongst 1728 processors (this corresponds to 12 processors assigned along each coordinate direction), Table 1 illustrates a range of problem sizes that can be effectively handled. If 120^3 nodes are used to describe the forward- and inverse-modelling domain, the total number of transmitters and receivers that can be used is 700. To increase the number of transmitters and receivers it appears that it is necessary to reduce the number of nodes.

Table 1. Maximum problem size that can be treated by the Intel Paragon assuming 1728 processors. Problem size is determined by the number of cells used in the forward modelling and inversion and the number of transmitters (Tx's) and unique receivers (Rx's) specifying a data set. Each Tx and Rx position is for a unique frequency.

Problem size (nodes):	120^3	96^3	72^3
No. Tx's and Rx's:	700	1300	3000

One way to increase the size of inverse problems that can be tackled is to use a coarser parametrization for the inverse problem, but retain the finer parametrization level for the forward problem. The key idea here is to reduce the storage of the electric field vectors needed in the inverse. For a given source, the electric field and predicted data are computed at the parametrization level specified in the forward modelling. The electric field is then interpolated to the coarser or skeletonized grid corresponding to the inverse and stored in the memory. Hence the forward-modelling accuracy is still retained in the inverse. Note that the coarser grid can still produce smooth images, since it can involve tens of thousands to hundreds of thousands of cells.

The skeletonized electric field vectors allow for the number of transmitters and receivers to increase dramatically. Consider a problem where the inversion grid is eight times coarser than the forward-modelling grid. If 120^3 nodes are used in the forward calculations, the skeletonized inversion grid, which still comprises 216 000 cells, allows for the number of transmitters and receivers to increase from 700 to over 3000.

CONCLUSIONS

A 3-D EM inversion code has been successfully implemented and tested on an MP platform. Reasonable, overnight to one-to-two day, processing times have been obtained. Because of the MP platform, reconstructions have been produced that do not underparametrize the Earth; these are reconstructions that involve tens of thousands of cells. Since the 3-D MP inverse also includes rigorous 3-D forward modelling for computing model sensitivities and predicted data, it is our hope that this solution will also serve as an accuracy benchmark on approximate inverse methods now being implemented on workstation platforms (see Torres-Verdin & Habashy 1995, 1994; Zhdanov & Fang 1995; Habashy *et al.* 1995; Farquharson & Oldenburg 1995).

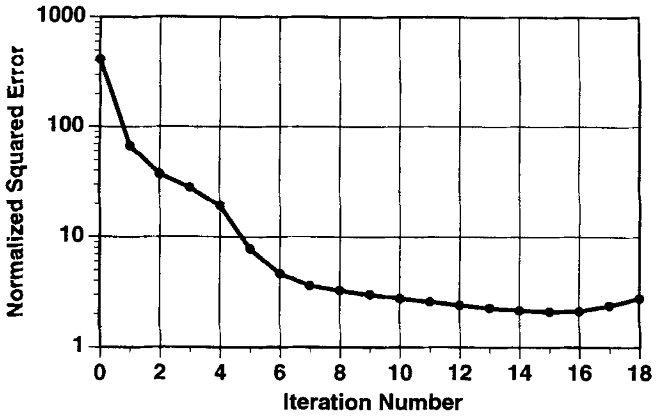


Figure 9. The normalized squared error is plotted against iteration number for the 5 Ωm test body shown in Fig. 7. Ideally the squared error should approach 1 for convergence. Its failure to do so indicates that the data are biased.

In this paper, we have presented the theory and demonstrated the 3-D inversion capability on synthetic data. Because the ultimate goal of any inversion scheme is to use it to image field data, in Paper II we demonstrate how this scheme can be used to design a 3-D crosswell survey and invert a crosswell data set collected at the Richmond field station north of Berkeley, California. Images before and after the injection of a salt-water plume will be compared to determine the location of the injected plume. In addition, we will also show how the scheme can be employed to analyse the reliability of the images as well as the accuracy and errors in the data.

ACKNOWLEDGMENTS

This work was performed at Sandia National Laboratories, which is operated for the US Department of Energy (DOE). Funding for this project was provided by (1) the DOE's office of Basic Energy Sciences, Division of Engineering and Geoscience under contract DE-AC04-94AL85000, (2) the Sandia National Laboratories Director's Research funds and (3) the VETEM project, which is supported by the Landfill Stabilization Focus Area of the DOE Office of Technology and Development.

REFERENCES

Alumbaugh, D.L. & Newman, G.A., 1996. Three-dimensional massively parallel inversion—II. Analysis of a cross-well EM experiment, *Geophys. J. Int.*, **128**, 355–363 (this issue).

Alumbaugh, D.L., Newman, G.A., Prevost, L. & Shadid, J.N., 1996. Three-dimensional wide band electromagnetic modeling on massively parallel computers, *Radio Science,* **31,** 1–23.

Farquharson, C.G. & Oldenburg, D.W., 1996. Approximate sensitivities for the electromagnetic inverse problem, *Geophys. J. Int.,* **126,** 235–252.

Gropp, W., Lusk, E. & Skjellum, A., 1995. *Using MPI–portable parallel programming with the message-passing interface,* MIT Press, Cambridge, MA.

Habashy, T.M., Torres-Verdin, C. & Oristaglio, M.L., 1995. An overview of recent advances in inversion of large-scale electromagnetic data, in *Int. Symp. on Three-Dimensional Electromagnetics,* expanded abstracts.

Hestenes, M.R. & Stiefel, E., 1952. Methods of conjugate gradients for solving linear systems, *J. Res. Nat. Bur. Standards,* **49,** 409–435.

McGillivray, P.R. & Oldenburg, D.W., 1990. Methods for calculating Fréchet derivatives and sensitivities for the non-linear inverse problem, *Geophys. Prospect.,* **38,** 499–524.

Mackie, R.L. & Madden, T.R., 1993. Three-dimensional magnetotelluric inversion using conjugate gradients, *Geophys. J. Int.,* **115,** 215–229.

Menke, W., 1984. *Geophysical Data Analysis: Discrete Inverse Theory,* Academic Press, Inc., Orlando, FL.

Newman, G.A., Hohmann, G.W. & Anderson, W.L., 1986. Transient electromagnetic responses of a three-dimensional body in a layered earth, *Geophysics,* **51,** 1608–1627.

Oldenburg, D.W., 1990. Inversion of electromagnetic data: an overview of new techniques, *Surv. Geophys.,* **11,** 231–270.

Park, S.K., 1983. Three-dimensional magnetotelluric modeling and inversion, *PhD thesis,* Massachusetts Institute of Technology, MA.

Tikhonov, A.N. & Arsenin, V.Y., 1977. *Solutions to Ill-posed Problems,* John Wiley and Sons, New York, NY.

Torres-Verdin, C. & Habashy, T.M., 1994. Rapid 2.5-dimensional forward modeling and inversion via a new nonlinear scattering approximation, *Radio Sci.,* **29,** 1051–1079.

Torres-Verdin, C. & Habashy, T, M., 1995. A two step linear inversion of two dimensional electrical conductivity, *IEEE Trans. Antenna Propagat.,* **43,** 405–415.

Zhang, J. Mackie, R.L. & Madden, T.R., 1995. Three-dimensional resistivity forward modeling and inversion using conjugate gradients, *Geophysics,* **60,** 1313–1325.

Zhdanov, M.S. & Fang, S., 1995. 3D electromagnetic inversion based on the quasi-linear approximation, in *Int. Symp. on Three-Dimensional Electromagnetics,* expanded abstracts.

APPENDIX A: FORMULATION OF THE INVERSE PROBLEM USING LOG PARAMETERS

The inverse problem can be formulated to allow for positive parameters with a lower bounding constraint by using a log parametrization. To accomplish this we first define a perturbation in a given earth model $\mathbf{m}^{(i)}$ at a given iteration i. For cell k, we can write

$$\delta m_k = (m_k - m_k^{(i)}), \tag{A1}$$

where m_k is the updated model component we are seeking. We then use a Taylor series to expand the natural log function, $\ln(m_k - \varepsilon_k)$, about the point $m_k^{(i)}$ with lower bounding constraint ε_k such that both m_k and $m_k^{(i)} > \varepsilon_k$, and $\varepsilon_k > 0$ to write

$$\ln(m_k - \varepsilon_k) = \ln(m_k^{(i)} - e_k) + (m_k^{(i)} - \varepsilon_k)^{-1}\delta m_k, \tag{A2}$$

where second-order terms have been neglected. Thus, we arrive at

$$\delta m_k = (m_k^{(i)} - \varepsilon_k)\delta \ln(m_k - \varepsilon_k), \tag{A3}$$

where

$$\delta \ln(m_k - \varepsilon_k) = \ln(m_k - \varepsilon_k) - \ln(m_k^{(i)} - \varepsilon_k). \tag{A4}$$

One can set up an inverse problem involving log parameters by modifying elements of the original Jacobian matrix using eq. (A3) and the chain rule, such that

$$\partial d_j/\partial m_k' = \partial d_j/\partial m_k(m_k^{(i)} - \varepsilon_k), \tag{A5}$$

where $m_k' = \ln(m_k - \varepsilon_k)$ and where $\partial d_j/\partial m_k$ is evaluated at $m_k^{(i)}$. Following the form of eq. (1) we can define a new functional,

$$S' = [\{\mathbf{D}[(\mathbf{d} - \mathbf{d}^{\mathrm{p}(i)}) - \mathbf{A}'^{\mathrm{p}(i)}\delta\mathbf{m}']\}^{\mathrm{T}}\{\mathbf{D}[(\mathbf{d} - \mathbf{d}^{\mathrm{p}(i)}) - \mathbf{A}'^{\mathrm{p}(i)}\delta\mathbf{m}']\}$$
$$- \chi^2] + \lambda(\mathbf{W}\mathbf{m}')^{\mathrm{T}}(\mathbf{W}\mathbf{m}'), \tag{A6}$$

where elements of the modified Jacobian matrix $\mathbf{A}'^{\mathrm{p}(i)}$ are given by eq. (A5), and the perturbation vector $\boldsymbol{\delta}\mathbf{m}'$ is defined as

$\mathbf{m}'-\mathbf{m}'^{(i)}$. Minimizing the above expression with respect to $\mathbf{m}'$, enforces the lower-bound positivity constraint, where

$$\mathbf{m}'=[(\mathbf{DA}'^{\mathrm{P}(i)})^{\mathrm{T}}(\mathbf{DA}'^{\mathrm{P}(i)})+\lambda(\mathbf{W})^{\mathrm{T}}(\mathbf{W})]^{-1}(\mathbf{DA}'^{\mathrm{P}(i)})^{\mathrm{T}}(\mathbf{D}\boldsymbol{\delta}\mathbf{d}'^{(i)}) \tag{A7}$$

and

$$\boldsymbol{\delta}\mathbf{d}'^{(i)}=(\mathbf{d}-\mathbf{d}^{\mathrm{P}(i)}+\mathbf{A}'^{\mathrm{P}(i)}\mathbf{m}'^{(i)}). \tag{A8}$$

Once $\mathbf{m}'$ is determined, the parameter components follow from the expression

$$m_k=\mathrm{e}^{m'_k}+\varepsilon_k. \tag{A9}$$

With this new formulation, the inversion process is designed to deliver smooth estimates of $\mathbf{m}'$. Nevertheless, with a prudent selection of the regularization parameter, we can also expect smooth reconstructions for the model parameters, $\mathbf{m}$, themselves.

APPENDIX B: DERIVATION OF THE JACOBIAN MATRIX–VECTOR MULTIPLICATIONS

Consider fully expressing the matrix–vector multiplication in eq. (9) as

$$y_j=\sum_{k=1}^{M} D_{jj}A_{jk}u_k, \tag{B1}$$

where the summation is over M electrical parameters. The entry D_{jj} is the jth entry of the data-weighting matrix and A_{jk} is an element of the Jacobian matrix. These elements are assumed to be real-valued, since the real and imaginary components of the data are treated as separate entries. The index j ranges from 1 to $2N$, where N is the number of complex data used in the inversion. Thus $j=1, N$ correspond to real entries, while components $j=N+1, 2N$ correspond to imaginary ones. The second matrix–vector multiplication in eq. (10) can be expressed as

$$z_k=\sum_{j=1}^{2N} A_{jk}D_{jj}y_j, \tag{B2}$$

By associating real and imaginary components as a joint term in the above summation, we can also express eq. (B2) as

$$z_k=\sum_{j=1}^{N} (A_{jk}D_{jj}y_j+A_{j+N\,k}D_{j+N\,j+N}\,y_{j+N}). \tag{B3}$$

Next, combining elements as $\mathrm{Complx}(A_{jk,}\,A_{j+N\,k})$, $\mathrm{Complx}(D_{jj}, D_{j+N\,j+N})$ and $\mathrm{Complx}(y_j, y_{j+N})$ and because z_k must always be real, we find

$$z_k=\mathscr{Re}\sum_{j=1}^{N} \mathrm{Complx}(D_{jj}y_j,\, D_{j+N\,j+n}\,y_{j+n})^* \times \mathrm{Complx}(A_{jk},\, A_{j+N\,k}), \tag{B4}$$

where '*' stands for complex conjugation.

Because the real and imaginary components of the Jacobian matrix are jointly expressed in eq. (8) as

$$\partial d_j/\partial m_k=\mathbf{g}_j^{\mathrm{T}}\mathbf{K}^{-1}(\partial \mathbf{s}/\partial m_k-\partial\mathbf{K}/\partial m_k\mathbf{E}_{\mathrm{s}}), \tag{B5}$$

we can redefine A_{jk}, D_{jj} and y_j as complex so that they can be conveniently stored in computer memory to arrive at the following compact programmable expressions for the two matrix–vector multiplications:

$$y_j=\mathrm{Cmplx}\left\{\mathscr{Re}\left[\mathbf{g}_j^{\mathrm{T}}\mathbf{K}^{-1}\sum_{k=1}^{M} u_k(\partial \mathbf{s}/\partial m_k-\partial\mathbf{K}/\partial m_k\mathbf{E}_{\mathrm{s}})\right]\mathscr{Re}(D_{jj}),\ \mathscr{Im}\left[\mathbf{g}_j^{\mathrm{T}}\mathbf{K}^{-1}\sum_{k=1}^{M} u_k(\partial \mathbf{s}/\partial m_k-\partial\mathbf{K}/\partial m_k\mathbf{E}_{\mathrm{s}})\right]\mathscr{Im}(D_{jj})\right\}, \tag{B6}$$

and

$$z_k=\mathscr{Re}\left\{\sum_{j=1}^{N} \mathrm{Complx}[\mathscr{Re}(D_{jj})\mathscr{Re}(y_j),\, \mathscr{Im}(D_{jj})\mathscr{Im}(y_j)]^* \times[\mathbf{g}_j^{\mathrm{T}}\mathbf{K}^{-1}(\partial \mathbf{s}/\partial m_k-\partial\mathbf{K}/\partial m_k\mathbf{E}_{\mathrm{s}})]\right\}. \tag{B7}$$

APPENDIX C: PROOF ON THE SPARSITY OF THE FINITE-DIFFERENCE STIFFNESS MATRIX AND SOURCE VECTOR DERIVATIVES

To show that the vector $\partial\mathbf{s}/\partial m_k$ and matrix $\partial\mathbf{K}/\partial m_k$ each has 12 non-zero entries, we start with the vector Helmholtz equation for the scattered electric field, $\mathbf{E}_{\mathrm{s}}$ (eq. 1 in Alumbaugh *et al.* 1996), but we will modify it such that magnetic permeability changes from free space, $\mu_{0,}$ are minimal. Thus,

$$\nabla\mathrm{x}\nabla\mathrm{x}\mathbf{E}_{\mathrm{s}}(\mathbf{r})+i\omega\mu_0(\sigma(\mathbf{r})+i\omega\varepsilon(\mathbf{r}))\mathbf{E}_{\mathrm{s}}(\mathbf{r})=-i\omega\mu_0\mathbf{J}_{\mathrm{s}}(\mathbf{r}), \tag{C1}$$

with the source of the scattering given by

$$\mathbf{J}_{\mathrm{s}}(\mathbf{r})=\{[\sigma(\mathbf{r})-\sigma^{\mathrm{b}}(\mathbf{r})]+i\omega[\varepsilon(\mathbf{r})-\varepsilon^{\mathrm{b}}(\mathbf{r})]\}\,\mathbf{E}^{\mathrm{b}}(\mathbf{r}). \tag{C2}$$

Here we have assumed an $\exp(i\omega t)$ time dependence with $i=\sqrt{-1}$, where ω represents the angular frequency. In eqs (C1) and (C2), the 3-D conductivity and permittivity variations are given by $\sigma(\mathbf{r})$ and $\varepsilon(\mathbf{r})$, with $\sigma^{\mathrm{b}}(\mathbf{r})$ and $\varepsilon^{\mathrm{b}}(\mathbf{r})$ representing the corresponding background properties, which for the present purposes are either a uniform space or a layered space. The electric field of the background media, $\mathbf{E}^{\mathrm{b}}(\mathbf{r})$, drives the source vector, and arises from an impressed dipole source.

The scattered fields are determined by imposing a staggered finite-difference approximation on eq. (C1), using a rectangular grid with a Dirichlet boundary condition. Each cell in this grid has a conductivity and dielectric permittivity assigned to it, where the scattered and source fields are sampled at the edges of the cell as illustrated in Fig. 2. Because of this sampling scheme, the averaged electrical properties have to be determined at the cell edges (*cf.* Alumbaugh *et al.* 1996). These averages can be evaluated by tracing out a line integral of the magnetic field centred on the midpoint of the cell edge. The resulting average conductivity and permittivity are simply a weighted sum of the conductivities and permittivities of the four adjoining cells, where the weighting is based on the area of each cell that is bounded by the line integral. This is a simple application of Ampere's Law. A study of Fig. 2 shows that, with the 12 field samples, eqs (C1) and (C2) will require 12 averages of conductivity and permittivity, with each average involving the conductivity and permittivity of the indicated cell. Since with every field sample, we have one equation in the linear system, $\mathbf{KE}_{\mathrm{s}}=\mathbf{s}$, where $\mathbf{s}=\mathbf{J}_{\mathrm{s}}$, it follows that $\partial\mathbf{s}/\partial m_k$ and the matrix $\partial\mathbf{K}/\partial m_k$ each have 12 non-zero entries.